Available online at www.sciencedirect.com
ScienceDirect
Information Fusion xxx (2007) xxx–xxx
INFORMATION FUSION
www.elsevier.com/locate/inffus

Using classifier ensembles to label spatially disjoint data

Larry Shoemaker [a,*], Robert E. Banfield [a], Lawrence O. Hall [a], Kevin W. Bowyer [b], W. Philip Kegelmeyer [c]

[a] Department of Computer Science and Engineering, ENB118, University of South Florida, 4202 E. Fowler Avenue Tampa, FL 33620-9951, USA
[b] Department of Computer Science and Engineering, University of Notre Dame, South Bend, IN 46556, USA
[c] Sandia National Laboratories, Computational Science and Math Research Department, P.O. Box 969, MS 9159 Livermore, CA 94551-0969, USA

Received 2 March 2007; received in revised form 26 July 2007; accepted 1 August 2007

Abstract

We describe an ensemble approach to learning from arbitrarily partitioned data. The partitioning comes from the distributed processing requirements of a large scale simulation. The volume of the data is such that classifiers can train only on data local to a given partition. As a result of the partition reflecting the needs of the simulation, the class statistics can vary from partition to partition. Some classes will likely be missing from some partitions. We combine a fast ensemble learning algorithm with probabilistic majority voting in order to learn an accurate classifier from such data. Results from simulations of an impactor bar crushing a storage canister and from facial feature recognition show that regions of interest are successfully identified in spite of the class imbalance in the individual training sets.

Keywords: Random forest; Saliency; Probabilistic voting; Out-of-partition; *k*-Nearest centroids; Imbalanced training data

1. Introduction

We consider the problem of dealing with data sets too large to fit in the memory of any one computer node and too bandwidth-intensive to move between neighboring nodes [1]. In essence, there is no practical partitioning of the data other than that originally used by the simulation model that generated the data. Such problems exist in the United States Department of Energy's Advanced Simulation and Computing (ASC) program [2,3], wherein a supercomputer simulates a hypothetical real-world event. The simulation data is partitioned and distributed across separate disks, to facilitate parallel computation. The storage allocation for the simulation optimizes for balanced and efficient computation of the simulation, without regard to conditions that might make it easy or difficult for a machine learning algorithm to use the resulting data.

In analyzing the results of these simulations, developers and analysts want to find specific phenomena that may take days to find by manually visualizing and browsing a massive simulation. Developers are interested in anomalies in general. Analysts are similarly often interested in phenomena which, like an anomaly, may be easier to recognize than describe. Therefore, manually marking some example areas of interest and automatically finding others in the same or similar types of simulations can greatly reduce debugging and analysis time.

Learning from massive amounts of data has been the subject of various research projects [4–6]. The majority of current research focuses on how to distribute learning tasks across multiple processors. In nearly all cases, existing data mining algorithms have been tweaked in order to accomplish this. Modifications typically are done by preprocessing the data, as in bagging or random subspaces, or in the core algorithm, as in random forests. Our inability to

* Corresponding author.
E-mail addresses: lwshoema@cse.usf.edu (L. Shoemaker), rbanfiel@cse.usf.edu (R.E. Banfield), hall@cse.usf.edu (L.O. Hall), kwb@cse.nd.edu (K.W. Bowyer), wpk@sandia.gov (W.P. Kegelmeyer).

1566-2535/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.inffus.2007.08.001

migrate data prohibits the former. Instead, we focus our attention on the later, additionally concentrating on the fusion of the individual classifiers.

In this paper, we give examples of learning from several simulations of a storage canister being crushed by an impactor bar. We performed separate experiments using vertical and horizontal partitionings of the canister. An illustration of the simulation model with vertical partitions appears in Fig. 1, where the different shades of gray represent the partitioning of the simulation in a distributed environment. While the impactor bar is also broken up spatially for vertical partitions, the impactor bar data is not used in our experiments. A visualization of the horizontal partitions is shown in Fig. 2. Table 1 provides data on the number of examples in each partition. Partitions are more uniformly sized for the horizontal case than for the vertical case.

As a result of partitioning, the points of interest, or "salience", in some partitions may be limited to only a few nodes. Salient points, being few in number, exhibit a pathological minority class classification problem. The problems associated with imbalanced datasets and various strategies for dealing with those problems are described in [7,8]. Techniques include various forms of undersampling and oversampling as in [9], and cost-sensitive learning methods as in [10]. In the case of a partition having zero salient points, a single-class "classifier" will be learned. This motivated an adjustment to our voting scheme for improved accuracy, as shown in Section 5.

Table 1
Canister nodes for partitions (each time step)

Type	Simulation	Partition			
		0	1	2	3
Vertical	1–4	1640	1886	1886	1312
Horizontal	1–4	1640	1640	1640	1804

As both designers and simulation users are most interested in finding a salient region rather than individual salient nodes, we have evaluated how well our approach can detect connected groups of salient nodes as well as how accurate it is for individual nodes. We show that it is possible to obtain an accurate prediction of salient points, and more crucially, regions, even when the data is broken up arbitrarily in 3D space with no particular relation to feature space. To explore whether the idea generalizes to other sorts of data, we also show examples of ensembles of classifiers trained on partitioned face image data to learn interesting regions. Results on the canister and face image data sets indicate that experts working with much larger simula-

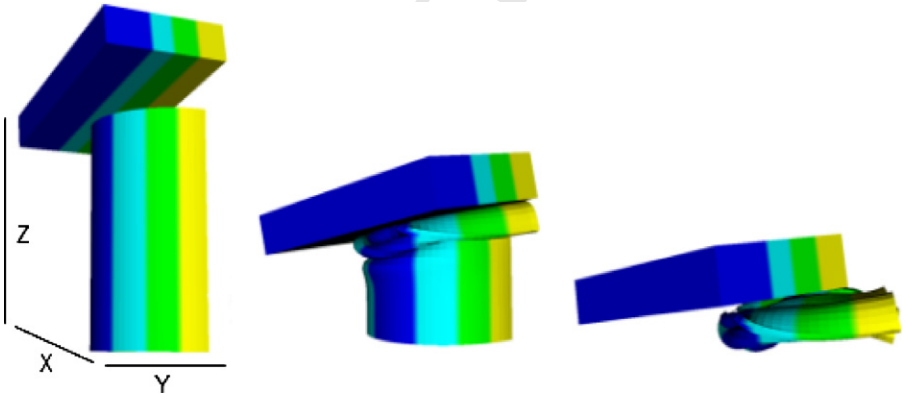

Fig. 1. A visualization of the data as distributed across compute nodes for vertical partitions. Four partitions are shown in different gray levels as the storage canister is crushed. Partitions 0–3 in numerical order are shown from right to left in each view.

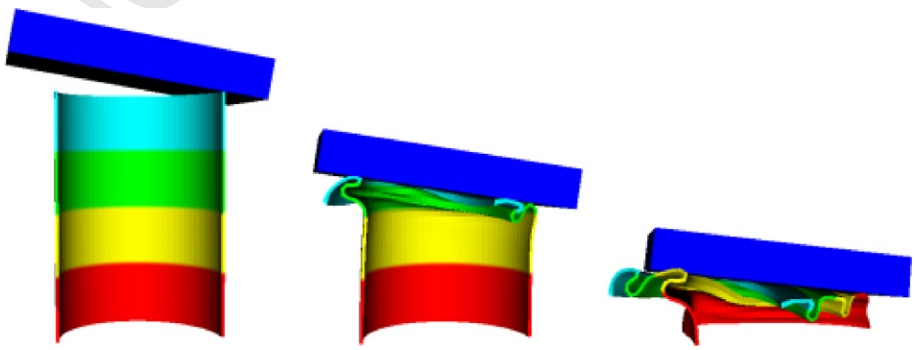

Fig. 2. A visualization of the data as distributed across compute nodes for horizontal partitions. Four canister partitions and an impactor bar partition are shown in different gray levels as the storage canister is crushed. Partitions 0–3 in numerical order from top to bottom are beneath the impactor bar in the left view.

Please cite this article in press as: L. Shoemaker et al., Using classifier ensembles to label spatially disjoint data, Informat. Fusion (2007), doi:10.1016/j.inffus.2007.08.001

tions can benefit from the predictive guidance obtained from only a small amount of relevant data.

2. Related work

Incremental learning [11–14], where the model changes as training data becomes available over time, provides a potential approach for creating a model from a very large training data set. The model could be built on one set of data and then moved to another processor for continued learning on a second set of data, etc. Incremental learning models that require the storage of previous training examples, such as instance-based learning approaches [12], and decision tree approaches [5] are time consuming for very large data sets. Also, we could find no work evaluating their performance on very large data sets. Alternatively, data mining of streaming data [4,15] has been developed precisely for endless streams of data. The data sets considered in this paper could be treated as a stream. This is, potentially, an approach that could be adapted.

There are distributed learning algorithms, such as distributed boosting [6], that could be applied in this problem. In [16], several distributed boosting algorithms are evaluated, one of which deals specifically with learning from homogeneous distributions of data scattered between different data sites. They consider the problem from the standpoint of data privacy, where data examples may not be propagated to other computers. In this algorithm, they compute statistics on the data such as mean and covariance in order to calculate the Mahalanobis distance between sites. Sites containing similar distributions employ the authors' distributed boosting algorithm, while those without similarity use standard boosting.

In the distributed boosting algorithm, a boosted classifier was built in each partition and broadcast to the other partitions. Using this ensemble of classifiers, the weight of each example was updated. A global weight array stores the sum of the updated weights for each individual site, thus providing information on how difficult it is to learn at any one site, and weighting that partition accordingly for the next iteration. The authors showed this algorithm was at least as accurate as standard boosting on the centralized data base. The only spatially disjoint sets used in [16] were two very small synthetic data sets with three equal size classes, two physical dimensions, and no time dimension. In contrast, our much larger canister data sets simulate real world events with unequal size classes, three physical dimensions, a time dimension, and different partition schemes that present unique data mining challenges.

Distributed learning models have been shown to be able to provide classification performance that is competitive with that obtained on all of the data [17]. There is some work that indicates it is possible to do effective distributed learning with cost sensitive data [18]. Further, any approach that builds independent classifiers or models and combines them could potentially be applied [19]. Of the work discussed here, only [16] used spatially disjoint data sets, with significant differences from our work as mentioned above. In addition, we are developing smoothing and thresholding methods to obtain regional predictions.

3. General approach

Initially, a classifier or ensemble of classifiers is constructed using the labeled, spatially disjoint, training data local to each partition. Each of these classifiers or ensembles is then transferred to a test partition of either the same or similar simulations. Once there, each classifier or fast ensemble of classifiers is used to predict the class of each instance of test data local to that compute node. Due to possible class imbalances, a probabilistic majority vote of all class predictions is taken to determine the consensus class of each instance of test data. Because regional predictions are the ultimate goal, connected-component regions of the predicted data are constructed, smoothed, and thresholded for better accuracy. For evaluation purposes, these predicted regions are compared to the labeled ground truth test regions, possibly using different overlap thresholds to determine the quality of each result.

4. Simulation dataset description

In the can-crush simulation, an impactor bar crushes a canister from above [20]. The wall of the canister buckles under the pressure and the top of the canister travels downward until it meets the bottom or the impactor bar stops. In our experiments, depending on the particular simulation, we observe 25 to 44 time steps for the simulated event.

4.1. Physical and spatial characteristics

In the four different instances of the can-crush simulation provided to us by the Department of Energy, all in the EXODUS II format [21], either six or nine physical variables were stored for each node within each of the time steps. They are the displacement on the X, Y, and Z axes; velocity on the X, Y, and Z axes; and in canister simulation 1 only: acceleration on the X, Y, and Z axes (as shown in Fig. 1). The nodes and finite elements of the simulation model are embedded in a mesh framework. Table 2 shows the parameter settings for each simulation. Table 3 shows the ranges taken on by the features available in each simulation.

Fig. 3 shows a visualization of ground truth data in the final time step of each simulation. Simulation 2 ends before much of the canister has been crushed. Simulation 4 extends past the point of the impactor bar itself being deformed.

The data for each of the time steps is divided spatially according to the compute node to which it is assigned. The vertical partitioning was performed along the Y-axis of the canister, dividing the canister into four disjoint spatial partitions of roughly equal size. The horizontal partitioning was performed along the Z-axis of the canister,

Please cite this article in press as: L. Shoemaker et al., Using classifier ensembles to label spatially disjoint data, Informat. Fusion (2007), doi:10.1016/j.inffus.2007.08.001

separating four disjoint spatial partitions from the impactor bar partition. Table 4 shows the percentage of salient nodes in each vertical and horizontal canister partition. Horizontal partitions, especially for simulation 2, have a larger class imbalance among partitions than vertical partitions. Data from the impactor bar is not used for training or testing in either the vertical or horizontal experiments. This represents the focus of the simulation designers on modeling the integrity of the canister.

Table 2
Physical and spatial characteristics for the canister simulations

Canister simulation	1	2	3	4
Bar initial velocity	5000	2500	5000	7500
No. of nodal variables	9	6	6	6
No. of canister nodes per time step	6724	6724	6724	6724
No. of bar nodes per time step	3364	1740	1740	1740
Total no. of nodes per time step	10,088	8464	8464	8464
No. of time steps	44	31	31	25
Total no. of canister nodes	295,856	208,444	208,444	168,100
% of salient canister nodes	64.7	27.3	51.3	60.7

Impactor bar velocity is in inches per second.

4.2. Train and test sets

To create labeled training data for every time step, those pieces of the canister that have buckled and been crushed are marked as salient by manual editing of the data via a custom plug-in to an open source visualization tool called ParaView [22]. At the beginning of the simulation, before the impactor bar has made contact, there are no salient nodes within the mesh. As time progresses and the canister collapses, more and more nodes are marked salient.

The marking of the salient nodes within the mesh can in principle be as precise as desired, but more precision requires greater effort in manual marking. In order to model a practical scenario where an expert is more interested in saving time than catering to the nuances of machine learning, we have allowed noise in the class labels by using tools that mark areas rather than individual points in the simulation model—there are 6724 canister points per time step. Almost 2.5% of all canister points change class as a horizontal face of the salient marking box is adjusted to include or exclude an entire horizontal layer of points. Smoothing of the output to create regions

Table 3
Feature ranges for canister data in simulations 1–4

Feature	Simulation 1		Simulation 2		Simulation 3		Simulation 4	
	min	max	min	max	min	max	min	max
DISPLX (in.)	−7.2	1.4	−4.0	0.5	−4.2	0.4	−4.5	1.0
DISPLY (in.)	−5.5	1.5	−1.2	1.5	−0.8	1.6	−1.6	5.9
DISPLZ (in.)	−17.8	0.1	−7.0	0.0	−13.2	0.0	−16.1	0.0
VELX (in./s)	−4820	2252	−4529	1161	−4562	2138	−30,840	14,385
VELY (in./s)	−7891	3357	−1327	2541	−2113	3616	−15,703	59,456
VELZ (in./s)	−8862	3287	−4837	493	−8226	998	−15,980	3986
ACCLX (in./s^2)	−1.75E+09	2.39E+09	NA	NA	NA	NA	NA	NA
ACCLY (in./s^2)	−2.47E+09	3.38E+09	NA	NA	NA	NA	NA	NA
ACCLZ (in./s^2)	−3.99E+09	3.02E+09	NA	NA	NA	NA	NA	NA

NA denotes not applicable.

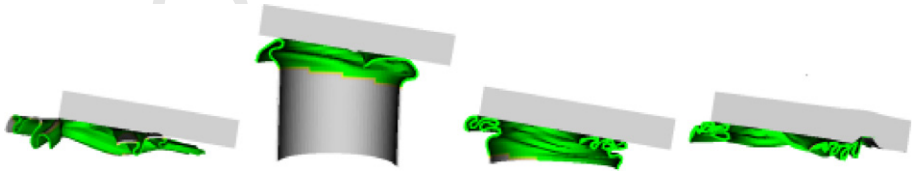

Fig. 3. Final time step in simulations 1–4 (left to right). Ground truth salient regions are darker than unknown regions.

Table 4
Salient class statistics by partition for the canister simulations

Simulation	% of salient nodes in each partition							
	Vertical partition				Horizontal partition			
	0	1	2	3	0	1	2	3
1	63.3	66.4	65.9	62.2	89.1	78.3	59.4	34.8
2	27.3	27.3	27.2	27.4	74.7	35.8	1.3	0.0
3	51.1	51.5	51.4	51.3	86.4	66.3	41.5	14.8
4	60.4	60.6	60.9	60.9	88.7	74.1	52.8	30.1

Please cite this article in press as: L. Shoemaker et al., Using classifier ensembles to label spatially disjoint data, Informat. Fusion (2007), doi:10.1016/j.inffus.2007.08.001

may reduce the noise in predictions created by imprecise labeling, as we shall see.

In each time step and in each partition, saliency is designated in the above fashion. Every node not designated salient receives the label "unknown", rather than "not salient", to reflect the fact that, in general, the users will indicate only salient regions. A classifier or an ensemble of classifiers is trained on each of the four partitions. Testing on each partition is performed using a probabilistic combination of the votes (to be reviewed in Section 5) from the three ensembles not trained on that partition. Additionally, the classifiers generated in one simulation are tested on the data from other similar simulations. Therefore each test example is classified by using classifiers trained on examples from other partitions in the simulation, or from classifiers from other simulations.

The classifiers predict each test example based on the attributes associated with that example. We obtain region-based results by smoothing and thresholding the point-based predictions. Because of the size of the data, our application focuses on a regional scale, where a few improperly labeled or predicted examples are not important.

5. Classification system

First, to establish a baseline for each partition we used a single default pruned C4.5 decision tree (DT) with certainty factor = 25 trained on the data at that partition. Then we used Breiman's random forest (RF) algorithm [23], with 250 unpruned trees per partition with both unweighted and weighted (RFW) predictions. Its accuracy was evaluated in [24] and shown to be comparable with or better than other well known ensemble generation techniques. The number of random features chosen at each decision tree node was $\log_2 n + 1$ given n features. Unweighted predictions produce a single class vote for the forest, while weighted predictions are based on the percentage of trees that vote for a class. The motivation for using this ensemble technique stems from the inherent speed benefit of analyzing only a few possible attributes from which a test is selected at an internal tree node.

For simulation 1 we also used k-nearest neighbors (KNN) and a variation of KNN which we call k-nearest centroids (KNC) [1]. The slower KNN classifier requires access to all of the training data at a compute node, and is included only to have a comparison point for KNC. The KNC classifier only requires one centroid for each training class be present at each time step at a compute node. The feature ranges of all training data were used to linearly normalize the training data before KNN and KNC classifiers were built. The test data was normalized linearly based on the training data feature ranges.

Classification of a test point within the simulation involves prediction by each partition's ensemble. Because our algorithms need to work when only a few compute nodes have salient examples, a simple majority vote algorithm may fail to classify any points as salient. In a large-scale simulation it is likely that there will be nodes which have no salient examples in training. If many individual classifiers are unable to predict salient because there are no salient examples in the individual training sets, then it may be impossible for a majority vote to predict salient. Therefore we must consider the prior probability that any given node contained salient examples during training and therefore is capable of producing a classifier that can predict an example as salient. A breakdown of this algorithm as presented in [25] is as follows:

$p(w_1|x) = \%$ of ensembles voting for class w_1 for example x

$P(w_1) = \%$ of ensembles capable of predicting class w_1

$$\text{Classify as } w_1 \text{ if : } \frac{p(w_1|x)}{P(w_1)} > \frac{p(w_2|x)}{P(w_2)}$$

$$\text{Classify as } w_2 \text{ if : } \frac{p(w_1|x)}{P(w_1)} < \frac{p(w_2|x)}{P(w_2)}$$

Thus, a probabilistic majority vote was applied for a two-class problem. An n-class problem as addressed in [25] is solved as follows:

$$\text{Classify as } w_n : \operatorname{argmax}_n \left(\frac{p(w_n|x)}{P(w_n)} \right)$$

In the case of a tie vote, the unknown class is predicted, since a definite salient vote has not been determined. We are interested in directing people to salient regions so, presumably, missing a few salient points that are tied in a vote will not be important for region recognition.

6. Can crush experiments

Two types of experiments were performed. In out-of-partition tests, we trained ensembles ensembles, each on data from 3 of 4 separate partitions in order to test the accuracy on the held out partition of that simulation. This was repeated until each partition was held out once. In cross simulation experiments, we trained four ensembles, each on the data within a separate partition of one simulation, in order to test the ensembles against the data in a different simulation. The cross simulation scenario should be more realistic than the out-of-partition scenario.

We will first report error rates per mesh node or point. Then we will examine how this translates into regional accuracy for the cross simulation experiments.

6.1. Out-of-partition (OOP) experiments

Training was performed on the data contained in each partition. The classifier or ensemble of the non-test partition returns a single prediction (or a weighted prediction in the case of random forests weighted) for each example in the separate test partition. Those three OOP predictions are combined into a single prediction for each example using the probabilistic majority vote. Predictions on the test examples are compared to the marked saliency of the test

examples to determine error rates. This process is repeated until each partition in the simulation has been tested.

6.2. Cross simulation experiments

Training is performed on the data contained in each of the four partitions of simulations 1–4 to create both a single pruned decision tree and a 250-tree random forest ensemble for each partition. The decision tree classifier or the random forest ensemble of each partition in a training simulation returns a single prediction (or a weighted prediction in the case of random forests weighted) for each test example of a separate test simulation. The four predictions from those classifiers or ensembles are combined into a single prediction for each example in the separate test simulation using the probabilistic majority vote. Predictions on the test examples are compared to the marked saliency of the test examples to determine error rates. We also process the nodal ground truth data and nodal predicted data in order to determine regional accuracy.

7. Can crush results

Results are separated into out-of-partition results for experiments within each simulation, and cross simulation results for experiments using one simulation for training data and a different simulation for test data. Results on the accuracy in detecting salient regions are also reported for cross simulation experiments.

7.1. Out-of-partition results

Table 5 shows the vertical out-of-partition error rates for each test partition of simulation 1, using the probabilistic majority vote. This result is further broken down by class. The FP (false positives) entries measure the classification error rate of examples in the unknown class. The FN (false negatives) entries do the same for examples which were ground truth labeled as the salient class. The entries labeled "All" measure the classification error rate of all examples in the test partition. The total error rate is obtained by averaging the error rate of each partition's classifier according to the number of nodes it has classified. This rate is also broken down by class. Random forest ensembles result in the lowest overall error rates. The KNC classifier results in higher error rates on salient examples than the much slower KNN classifier, and both exceed the overall error rates of random forest ensembles.

Table 6 shows the horizontal out-of-partition error rates for each test partition of simulation 1, using the probabilistic majority vote. Error rates were computed as described above. A single pruned decision tree classifier built on each partition narrowly edges the random forest ensembles for lowest overall error rate. The error rates are generally higher for horizontal partition experiments than those for vertical partitions. In horizontal partitioning experiments, most of the nodes in the partition (0) closest to the impactor bar at time step 0 have salient ground truth after the first five time steps. In contrast, most of the nodes in the partition (3) farthest from the impactor bar at time step 0 do not have salient ground truth until more than 24 time steps have elapsed. This leads to 89.1% of partition 0 nodes with salient examples and only 34.8% of partition three nodes with salient examples. Vertical partitions have much more uniform saliency percentages.

Table 7 shows the vertical and horizontal test partition error rates for each test partition of simulations 2, 3, and 4, using the probabilistic majority vote. Error rates were computed as described previously. Horizontal partition 3 (farthest from impactor bar at the first time step) of simulation 2 does not have any example whose ground truth is salient. Only 27% of canister nodes in simulation 2 were marked salient. This resulted in the salient (FN) error rates in horizontal simulation 2 being more than 20% except for random forests weighted (14.4%). If a simple majority vote had been used instead of the probabilistic majority vote, the only out-of-partition results that would change are those using random forest (weighted) for horizontal partitions in simulation 2. In this case, a simple majority vote would be about 1% more accurate overall, but would be 19% less accurate for salient examples.

Only 39.3% of canister nodes in simulation 4 were labeled unknown, and the unknown (FP) error rates in horizontal simulation 4 are higher than the salient (FN) error rates. Simulation 3 has the closest balance between nodes marked unknown and salient (48.5% and 51.5%) and has low unknown and low salient error rates for the horizontal simulation. The canister in simulation 2 is not fully crushed, while the canister in simulation 4 is crushed past the point of impactor bar deformation.

Table 5
Out-of-partition error rates for four vertical partitions of the canister in simulation 1

Classfier/Ensemble	Test partition error rates (%)												Total error rates (%)		
	Part 0			Part 1			Part 2			Part 3					
	FP	FN	All	FP	FN	All	FP	FN	All	FP	FN	All	FP	FN	All
DT	44	1	17	8	2	4	7	1	3	18	1	7	18.7	1.4	7.5
RF	36	1	13	6	1	3	6	1	3	17	1	7	15.8	0.9	6.2
RFW	31	1	12	4	1	2	5	1	2	16	1	7	13.7	1.0	5.5
KNC	5	22	15	0	21	14	2	16	11	12	16	15	4.3	18.9	13.7
KNN	30	2	12	8	2	4	8	1	4	21	1	9	16.4	1.7	6.9

FP denotes false positives. FN denotes false negatives. All is the total error percentage.

Please cite this article in press as: L. Shoemaker et al., Using classifier ensembles to label spatially disjoint data, Informat. Fusion (2007), doi:10.1016/j.inffus.2007.08.001

Table 6
Out-of-partition error rates for four horizontal partitions of the canister in simulation 1

Classfier/Ensemble	Test partition error rates (%)												Total error rates (%)		
	Part 0			Part 1			Part 2			Part 3					
	FP	FN	All	FP	FN	All	FP	FN	All	FP	FN	All	FP	FN	All
DT	27	4	7	12	2	4	11	4	7	23	6	17	18.5	4.1	8.9
RF	37	2	5	11	1	4	23	12	10	36	1	24	26.8	3.8	10.9
RFW	36	2	5	12	1	3	20	0	8	34	1	22	25.5	0.9	10.0
KNC	33	5	8	4	9	8	2	19	12	4	44	18	5.3	14.9	11.5
KNN	36	2	6	9	3	4	24	1	10	46	0	30	33.5	1.7	12.9

FP denotes false positives. FN denotes false negatives. All is the total error percentage.

Table 7
Out-of-partition error rates for canister simulations 2, 3, and 4 with the lowest overall error rates in bold. FP denotes false positives

Classifier/Ensemble	Simulation	Error rates(%)					
		Vertical partitions			Horizontal partitions		
		FP	FN	All	FP	FN	All
DT	2	4.4	6.5	5.0	6.3	37.4	14.8
RF	2	4.6	4.9	4.7	4.0	20.4	**8.5**
RFW	2	3.7	4.8	**4.0**	11.3	14.4	12.2
DT	3	3.5	2.9	3.2	12.4	1.4	6.8
RF	3	3.9	2.4	3.1	11.7	1.0	**6.2**
RFW	3	3.2	2.4	**2.8**	12.4	0.8	6.5
DT	4	4.2	1.2	2.4	23.9	0.9	9.9
RF	4	3.4	1.1	2.0	17.2	0.5	**7.1**
RFW	4	2.6	1.1	**1.7**	13.0	0.6	5.4

FN denotes false negatives. All is the total error percentage.

Interestingly, random forests weighted is slightly more accurate overall four out of six times for the results shown in Table 7. In addition, random forests weighted provides a lower maximum error rate for the two classes five out of six times.

7.2. Cross simulation nodal results

Tables 8 and 9 show the error rates for cross simulation experiments. Since the final vote is computed using one vote from each of the four partitions, ties may exist in this two-class experiment. Ties are assigned to the unknown class. The number of ties is such that assigning them instead to the salient class typically causes less than a 5% swing in the overall error rate, with a maximum outlier of about 10%.

If a simple majority vote had been used instead of the probabilistic majority vote, the only cross simulation results that would change are those using horizontal partitions in simulation 2 for training. In this case, a simple majority vote would be less accurate overall in 6 of the 9 cases, and would be less accurate for salient examples in every case.

A version of random forests typically results in the lowest overall error rate in the cross simulation experiments. However, the decision tree had a lower overall error rate for all three horizontal experiments that train on simulation 4. The random forests were most accurate in the out-of-partition experiments for simulation four, as shown in Table 7. So, it might not be advantageous to take the maximum accuracy model from an out-of-partition experiment and use it for cross simulation prediction.

7.3. Cross simulation regional results

The goal of the prediction stage is to direct experts to additional salient regions. Assessing the accuracy of an algorithm in finding and classifying regions is more difficult than determining the above node-level results. We compute a quantitative measure of region detection accuracy.

The salient regions of the data were marked using region-based tools of the ParaView application [22]. The ensembles of classifiers used to classify the test data often produce smaller salient clusters of nodes or even individual isolated salient nodes, which do not correspond well to the larger marked, ground truth regions. In order to improve the regional accuracy of these ensembles, we employed some of the regional tools in the Feature Characterization Library (FCLib-1.2.0) toolkit [26] to process the ensemble prediction data. The numerical class label (0.5 for unknown, 1.0 for salient) of all nodes within a physical radius of 2 inches of each node was averaged in a smoothing operation. After smoothing, nodes had numerical class labels in the range from $[0.5, 1]$. The midpoint of this range, 0.75, was chosen as the threshold used to label the nodes as salient. Predicted regions were created from connected components of salient nodes before and after smoothing. Smoothing tended to remove the smaller salient regions

Table 8
Cross simulation nodal error rates (part 1) for canister simulations 1, 2, 3, and 4 with the lowest overall error results in bold

Classifier/Ensemble	Simulation Train/Test	Error rates (%)					
		Vertical partitions			Horizontal partitions		
		FP	FN	All	FP	FN	All
DT	1/2	2.3	14.3	5.5	3.3	13.2	6.0
RF	1/2	2.1	10.3	4.3	2.5	11.1	**4.8**
RFW	1/2	2.6	8.6	**4.2**	4.8	8.4	5.8
DT	1/3	5.2	4.1	4.7	11.1	7.1	9.0
RF	1/3	4.1	2.8	**3.4**	5.3	2.8	**4.0**
RFW	1/3	5.3	1.9	3.6	10.4	1.5	5.8
DT	1/4	5.5	4.5	4.9	8.2	11.6	10.2
RF	1/4	7.2	2.4	**4.3**	5.7	2.3	**3.6**
RFW	1/4	8.8	1.4	**4.3**	10.8	1.2	5.0
DT	2/1	24.9	4.2	**11.5**	39.7	3.0	16.0
RF	2/1	31.3	1.1	11.8	41.6	0.8	15.2
RFW	2/1	40.5	0.6	14.7	32.3	1.4	**12.3**
DT	2/3	10.2	5.4	7.7	18.9	5.1	11.8
RF	2/3	11.9	2.0	**6.8**	12.8	1.0	6.7
RFW	2/3	13.5	1.2	7.2	11.9	1.4	**6.5**
DT	2/4	14.9	6.7	9.9	17.1	5.4	10.0
RF	2/4	13.0	1.6	**6.1**	10.9	0.8	4.8
RFW	2/4	16.9	0.9	7.2	7.8	1.2	**3.8**

FP denotes false positives. FN denotes false negatives. All is the total error percentage.

Table 9
Cross simulation nodal error rates (part 2) for canister simulations 1, 2, 3, and 4 with the lowest overall error results in bold

Classifier/Ensemble	Simulation Train/Test	Error rates (%)					
		Vertical partitions			Horizontal partitions		
		FP	FN	All	FP	FN	All
DT	3/1	15.9	3.0	**7.5**	16.3	2.6	**7.4**
RF	3/1	20.4	1.5	8.2	17.9	2.0	7.6
RFW	3/1	24.8	1.0	9.4	27.4	0.9	10.2
DT	3/2	2.4	9.7	4.4	2.7	9.1	4.4
RF	3/2	2.2	9.2	**4.1**	1.9	9.0	**3.8**
RFW	3/2	2.7	7.9	**4.1**	2.8	6.3	**3.8**
DT	3/4	9.0	2.0	5.4	5.6	2.1	3.5
RF	3/4	8.2	1.8	**5.0**	5.8	1.4	**3.1**
RFW	3/4	10.1	1.4	5.7	8.5	0.9	3.9
DT	4/1	3.3	13.2	6.0	18.4	3.0	**8.4**
RF	4/1	2.5	11.1	**4.8**	24.7	1.6	9.8
RFW	4/1	4.8	8.4	5.8	27.2	1.2	10.4
DT	4/2	11.1	7.1	9.0	19.9	8.6	**16.8**
RF	4/2	5.3	2.8	**4.0**	23.5	6.7	18.9
RFW	4/2	10.4	1.5	5.8	22.0	6.0	17.6
DT	4/3	8.2	11.6	10.2	18.4	1.5	**9.7**
RF	4/3	5.7	2.3	**3.6**	21.2	0.9	10.8
RFW	4/3	10.8	1.2	5.0	21.4	0.7	10.8

FP denotes false positives. FN denotes false negatives. All is the total error percentage.

and the isolated salient nodes. Ground truth regions were also created without smoothing for comparison purposes. All pairs of salient regions separated by no more than the maximum edge distance between nodes in the simulation were assigned the same region label. Another tool was used to generate overlap matrices of connected component ground truth and predicted regions.

A previous approach in [27] did not consider the actual node intersection percentage of ground truth and predicted salient regions. We extend that approach by establishing 0.1%, 10%, and 50% thresholds for the overlap percentage of the nodes in a ground truth salient region and a predicted salient region for the prediction to be counted as correct or true positive. The overlap required for a true positive at given threshold is applied separately to the ground truth region and to the predicted region. If no predicted salient regions sufficiently overlap a ground truth salient region, a false negative is registered for the failure to adequately predict the ground truth region. The salient (FN) regional error rate is calculated as the percentage of

Please cite this article in press as: L. Shoemaker et al., Using classifier ensembles to label spatially disjoint data, Informat. Fusion (2007), doi:10.1016/j.inffus.2007.08.001

FN instances compared to the total number of ground truth salient regions.

A false positive is recorded for each predicted region that does not sufficiently overlap any ground truth region. This may result in more total predicted regions than actual regions. It is possible that more than one predicted salient region will satisfy a given overlap threshold for intersection with a labeled salient region. We count this as a single discovery of the ground truth region (true positive). For the purposes of people searching for interesting events, this appears sensible because they would be directed to the region.

Only the first time step has no salient regions. Hence, for that time step one could say that there is a true negative, if no regions are predicted salient. For all other time steps there is one spatially contiguous unknown region, that contains salient "island" region(s). Since we only predict salient regions, evaluating whether a potential true negative satisfies a specified overlap threshold between a ground truth unknown region and a "predicted" unknown region is not logical. Therefore, we show the number of false positives (FP) as an absolute number, rather than as a false positive rate.

An overall regional error rate that corresponds to previous overall nodal error rates might be misleading because of the true negative requirement mentioned above. However, the *F*-measure provides an overall measure of regional accuracy without the need for the number of true negatives, as shown below [28].

$$F\text{-measure} = \frac{2 \cdot \mathrm{TP}}{2 \cdot \mathrm{TP} + \mathrm{FP} + \mathrm{FN}}$$

We use the traditional *F*-measure or *F*-score, which weighs false positives and false negatives equally. The regional results are shown in Tables 10–13.

For many users, the 0.1% overlap threshold is an appropriate regional metric, since coarsely pointing those users to suspicious regions for further investigation is the main goal. From a machine learning viewpoint, the 0.1% overlap does not address the case where a very large region is always predicted salient. As long as this region minimally overlaps a given ground truth region, a true positive is counted. By increasing the overlap requirement to 10% or 50% for example, a more precise match can be used. The stricter requirements also provide useful discrimination between classifier methods that would not be possible with a minimal overlap requirement.

Salient regions are always detected without smoothing for overlap thresholds of 0.1% and 10%. Smaller regions that are predicted salient are either removed or consolidated into larger regions by smoothing. Fig. 4 shows an example of smoothing applied to a single time step of simulation 1 as predicted by the ensembles of random forests unweighted that were trained on data of simulation 4 horizontal partitions. The leftmost image shows ground truth. The middle image shows the false positive regions in this time step without smoothing. The rightmost image is after smoothing with a radius of 2 in. and contains no false positive regions.

For an overlap requirement of 0.1%, smoothing generally improves the *F*-measure of regional accuracy by removing small predicted regions that would be counted as false positive regions (predicted regions not connected to ground truth) as the radius is increased. There are a few exceptions where smoothing decreases the regional *F*-measure by removing small, correctly predicted regions. The *F*-measure in those cases decreases by removing a true positive predicted region, and possibly by adding a false negative (if no other predicted region overlaps the corresponding ground truth region).

Table 10
Cross simulation regional error rates for canister simulations 1, 2, 3, and 4 using four vertical partitions for training (part 1)

Classifier/Ensemble	Simulation Train/Test	Unsmoothed overlap = 0.1%			Smoothed overlap = 0.1%			Smoothed overlap = 10%			Smoothed overlap = 50%		
		FP	FN%	*F*-m	FP	FN%	*F*-m	FP	FN%	*F*-m	FP	FN%	*F*-m
DT	1/2	1	0	0.98	0	3	0.97	0	3	0.97	1	7	0.92
RF	1/2	0	0	1.00	0	3	0.97	0	3	0.97	0	3	0.97
RFW	1/2	0	0	1.00	0	3	0.97	0	3	0.97	0	3	0.97
DT	1/3	8	0	0.88	1	0	0.98	2	3	0.94	2	3	0.94
RF	1/3	3	0	0.95	0	0	1.00	0	0	1.00	1	3	0.95
RFW	1/3	2	0	0.97	0	0	1.00	0	0	1.00	1	3	0.95
DT	1/4	2	0	0.96	0	0	1.00	0	0	1.00	1	4	0.94
RF	1/4	1	0	0.98	0	0	1.00	0	0	1.00	0	0	1.00
RFW	1/4	0	0	1.00	0	0	1.00	0	0	1.00	0	0	1.00
DT	2/1	23	0	0.79	0	2	0.98	1	5	0.94	1	5	0.94
RF	2/1	17	0	0.83	0	2	0.98	0	2	0.98	1	5	0.94
RFW	2/1	13	0	0.87	0	2	0.98	0	2	0.98	0	2	0.98
DT	2/3	13	0	0.82	4	3	0.91	4	3	0.91	5	7	0.86
RF	2/3	9	0	0.87	1	3	0.95	1	3	0.95	1	3	0.95
RFW	2/3	6	0	0.91	2	3	0.94	2	3	0.94	2	3	0.94
DT	2/4	7	0	0.87	1	0	0.98	2	4	0.92	2	4	0.92
RF	2/4	7	0	0.87	1	0	0.98	2	4	0.92	2	4	0.92
RFW	2/4	4	0	0.92	0	0	1.00	0	0	1.00	1	4	0.94

FP denotes false positives. FN denotes false negatives. *F*-m denotes *F*-measure.

Table 11
Cross simulation regional error rates for canister simulations 1, 2, 3, and 4 using four vertical partitions for training (part 2)

Classifier/Ensemble	Simulation Train/Test	Unsmoothed overlap = 0.1%			Smoothed overlap = 0.1%			Smoothed overlap = 10%			Smoothed overlap = 50%		
		FP	FN%	*F*-m	FP	FN%	*F*-m	FP	FN%	*F*-m	FP	FN%	*F*-m
DT	3/1	7	0	0.93	0	0	1.00	1	2	0.97	1	2	0.97
RF	3/1	5	0	0.95	0	0	1.00	1	2	0.97	1	2	0.97
RFW	3/1	3	0	0.97	0	0	1.00	0	0	1.00	1	2	0.97
DT	3/2	4	0	0.94	2	3	0.94	2	3	0.94	3	7	0.89
RF	3/2	0	0	1.00	1	3	0.95	1	3	0.95	2	7	0.90
RFW	3/2	0	0	1.00	1	3	0.95	1	3	0.95	2	7	0.90
DT	3/4	1	0	0.98	0	0	1.00	0	0	1.00	0	0	1.00
RF	3/4	0	0	1.00	0	0	1.00	0	0	1.00	0	0	1.00
RFW	3/4	0	0	1.00	0	0	1.00	0	0	1.00	0	0	1.00
DT	4/1	30	0	0.75	0	0	1.00	0	0	1.00	1	2	0.97
RF	4/1	7	0	0.93	0	0	1.00	0	0	1.00	1	2	0.97
RFW	4/1	6	0	0.94	0	0	1.00	0	0	1.00	1	2	0.97
DT	4/2	2	0	0.97	0	3	0.97	0	3	0.97	0	3	0.97
RF	4/2	2	0	0.97	1	3	0.95	1	3	0.95	1	3	0.95
RFW	4/2	1	0	0.98	0	3	0.97	0	3	0.97	0	3	0.97
DT	4/3	2	0	0.97	1	0	0.98	2	3	0.94	2	3	0.94
RF	4/3	1	0	0.98	1	0	0.98	2	3	0.94	2	3	0.94
RFW	4/3	1	0	0.98	2	0	0.97	2	0	0.97	3	3	0.92

FP denotes false positives. FN denotes false negatives. *F*-m denotes *F*-measure.

Table 12
Cross simulation regional results for canister simulations 1, 2, 3, and 4 using four horizontal partitions for training (part 1)

Classifier/Ensemble	Simulation Train/Test	Unsmoothed overlap = 0.1%			Smoothed overlap = 0.1%			Smoothed overlap = 10%			Smoothed overlap = 50%		
		FP	FN%	*F*-m	FP	FN%	*F*-m	FP	FN%	*F*-m	FP	FN%	*F*-m
DT	1/2	2	0	0.97	0	3	0.97	0	3	0.97	2	10	0.87
RF	1/2	0	0	1.00	0	3	0.97	1	7	0.92	2	10	0.87
RFW	1/2	2	0	0.97	0	3	0.97	0	3	0.97	1	7	0.92
DT	1/3	5	0	0.92	0	3	0.97	0	3	0.97	0	3	0.97
RF	1/3	1	0	0.98	0	3	0.97	0	3	0.97	0	3	0.97
RFW	1/3	6	0	0.91	1	3	0.95	1	3	0.95	1	3	0.95
DT	1/4	4	0	0.92	2	4	0.92	2	4	0.92	2	4	0.92
RF	1/4	0	0	1.00	1	4	0.94	1	4	0.94	1	4	0.94
RFW	1/4	0	0	1.00	0	0	1.00	0	0	1.00	1	4	0.94
DT	2/1	16	0	0.85	3	2	0.95	3	2	0.94	7	9	0.84
RF	2/1	15	0	0.85	0	2	0.98	0	2	0.98	0	2	0.98
RFW	2/1	10	0	0.90	0	2	0.98	0	2	0.98	0	2	0.98
DT	2/3	23	0	0.72	1	0	0.98	2	3	0.94	8	17	0.74
RF	2/3	6	0	0.91	1	3	0.95	1	3	0.95	1	3	0.95
RFW	2/3	5	0	0.92	0	3	0.97	0	3	0.97	0	3	0.97
DT	2/4	24	0	0.67	4	4	0.88	4	4	0.88	5	8	0.83
RF	2/4	6	0	0.89	1	0	0.98	1	0	0.98	2	4	0.92
RFW	2/4	13	0	0.79	0	4	0.96	0	4	0.96	0	4	0.96

FP denotes false positives. FN denotes false negatives. *F*-m denotes *F*-measure.

Regional results in Table 13 for simulation 2 that use horizontal partitions of simulation 4 for training, show a substantial decrease in *F*-measure when the overlap threshold is increased from 10% to 50%. Simulations 2 and 4 have the greatest difference in the initial impactor bar velocity and a substantial difference in the bar final position, which makes this decrease less surprising. While random forests more often have a higher unsmoothed regional *F*-measure than decision tree ensembles, smoothing reduces the random forests advantage. In general, with a smoothing radius of 2, over 98% of the salient regions are correctly identified with random forests ensembles for an overlap threshold of 10% or less.

8. Face image data

In order to determine how transferable this approach may be to a different domain, we revisited our previous work [1,25]. The classification task is to identify different regions of face images, an "Interesting" region containing eyes and mouth, and a "Somewhat Interesting" region containing eyebrows. Face images obtained from the FERET database [29,30] and preprocessed [31,32] were partitioned, in a four row by two column arrangement, vertically, and horizontally. An example is shown in Fig. 5.

Five training images, each of a different person, were used (only one is shown). Six simple features were gener-

Table 13
Cross simulation regional results for canister simulations 1, 2, 3, and 4 using four horizontal partitions for training (part 2)

Classifier/Ensemble	Simulation Train/Test	Unsmoothed overlap = 0.1%			Smoothed overlap = 0.1%			Smoothed overlap = 10%			Smoothed overlap = 50%		
		FP	FN%	*F*-m	FP	FN%	*F*-m	FP	FN%	*F*-m	FP	FN%	*F*-m
DT	3/1	24	0	0.79	0	0	1.00	0	0	1.00	1	2	0.97
RF	3/1	7	0	0.93	0	0	1.00	1	2	0.97	1	2	0.97
RFW	3/1	3	0	0.97	0	0	1.00	1	2	0.97	1	2	0.97
DT	3/2	9	0	0.87	1	3	0.95	1	3	0.95	2	7	0.90
RF	3/2	3	0	0.95	0	3	0.97	0	3	0.97	1	7	0.92
RFW	3/2	2	0	0.97	1	3	0.95	1	3	0.95	1	3	0.95
DT	3/4	6	0	0.89	1	0	0.98	1	0	0.98	2	4	0.92
RF	3/4	1	0	0.98	0	0	1.00	0	0	1.00	1	4	0.94
RFW	3/4	1	0	0.98	0	0	1.00	0	0	1.00	1	4	0.94
DT	4/1	18	0	0.83	1	0	0.99	1	0	0.99	2	2	0.95
RF	4/1	8	0	0.92	0	0	1.00	0	0	1.00	1	2	0.97
RFW	4/1	19	0	0.82	0	0	1.00	0	0	1.00	1	2	0.97
DT	4/2	19	0	0.76	4	3	0.91	4	3	0.91	10	23	0.66
RF	4/2	5	0	0.92	0	3	0.97	0	3	0.97	8	30	0.62
RFW	4/2	2	0	0.97	1	3	0.95	1	3	0.95	9	30	0.61
DT	4/3	10	0	0.86	1	0	0.98	1	0	0.98	2	3	0.94
RF	4/3	0	0	1.00	0	0	1.00	0	0	1.00	1	3	0.95
RFW	4/3	2	0	0.97	0	0	1.00	0	0	1.00	1	3	0.95

FP denotes false positives. FN denotes false negatives. *F*-m denotes *F*-measure.

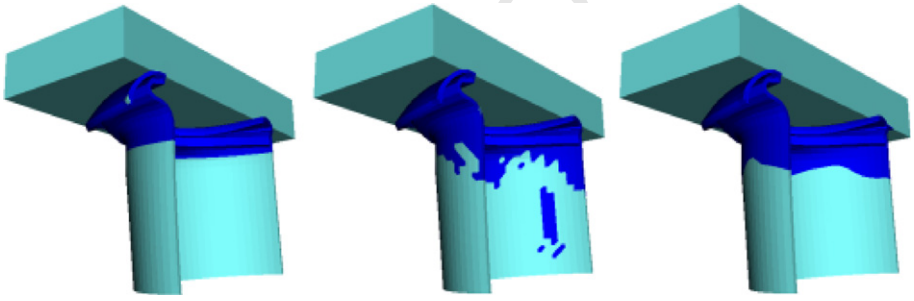

Fig. 4. Left: Ground truth as labeled in time step 15 of Simulation 1. Center: Predicted salient regions including false positives (smaller regions) before smoothing. Right: Predicted salient regions after smoothing with no false positives.

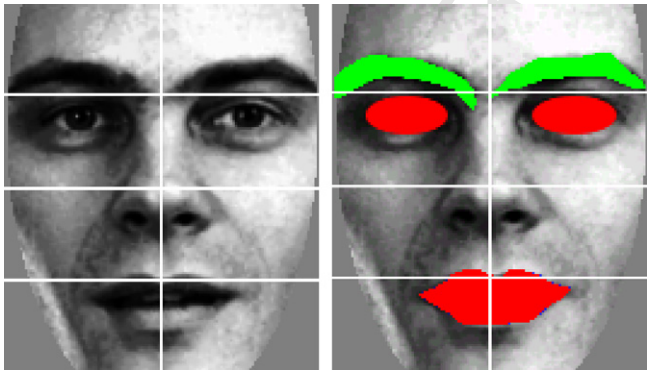

Fig. 5. Training image from the FERET database showing marked saliency for both "Interesting" and "Somewhat Interesting" classes for eight partitions delineated by white lines. The "Interesting" class contains the eyes and mouth. The "Somewhat Interesting" class contains only the eyebrows.

ated for each pixel, including its intensity, and for a 5×5 neighborhood of the pixel, the maximum, minimum, range, arithmetic mean, and standard deviation of the pixel intensities in the neighborhood. Using 40 KNC classifiers, each trained on 1 of the 8 partitions of the five training images, we were able to identify salient regions. However, other regions were also labeled. We compare those results with results using 40 random forests each with 1000 trees. Each forest was created by also training on 1 of the 8 partitions of a training image in combination with probabilistic majority voting using priors. It was tested on a separate test image taken under similar lighting conditions with a similar expression.

Region detection was performed on these images by first averaging the predicted saliency values with a 5×5 window. The resulting floating point pixel values which were greater than an automatically determined threshold were marked as salient. This threshold was determined by an image binarization algorithm by Otsu which chooses a threshold that minimizes the interclass variance [33]. We then observed the connected components and designated each one as a region.

A comparison of the *k*-nearest centroid algorithm using 11 centroids to 8 random forests of 1000 decision trees is

shown in Fig. 6. Neither provides for significant differentiation between the "interesting" and "somewhat interesting" classes. This is likely due to the weakness of the derived features. Random forest ensembles produce fewer false positives and likewise a more meaningful list of regions.

Figs. 7 and 8 show the test image results from KNC and random forests built on the same training image that was partitioned into vertical and horizontal partitions, and processed as before. In each case random forest ensembles produce fewer false positives. The high number of false positives using KNC may be detrimental for directing an examiner to salient regions. This is readily observed when KNC points users to large uninteresting regions while RF regions are much smaller.

We note that in this comparison, both of the algorithms were able to find all of the salient regions: eyes, eyebrows, and mouth. Because a low number of false positives is crucial to the users' perceived confidence in the program, we critically examine those regions which should not have been labeled salient. The KNC algorithm creates very large regions which include many uninteresting points, mostly around the nose and near the side of the face. This undesirable behavior is not easily corrected by simply invalidating the region. Instead, the user must manually correct the labeling using the available tools. In the RF algorithm, the nose and nostril areas are their own individual regions, which is easy to correct for. In no case did the RF algorithm mark the side of the face as salient, an issue which occurred in each of the KNC images.

In this experiment, there were different levels of interest for the regions and we were not able to differentiate among those. Shadows around the nose and chin were often dark and misinterpreted as eyebrows or eyes.

9. Summary and discussion

Large simulations must be partitioned across multiple processors in order to obtain results in a reasonable

Fig. 6. Left to Right: Test image. Test image showing marked saliency. Saliency predictions using KNC with 11 centroids. Saliency predictions using 1000 random forest trees per partition.

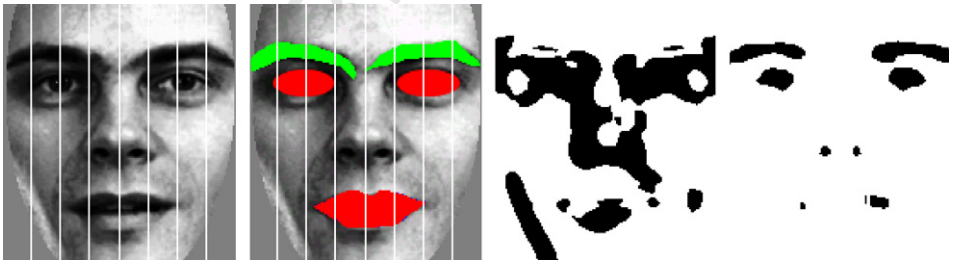

Fig. 7. Left to Right: One of five training images. Test image showing marked saliency. Saliency predictions using KNC with 11 centroids. Saliency predictions using 1000 random forest trees per partition.

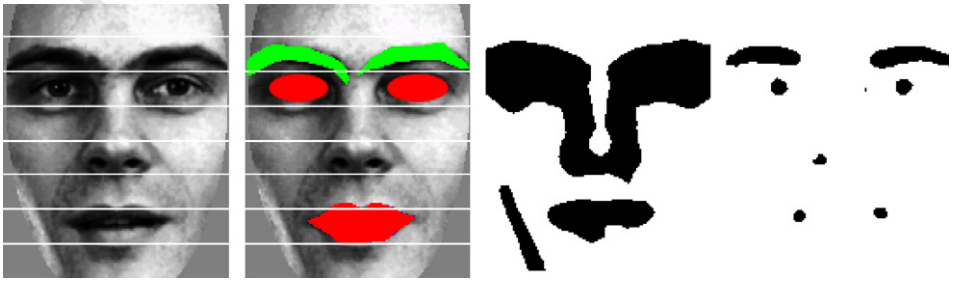

Fig. 8. Left to Right: One of five training images. Test image showing marked saliency. Saliency predictions using KNC with 11 centroids. Saliency predictions using 1000 random forest trees per partition.

Please cite this article in press as: L. Shoemaker et al., Using classifier ensembles to label spatially disjoint data, Informat. Fusion (2007), doi:10.1016/j.inffus.2007.08.001

amount of time. The method of breaking data into pieces may cause highly skewed class distributions, as it violates the usual assumption of independent and identically distributed data sets. In this paper, we show how such data may be nonetheless effectively used for data mining. Our approach uses fast ensemble learning algorithms and probabilistic majority voting.

Results on several canister crush simulations indicate that our approach has the ability to find most nodes in regions of interest. In our experiments using the data from several different simulation runs, the resultant predictions appear more accurate (in terms of matching the physical processes in the simulation) than the training data, which has been labeled approximately in accordance with the time constraints placed upon experts. This provides confidence that the algorithm is learning the underlying function that determines which points are salient, with the overlap of uninteresting points outweighing the very large number of uninteresting points overall.

For face images, we used very simple features and made no attempt to do any optimization and were still able to successfully find the regions of interest. In this experiment, there were different levels of interest for the regions and we were not able to differentiate among those. Shadows around the nose and chin were often dark and misinterpreted as eyebrows or eyes. However, these false positives were at least partly a function of the very simple features used. So, success in finding regions was high with a few false positives.

We evaluated how well regions of salience are found in canister crush cross simulation experiments. After smoothing the results of random forests weighted prediction, there were at most one false negative and/or two false positive regions per test simulation for an overlap threshold of 10% or less. Overall 98% of the salient regions are correctly identified in those cases. So, this is a promising result in terms of the utility of the approach. The results indicate that simulation developers and users would be accurately directed to regions of interest with only occasional misdirection. This has the potential for saving significant time during debugging and use by allowing for a much improved focus of attention on areas of interest without highly time-consuming search.

We believe the rapid generation of ensemble classifiers will make it tractable to predict saliency in much larger data sets. The general problem of creating an ensemble from data that was partitioned without regard to the effect on the machine learning algorithm is an important practical problem that merits additional attention.

Acknowledgements

This research was partially supported by the Department of Energy through the Advanced Simulation and Computing program (ASC) Visual Interactive Environment for Weapons Simulation (VIEWS) Data Discovery Program Contract number: DE-AC04-76DO00789.

References

[1] L. Hall, D. Bhadoria, K. Bowyer, Learning a model from spatially disjoint data, in: 2004 IEEE International Conference on Systems, Man, and Cybernetics, vol. 2, 2004, pp. 1447–1451
[2] D.F. Kusnezov, Advanced Simulation & Computing: the Next Ten years, Technical Report, Sandia National Labs, Albuquerque, NM 87185, 2004.
[3] ASC, National Nuclear Security Administration in collaboration with Sandia, Lawrence Livermore, and Los Alamos National Laboratories, http://www.sandia.gov/nnsa/asc/ (accessed 31.12.2006).
[4] C.C. Aggarwal, J. Han, J. Wang, P.S. Yu, On demand classification of data streams, in: KDD'04: Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM Press, New York, NY, USA, 2004, pp. 503–508.
[5] P. Domingos, G. Hulten, Mining high-speed data streams, in: KDD'00: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM Press, New York, NY, USA, 2000, pp. 71–80.
[6] N. Chawla, L. Hall, K. Bowyer, W. Kegelmeyer, Learning ensembles from bites: a scalable and accurate approach, Journal of Machine Learning Research 5 (2004) 421–451.
[7] S. Kotsiantis, D. Kanellopoulos, P. Pintelas, Handling imbalanced datasets: a review, GESTS International Transactions on Computer Science and Engineering 30 (1) (2006) 25–36.
[8] G. Weiss, Mining with rarity: a unifying framework, SIGKDD Explorations 6 (1) (2004) 7–19.
[9] N. Chawla, L. Hall, K. Bowyer, W. Kegelmeyer, SMOTE: synthetic minority over-sampling technique, Journal of Artificial Intelligence Research 16 (2002) 321–357.
[10] P. Domingos, Metacost: a general method for making classifiers cost-sensitive, in: KDD'99: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM Press, New York, NY, USA, 1999, pp. 155–164.
[11] Z. Erdem, R. Polikar, F. Gurgen, N. Yumusak, Ensemble of SVMs for incremental learning, in: Multiple Classifier Systems, 6th International Workshop, vol. 3541 of Lecture Notes in Computer Science, Springer, Seaside, CA, USA, 2005, pp. 246–256.
[12] M. Maloof, R. Michalski, Incremental learning with partial instance memory, Artificial Intelligence 154 (1–2) (2004) 95–126.
[13] G. Webb, J. Boughton, Z. Wang, Not so naive Bayes: aggregating one-dependence estimators, Machine Learning 58 (1) (2005) 5–24.
[14] R. Kong, B. Zhang, A fast incremental learning algorithm for support vector machine, Control and Decision 20 (10) (2005) 1129–1136.
[15] W. Fan, Systematic data selection to mine concept-drifting data streams, in: KDD'04: Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM Press, New York, NY, USA, 2004, pp. 128–137.
[16] A. Lazarevic, Z. Obradovic, Boosting algorithms for parallel and distributed learning, Distributed and Parallel Databases Journal 11 (2) (2002) 203–229.
[17] N. Chawla, T. Moore, L. Hall, K. Bowyer, W. Kegelmeyer, C. Springer, Distributed learning with bagging-like performance, Pattern Recognition Letters 24 (1–3) (2003) 455–471.
[18] W. Fan, H. Wang, P. Yu, S. Stolfo, A fully distributed framework for cost-sensitive data mining, in: Proceedings 22nd International Conference on Distributed Computing Systems, 2002, pp. 445–446.
[19] C. Shipp, L. Kuncheva, Relationships between combination methods and measures of diversity in combining classifiers, Information Fusion 3 (2) (2002) 135–148.
[20] B.S. Lee, R.R. Snapp, R. Musick, Toward a query language on simulation mesh data: an object-oriented approach, in: DASFAA'01: Proceedings of the 7th International Conference on Database Systems for Advanced Applications, IEEE Computer Society, Washington, DC, USA, 2001, pp. 242–249.

[21] L.A. Schoof, V.R. Yarberry, EXODUS II: a finite element data model, Tech. rep., Sandia National Labs, Albuquerque, NM 87185, 1998.
[22] A. Henderson, The ParaView Guide, Kitware, Inc., United States, 2004.
[23] L. Breiman, Random forests, Machine Learning 45 (1) (2001) 5–32.
[24] R.E. Banfield, L. Hall, K. Bowyer, D. Bhadoria, W. Kegelmeyer, S. Eschrich, A comparison of ensemble creation techniques, in: Multiple Classifier Systems, Fifth International Workshop, vol. 3077 of Lecture Notes in Computer Science, Springer, Cagliari, Italy, 2004, pp. 223–232.
[25] R.E. Banfield, L. Hall, K. Bowyer, W. Kegelmeyer, Ensembles of classifiers from spatially disjoint data, in: Multiple Classifier Systems, Sixth International Workshop, vol. 3541 of Lecture Notes in Computer Science, Springer, Seaside, CA, USA, 2005, pp. 196–205.
[26] W.S. Koegler, W.P. Kegelmeyer, FCLib: a library for building data analysis and data discovery tools, Advances in Intelligent Data Analysis VI IDA 2005 (2005) 192–203.
[27] L. Shoemaker, R.E. Banfield, L.O. Hall, K.W. Bowyer, W.P. Kegelmeyer, Learning to predict salient regions from disjoint and skewed training sets, in: 18th IEEE Conference on Tools with Artificial Intelligence (ICTAI 2006), Arlington, VA, USA, 2006, pp. 116–123.
[28] I.H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques, second ed., Morgan Kaufman, San Francisco, 2005.
[29] P.J. Philips, H. Wechsler, J. Huang, P. Rauss, The FERET database and evaluation procedure for face recognition algorithms, Image and Vision Computing 16 (1998) 295–306.
[30] The facial recognition technology (FERET) database, http://www.itl.nist.gov/iad/humanid/feret/ (accessed 31.12.2006).
[31] D. Bolme, R. Beveridge, M. Teixeira, B. Draper, The CSU face identification evaluation system: its purpose, features and structure, in: Computer Vision Systems, Third International Conference, Lecture Notes in Computer Science, vol. 2626, Springer, Graz, Austria, 2003, pp. 304–313.
[32] R. Beveridge, Evaluation of face recognition algorithms, http://www.cs.colostate.edu/evalfacerec (accessed on 31.12.2006).
[33] N. Otsu, A threshold selection method from gray level histograms, IEEE Trans. Systems, Man and Cybernetics 9 (1979) 62–66.

Please cite this article in press as: L. Shoemaker et al., Using classifier ensembles to label spatially disjoint data, Informat. Fusion (2007), doi:10.1016/j.inffus.2007.08.001