Approximation of Ensemble Boundary using Spectral Coefficients

T. Windeatt, C. Zor and N.C. Camgoz

***Abstract*—A spectral analysis of a Boolean function is proposed for approximating the decision boundary of an ensemble of classifiers, and an intuitive explanation of computing Walsh coefficients for the functional approximation is provided. It is shown that the difference between first and third order coefficient approximation is a good indicator of optimal base classifier complexity. When combining Neural Networks, experimental results on a variety of artificial and real two-class problems demonstrate under what circumstances ensemble performance can be improved. For tuned base classifiers, first order coefficients provide performance similar to majority vote. However, for weak/fast base classifiers, higher order coefficient approximation may give better performance. It is also shown that higher order coefficient approximation is superior to the Adaboost logarithmic weighting rule when boosting weak Decision Tree base classifiers.**

***Index Terms*—Boolean functions, boosting, decision trees, ensemble classifier, multilayer perceptrons, pattern analysis, spectral analysis, supervised learning**

I. INTRODUCTION

Ensembles or multiple classifier systems have become an important and well-recognised method of solving pattern recognition problems in many application areas [1] [2] [3], and there are useful surveys that describe the research approaches to the three phases of classifier generation, selection and aggregation [4]. Classifier generation has the aim of producing accurate yet diverse classifiers and has well established methods [5] although diversity is still an elusive concept [6]. Classifier selection is being actively researched [7] and either a single classifier or ensemble subset can be selected; selection may be static which is based on the training/validation dataset or dynamic, in which selection differs for each test pattern. While dynamic selection may improve generalisation, it has the disadvantage of introducing extra parameters to define the region of competence, competence criterion and selection strategy [5].

In this paper we concentrate on the aggregation phase, which may include both non-trainable and trainable combination rules for classifiers. Trainable rules have the advantage that they can be adapted to the problem at hand, but the disadvantage that over-training may occur unless the training strategy and parameters are carefully chosen [8] [9]. The simplest combination rule is the *majority vote*, which is a non-trainable rule like *sum*, *product*, *max*, *min*, but differs in that only class labels rather than continuous outputs are required. Weighted combination rules have been extensively investigated [10] but there is no established strategy for computing the weights, and the advantage of

according to [12] are easy to over-train unless there are vast amounts of training data. Ensemble pruning is an active research issue [13] and uses many different strategies to reduce the number of base classifiers without sacrificing the performance of the combination; its simplest implementation is to remove classifiers with low estimated weights [14].

Since interesting learning problems are ill-posed [15] we know that some form of parameter tuning will be required. But if we introduce too many parameters, such as with dynamic selection or trainable combination rule, the search space can become too large to be practical and parameters are difficult to tune. In this paper, we propose a novel combination rule that has the simplicity of a non-trainable voting rule, but introduces parameters, the spectral coefficients, that determine how well the ensemble boundary is approximated. While there has not been any previous attempt to explicitly model the ensemble boundary, the closest related approach is the *weighted vote*.

T. Windeatt, C Zor, N. C. Camgoz are with the Centre for Vision, Speech and Signal Processing, University of Surrey, Guildford GU2 7XH, UK.

Consider an ensemble framework of parallel base classifiers. If each base classifier is given a binary decision, and if the problem is two-class, a Boolean mapping is defined. This mapping may be analysed using Walsh spectral coefficients, which was first proposed for pattern recognition over four decades ago [16], although not in the context of ensemble classification. The relationship between added classification error and second order Walsh coefficients was established in [17]. In [13] first and second order Walsh coefficients were used for ensemble pruning. The motivation for using Walsh coefficients in ensemble design is fully explored in [6] and [18]. For further understanding of the meaning and applications of Walsh coefficients see [19]. The Boolean function may be partially specified, noisy and possibly contradictory, and therefore the computation will need to handle this kind of function.

Section II explains the computation of the Walsh coefficients, and Section III uses the Tumer-Ghosh model [20] to explain why the first and third order coefficients can predict over-fitting. In Section IV the datasets are defined and experimental results, when combining Neural Networks (*NN*) and boosting Decision Trees (*DT*), are described and discussed. For the experimental investigation in Section IV, classifier generation uses *NN* base classifiers with random starting weights, and an experimental comparison is made with the *majority* and *weighted vote*. There is no reason why the proposed method could not be combined with dynamic classifier selection and ensemble pruning, but that is not the purpose of this paper, so there is no selection or pruning.

II. DISTRIBUTION OF BINARY PATTERNS

Assume that there are N parallel base classifiers, and let X_m be the N-dimensional decision vector for the m^{th} training pattern, formed from the decisions of the N classifiers. For a two-class supervised learning problem of μ decision vectors, the target label given to each pattern X_m is denoted by $\Omega_m = \Phi(X_m)$ where $m = 1 \ldots \mu$, $\Omega_m \epsilon \{0,1\}$ and Φ is the unknown Boolean function that maps X_m to Ω_m. The binary vector X_m representing the m^{th} original training pattern is given by

$$X_m = (x_{m1}, x_{m2}, \ldots, x_{mN}) \tag{1}$$

where $x_{mi} \in \{0,1\}$ is a vertex in the N-dimensional binary hypercube. ...density function (p.d.f.) we approximate, where j can take any of μ possible values. In other words, we would like to calculate the probability of occurrence of the binary patterns. The following approach is similar to [16], but differs in that we are considering binary patterns in classifier space, rather than the original feature space. A good choice of basis functions for this problem is the Rademacher-Walsh (RW) polynomials, which contain 2^N terms and are formed by taking products of distinct terms of the form $(2x_{mk} - 1)$. The product is taken singly, pairs, triples,..., up to N times. Table I shows the RW discrete polynomial functions which are orthogonal, satisfying the property that

$$\sum_{m=1}^{2^N} \varphi_j(X_m)\varphi_k(X_m) = \begin{cases} 2^N & if\ j = k \\ 0 & if\ j \neq k \end{cases} \tag{2}$$

An approximation using q basis functions and μ vectors is given by

$$\hat{p}(X) = \sum_{j=1}^{q} c_j \varphi_j(X) \quad (3)$$

where coefficients are given by

$$c_k = \frac{1}{2^N \mu} \sum_{i=1}^{\mu} \varphi_k(X_i) \quad (4)$$

TABLE I
RADEMACHER-WALSH POLYNOMIAL FUNCTIONS

j	$\varphi_j(X)$
1	1
2	$2x_1 - 1$
⋮	⋮
$N+1$	$2x_N - 1$
$N+2$	$(2x_1 - 1)(2x_2 - 1)$
⋮	⋮
$N+2+N(N-1)/2$	$(2x_{N-1} - 1)(2x_N - 1)$
$N+2+N(N-1)/2$	$(2x_1 - 1)(2x_2 - 1)(2x_3 - 1)$
⋮	⋮
2^N	$(2x_1 - 1)(2x_2 - 1) \dots (2x_N - 1)$

A. Examples

As an example of basis functions, first consider the family of completely specified 2-dimensional Boolean functions, so that there are 4 patterns and $2^4 = 16$ possible functions. The patterns are (0,0), (0,1), (1,0), (1,1). From the first four rows of Table I, where $j \leq 4$, N = 2, it is easy to verify that (2) is satisfied.

Now consider a 3-dimensional example of a threshold logic function $\Omega = x_1x_2 + x_1x_3 + x_2x_3$, for which we would like to use a linear approximation. A single decision function will be formed by subtracting individual decision functions for the two classes using (3) and (4). The class ω_0 patterns are given by (0,0,0), (0,0,1), (0,1,0), (1,0,0) and class ω_1 patterns by (0,1,1), (1,0,1), (1,1,0), (1,1,1). There are 4 patterns in each class. From Table I, the linear approximation, which represents a *weighted vote*, uses four basis functions denoted by

$$\varphi_1(X) = 1, \quad \varphi_2(X) = (2x_1 - 1), \quad \varphi_3(X) = (2x_2 - 1), \quad \varphi_4(X) = (2x_3 - 1)$$

Since the factor $\frac{1}{2^N}$ is common to all terms in (4), we can neglect it in the computation. For class ω_1 the coefficients are given by

$$c_1 = \frac{1}{4}\sum_{i=1}^{4} \varphi_1(X_i) = \frac{1}{4}(+1 + 1 + 1 + 1) = 1$$

Similarly, $c_2 = c_3 = c_4 = {}^1/_2$

The density function linear approximation for class ω_1

$$\hat{p}(X|\omega_1) = 1 + \frac{1}{2}(2x_1 - 1) + \frac{1}{2}(2x_2 - 1) + \frac{1}{2}(2x_3 - 1)$$

Similar analysis for class ω_0 gives

$$\hat{p}(X|\omega_0) = 1 - \frac{1}{2}(2x_1 - 1) - \frac{1}{2}(2x_2 - 1) - \frac{1}{2}(2x_3 - 1)$$

Assuming prior probabilities can be determined from the number of patterns, so $p(\omega_1) = p(\omega_0) = \frac{1}{2}$, the decision functions are

$$\hat{p}(X|\omega_1)p(\omega_1) = \frac{1}{2} + \frac{1}{4}(2x_1 - 1) + \frac{1}{4}(2x_2 - 1) + \frac{1}{4}(2x_3 - 1)$$

$$\hat{p}(X|\omega_0)p(\omega_0) = \frac{1}{2} - \frac{1}{4}(2x_1 - 1) - \frac{1}{4}(2x_2 - 1) - \frac{1}{4}(2x_3 - 1)$$

Subtracting and multiplying by 2 gives combined decision function

$$d(X) = (2x_1 - 1) + (2x_2 - 1) + (2x_3 - 1)$$

It is easy to verify that $d(X)$ separates the patterns perfectly, with class ω_1 patterns giving $d(X) > 0$ and class ω_0 patterns giving $d(X) < 0$. If we now assume that there are only 3 class ω_1 patterns with (0,1,1) missing, $\hat{p}(X|\omega_1), p(\omega_1), p(\omega_0)$ need to be modified and the combined decision function becomes

$$d(X) = -\frac{1}{7} + \frac{5}{7}(2x_1 - 1) + \frac{3}{7}(2x_2 - 1) + \frac{3}{7}(2x_3 - 1)$$

which still perfectly separates the training patterns, but the unspecified (0,1,1) pattern has $d(X) = 0$.

An alternative interpretation of the spectral coefficients is given in [19]. The first order coefficients, *j = 2 … N+1* in Table I, represent the correlation with the class label. In the above example, note that if class label agrees with x_i then add +1, otherwise add -1. For example, if *j = 2* in Table I and we denote first order coefficient by $s_i \quad i = 1 \dots N$

$$s_1 = c_2 = \frac{1}{2^N \mu} \sum_{m=1}^{\mu} \mathcal{C}(x_{m1}, \Omega_m)$$

where $\mathcal{C}(a,b) = \begin{cases} +1 \; if \; a = b \\ -1 \; if \; a \neq b \end{cases}$

Second order coefficients represent correlation with the logic exclusive-OR (*xor* denoted $\oplus$) of the respective pair of coefficients. For example, if *j = N+2* in Table I, the second order spectral coefficient s_{12} is given by

$$s_{12} = c_5 = \frac{1}{2^N \mu} \sum_{m=1}^{\mu} \mathcal{C}(x_{m1} \oplus x_{m2}, \Omega_m)$$

For third order coefficients

$$s_{ijk} = \frac{1}{2^N \mu} \sum_{m=1}^{\mu} \mathcal{C}(x_{mi} \oplus x_{mj} \oplus x_{mk}, \Omega_m)$$

and similar analysis holds for higher order coefficients.

III. MODELLING SPECTRAL CONTRIBUTION

Fig. 1 shows the two class (ω_1, ω_0) one-dimensional Tumer-Ghosh model [20], with added classification error for k^{th} classifier boundary (E_k) shown as darkly shaded region. The assumption is made that both classes are Gaussian, but only the tails of the distribution are shown. The optimum (Bayes) boundary $(\tilde{x})$ in Fig. 1 is the loci of all points $\tilde{x}: p(\omega_1|\tilde{x}) = p(\omega_0|\tilde{x})$. The estimates $\hat{p}(\omega_1|x), \hat{p}(\omega_0|x)$ for the k^{th} classifier are shown as dashed lines, and cross at $x = x_b$. In Fig. 1 b is the amount that the k^{th} classifier boundary (x_b) differs from the ideal Bayes boundary. Assuming that b is a Gaussian random variable, closed-form expressions may be obtained for E_k [17] [20]. Further details about the model and assumptions may be found in [20].

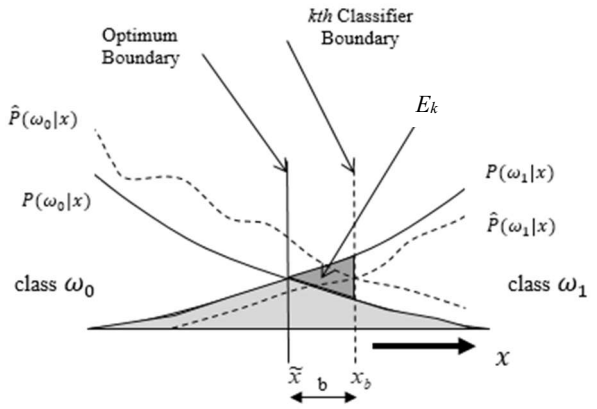

Fig. 1. Tumer-Ghosh model showing real and estimated probabilities and Added Error E_k for k^{th} classifier shown as darkly shaded region

Fig. 2 shows decision boundaries of three *(i,j,k)th* classifiers for which it is assumed that the complexity is not sufficient to approximate the Bayes boundary, so that all classifiers under-fit. Note in Fig. 2 that estimated probabilities corresponding to the three classifiers are omit-

ted for clarity. Mutually exclusive areas under the probability distribution are labelled 1 – 8 in Fig. 2 and area y is given by a_y. E_i corresponds to a_4, E_j to a_4+a_3, E_k to $a_4+a_3+a_2$.

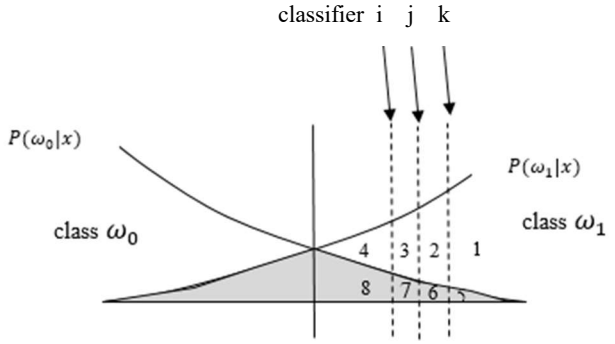

Fig. 2. Tumer Ghosh model showing *i,j,k*th classifier boundaries and areas 1-8

To compute the spectral coefficient contribution, we define n_{pq} to be the number of class ω_p patterns for which φ has value q where $p, q \in \{0,1\}$. Table II shows the first order contribution for each area for the k^{th} classifier. Assuming that there are approximately equal number of class ω_1 and class ω_0 patterns under the tail of the distribution in areas a_5 to a_8, the contributions cancel, so that we need only consider a_1 to a_4. Table III shows the first order contributions for the three classifiers individually, and the last row *ijk* corresponds to third order contribution. For example, for a_1 the individual $\varphi(x_i)$ for each classifier is 1 so third order $\varphi(x_i \oplus x_j \oplus x_k) = 1 \oplus 1 \oplus 1 = 1$. For the first order, we see that patterns in areas a_1 and a_2 are overall positively correlated with ω_1 (2 out of 3 classifiers for a_2), and areas a_3 and a_4 negatively correlated, giving an ensemble decision boundary close to classifier j. The third order contribution is positively correlated for a_1, a_3 and negatively correlated for a_2, a_4. By inspection of Fig. 2, we can see that the sum of a_2+a_4 is likely to be less than a_1+a_3, so we expect an overall positive correlation, which when added to the first order contribution would move the decision boundary closer to $\tilde{x}$.

TABLE II
FIRST ORDER CONTRIBUTIONS FOR *KTH* CLASSIFIER IN FIG. 2

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
ω_1	n_{11}	n_{10}	n_{10}	n_{10}	n_{11}	n_{10}	n_{10}	n_{10}
ω_0					n_{01}	n_{00}	n_{00}	n_{00}

TABLE III
FIRST AND THIRD ORDER CONTRIBUTIONS FOR *I,J,K TH* CLASSIFIERS IN FIG. 2

	a_1	a_2	a_3	a_4
k	n_{11}	n_{10}	n_{10}	n_{10}
j	n_{11}	n_{11}	n_{10}	n_{10}
i	n_{11}	n_{11}	n_{11}	n_{10}
ijk	n_{11}	n_{10}	n_{11}	n_{10}

Now consider the case that base classifiers are optimal, so that classifier *j* is close to $\tilde{x}$, and classifiers *i,k* are approximately equally spaced on either side of the Bayes boundary. The ensemble boundary for first order approximation would then be close to $\tilde{x}$. We may expect that the Boolean function would be quite complex, with approximately equal number of patterns in the two classes under the tail. Areas a_2+a_4 would be approximately equal to a_1+a_3, so that the addition of the third order contribution would be small. This suggests that the difference between the first and the third order coefficients would be indicative of optimal performance. Similarly, if the classifiers severely under-fit so that they are close to the mean of class ω_1, again areas a_2+a_4 would be approximately equal to a_1+a_3. Therefore, as classifier complexity is increased the difference between the first and third order approximations may not be monotonic, which can be seen in Section IV.

IV. EXPERIMENTAL EVIDENCE

A. Datasets and classifiers

There are three types of two-class problems labelled *Real2*, *Art2* and *Multi2* in Table IV which shows the number of patterns, number of features and Bayes estimate. *Real2* is selected from [21], and *Art2* is artificial data taken from [22]. To increase the number of difficult (in terms of boundary complexity) two-class problems, *Multi2* uses multi-class datasets with the Error Correcting Output Coding (*ECOC*) method [23], and approximate equi-split random code matrix. According to *ECOC*, each column of the binary code matrix defines a different two-class problem, with each original class assigned to one of two super-classes. We choose those dichotomies (up to a max number of ten problems) for which the imbalance in data is not greater than 65/35% between the two super-classes. The datasets with less than ten problems are *glass* with 6, *vehicle* with 3 and *dermo* with 5. For *Multi2* the results are reported as an average over the number of problems. The random train/test split is 80/20% for problems in *Real2* and *Multi2*, and for *Art2* there are 600 training patterns and 2400 testing patterns. Experiments for all datasets are repeated ten times and averaged. The Bayes estimate is performed for 90/10% split using a Support Vector Classifier (*SVC*) with polynomial kernel run 100 times. The polynomial degree and regularisation constant are varied, and lowest test error is given. Test error rates will have Bayes estimate subtracted, and the accuracy of the estimate is not crucial as the shape of the plots is not affected.

TABLE IV
DATASETS SHOWING NUM. PATTERNS, NUM. FEATURES, BAYES ESTIMATE, AND (SECTION IV.B) OPTIMAL NUM. EPOCHS, PREDICTED NUM. EPOCHS.

Data	*Problem*	*#pat*	*#feat*	*%BE*	*#Epo (opt)*	*#Epo (pred)*
Real2	card	690	15	9.9	4	4
Real2	credita	690	14	11.1	4	8
Real2	diabetes	768	8	20.1	8	8
Real2	heart	920	35	11.5	8	8
Real2	ion	351	34	5.2	16	16
Real2	vote	435	16	2.2	4	4
Art2	threnorm	3000	20	10.5	8	8
Art2	highleym	3000	2	5.1	16	8
Art2	circular	3000	2	12.8	16	16
Art2	gaussimp	3000	2	12.0	8	8
Art2	banana	3000	2	1.2	8	4
Art2	lithuani	3000	2	1.8	8	8
Multi2	ecoli	336	7	6.3	8	16
Multi2	vehicle	846	18	8.6	16	16
Multi2	glass	214	9	19.2	16	32
Multi2	segment	2310	19	0.2	32	32
Multi2	dermo	366	34	0.4	32	32
Multi2	soybean	683	35	1.9	8	4

In this section, the nomenclature for error rate is: *Majority vote* combiner (*MV*), Walsh coefficient combiner of order y (*Wy*), logarithmic Adaboost combiner (*AD*), base classifier (*BA*) and Bayes estimate (*BE*). The assumption is made that test error rate is specified, otherwise *tr* is

added to indicate training rate (e.g. *MV* is test error, *MVtr* is train error). For computing *Wy*, there are no parameters to set, as with *MV*.

The experiments are designed to test the following hypotheses: Firstly, that *W3-W1tr* is a good indicator of optimal base classifier complexity, secondly that the ensemble approximation using *W1* is a good alternative to *MV*, and thirdly that for weak/fast base classifiers, higher order Walsh approximation may give best performance.

Neural Networks (*NN*) and Decision Trees (*DT*) are used as base classifiers. For *NN*, the number of hidden nodes and training epochs of homogenous (same number of nodes and epochs) single hidden layer multilayer perceptron (*MLP*) base classifiers are systematically varied, and use the Levenberg-Marquardt training algorithm with default parameters. The ensemble has 25 *MLP* base classifiers, the diversity in each being due to random starting weights. In Section 1V.B, Fig. 3 shows various plots for Circular dataset, combining *NN* classifiers that are systematically varied from 4-32 nodes and 2-32 epochs. Preliminary experiments on other datasets showed that by selecting 32 nodes it was possible to achieve optimal error rates as epochs was varied, so all remaining experiments Figs 6-9 in Section IV.B use 32 nodes. Boosting refers to 25 *DT* classifiers using Adaboost implementation in [22] with varying number of decision splits.

B. Combining NN classifiers

Fig. 3 (a) – (d) show test error rates and (e) (f) train error rates. Fig. 3 (a) (b) show mean *BA* and *W1* with *BE* subtracted. As training epochs are increased, the optimal value for *W1* is generally lower than for *BA*, for example at 4 nodes, 8 epochs for *W1* versus 16 epochs for *BA*. Fig. 3 (c) (d) show the difference between *W1* and *MV*, and between *W3* and *W1*. *W1* is never worse than *MV*, and at low epochs is superior, quite dramatically for 4 nodes. *W1* is preferred to *MV* below 8 epochs at 4 nodes, and below 4 epochs at 8 nodes. Similarly, in Fig. 3 (d) W*3* is preferred to *W1* for low epochs. Fig. 3 (e) (f) show train error for (c) (d), and note that train and test curves look similar, so that it may be possible to infer test error performance from train error. In Fig. 3 (f) the difference *W3-W1tr* reaches maximum at 16 epochs for 4 and 8 nodes, and at 8 epochs for 16, 32 nodes which from (a) and (b) is when performance is optimal.

Example decision boundaries are shown for artificial data to demonstrate that *W3* may be superior to *MV* and *W1*. Ensemble decision boundaries for Circular dataset with 4 node *NN* base classifiers are shown for 16 epochs in Fig. 4, and for 4 epochs in Fig. 5. It can be seen from Fig. 4 that all ensemble decision boundaries achieve optimal performance at 16 epochs. Individual boundaries for five base classifiers are also shown in Fig. 5, from which it may be seen that there is great variation in individual boundaries, but only *W3* approaches optimal at 4 epochs.

Fig. 6, Fig. 7 and Fig. 8 show respectively the *Art2*, *Real2* and *Multi2* datasets with *NN* having 32 nodes and varying 2-32 epochs. Each graph shows four curves *W3-W1tr, W1-BE, W1-MV, BA-BE* with *W3-W1tr* multiplied by 5 for clarity. All datasets show similar trend as epochs increases, so that when *W3-W1tr* reaches maximum, *BA* (but not always *W1*) reaches minimum. As explained in Section III, optimality corresponds to a complex Boolean function. In some cases, e.g. *credita* in Fig. 7 at 4 epochs, *W3tr* is higher than *W1tr*. Note also in Fig. 8, that *glass* and *vehicle* demonstrate the non-monotonic behaviour in *W1-W3tr* referred to at the end of Section III. For all datasets, the plot of *W1-MV* indicates that *W1* is never worse than *MV*. In Table IV is listed the optimal (according to *BA*) and predicted (according to *W1-W3tr*) epochs at 32 nodes for all datasets.

To test robustness of the method, Fig. 9 shows average over *Real2* datasets as percentage label noise (probability of flipping labels) is increased. At 5% *(W3-W1)tr* peaks at 4 epochs which is optimal, but for 10% noise *(W3-W1)tr* peaks at 4 epochs compared to optimal at 8 epochs. This suggests that label noise of 10% or greater may affect the ability to predict optimal epochs, but further study is warranted.

To determine the performance for data with higher dimension and more patterns, 'dog' vs 'cat', the most difficult of the two class problems in the CIFAR-10 dataset [24] was selected. There are 6000 patterns in each class, the input images have been converted to grey scale 32x32 and the train/test split is chosen as 20/80%, to encourage overfitting. The architecture consists of the following layers: 2D convolutional layer of 40 filters having width and height of 5; rectified linear unit; max pooling with non-overlapping pooling regions that downsamples by factor of 2; fully connected output size 2; softmax. Training uses stochastic gradient descent with momentum and variable number of epochs. Fig. 10 shows mean values over ten runs, demonstrating that over-fitting of the base classifier is accurately predicted by *(W3-W1)tr* at 40 epochs.

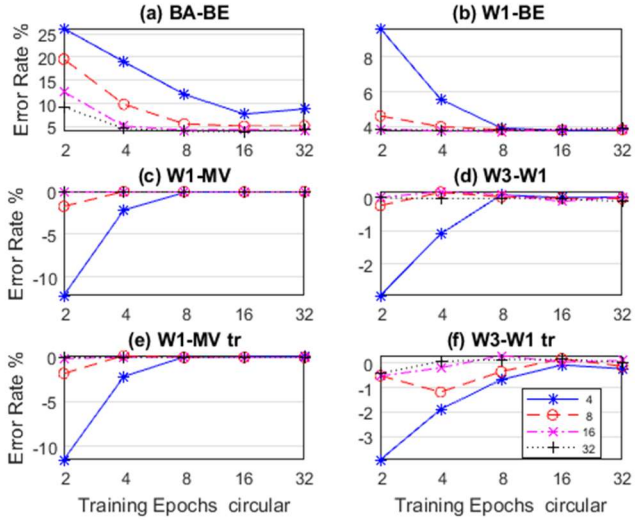

Fig. 3. Circular data with *NN* varying nodes/epochs

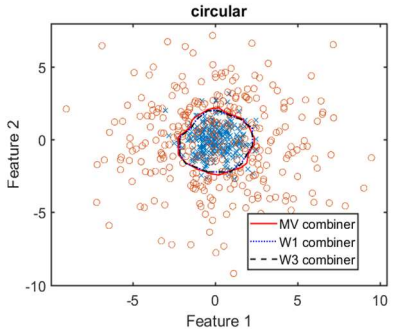

Fig. 4. Combiner boundaries for 4 nodes and 16 epochs

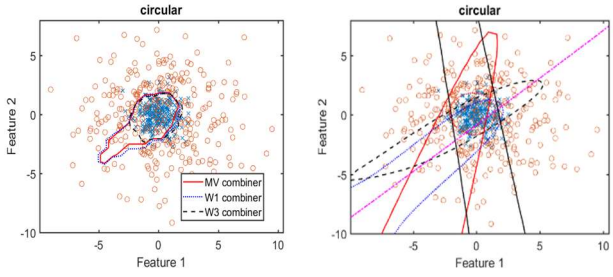

Fig. 5. Combiner and individual boundaries (right) for 4 nodes and 4 epochs

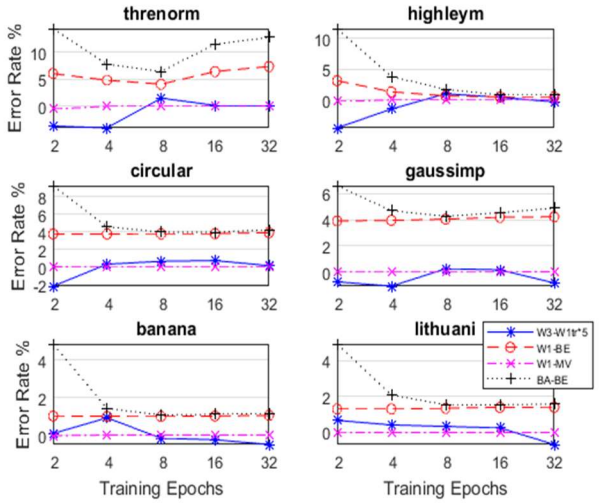

Fig. 6. *Art2* datasets with *NN* varying epochs for 32 nodes

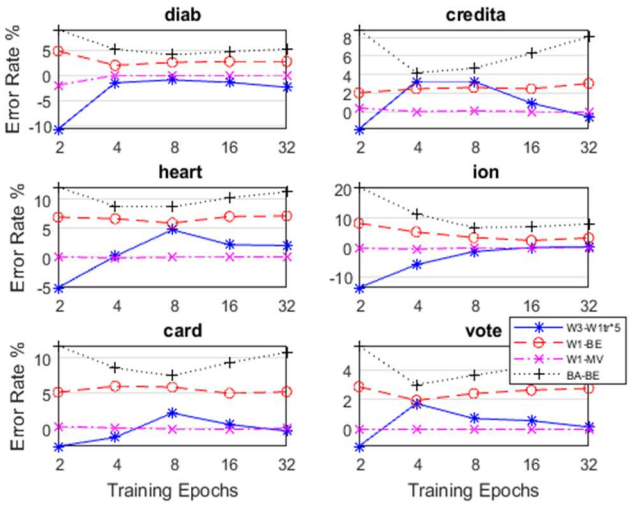

Fig. 7. *Real2* datasets with *NN* varying epochs for 32 nodes

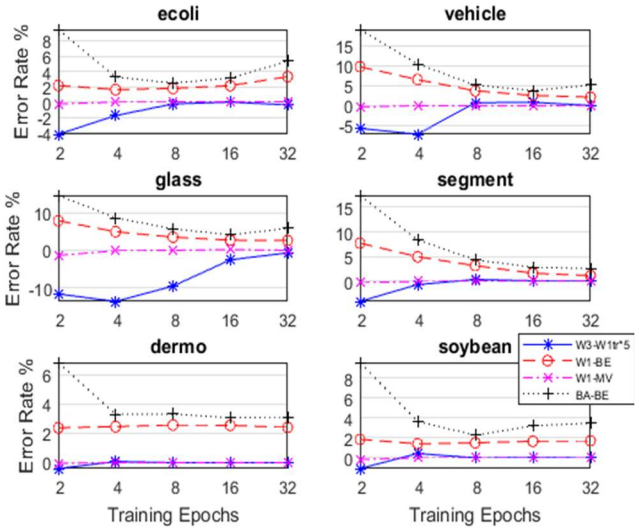

Fig. 8. *Multi2* datasets with *NN* varying epochs for 32 nodes

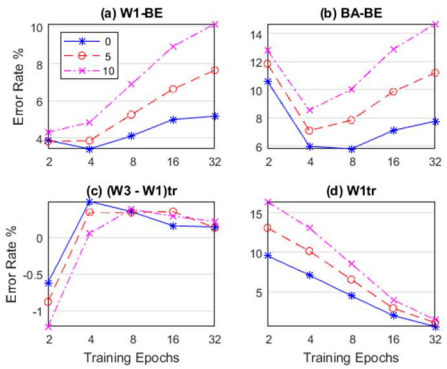

Fig. 9. Mean over *Real2* datasets for 32 nodes as label noise is increased

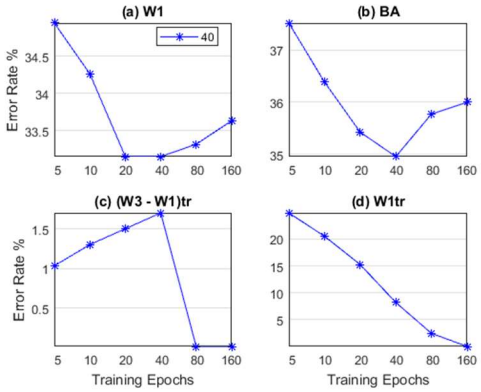

Fig. 10. CIFAR dataset varying epochs for 40 filters

C. Boosting DT and Combining NN with Higher Order Walsh

Fig. 11, Fig. 12 and Fig. 13 show results for boosting 1-3 *DT* splits with the x-axis showing weighted logarithmic combiner (*AD*) and increasing order of Walsh combiners, *W1 to W9*. *AD* is sometimes better than *W1* and sometimes worse so there is no clear trend. Note that the best result can sometimes be achieved using higher order Walsh approximation, for example *W7*/*highleym*, *W3*/*gaussimp*, *W5*/*diab*, *W5*/*vote W3*/*demo* and *W3*/*soybean*. Particularly for the Decision Stump, that is *DT* with 1 split, higher order Walsh combining gives the best result.

Mean over all eighteen datasets in Table IV for combining *NN* with 1 epoch and 16, 32, 64 nodes is shown in Fig. 14. On average, the *W5* approximation achieves the best performance.

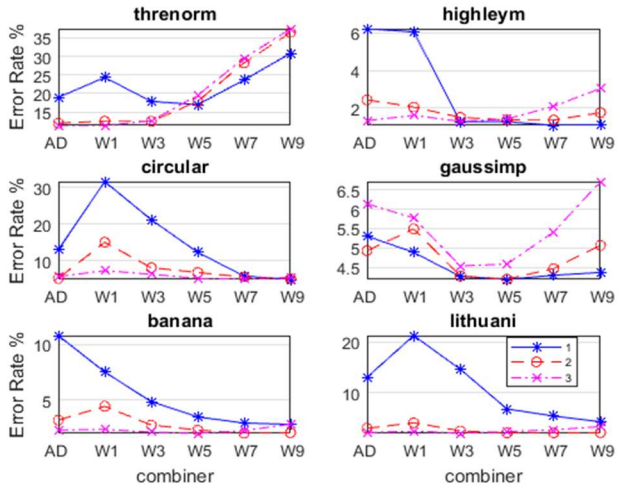

Fig. 11. *Art2* datasets with boosting *DT* splits 1-3

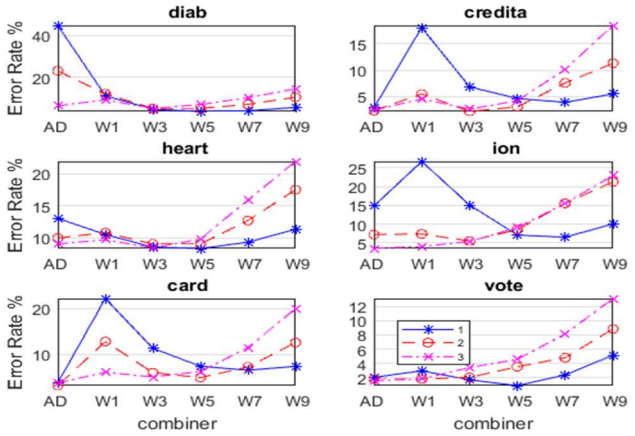

Fig. 12. *Real2* datasets with boosting *DT* splits 1-3

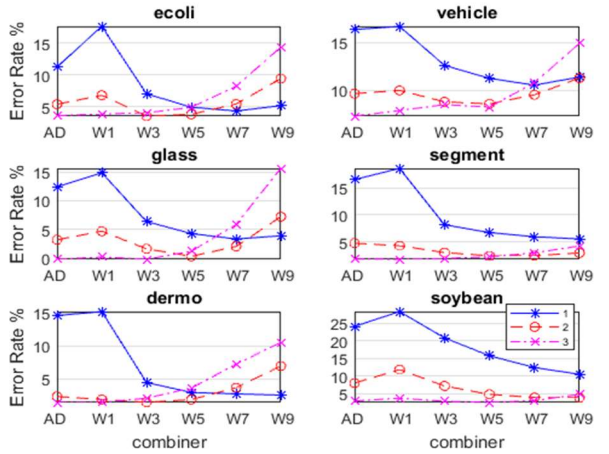

Fig. 13. *Multi2* datasets with boosting *DT* splits 1-3

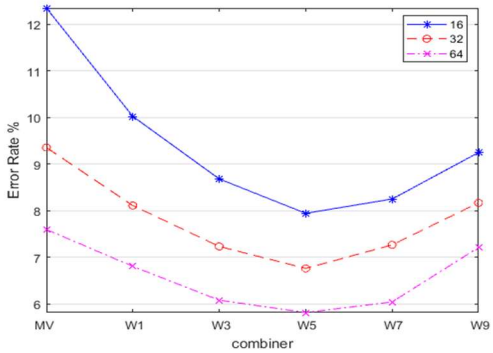

Fig. 14. Mean over 18 datasets in Table IV, *NN* with 1 epoch and 16-64 nodes

V. DISCUSSION

First order Walsh coefficient approximation *W1* provides an estimate for a *weighted vote*, and there are no parameters to set. Experimental results show that when base classifiers are optimised, there is no difference between *W1* and *majority vote MV*. Compared with *MV*, there is an advantage to using *W1* or higher order coefficients for weak *MLP* base classifiers. Also, for boosting weak Decision Tree classifiers, higher order coefficients give better performance than Adaboost logarithmic weighting rule, but the optimal order of the coefficients is problem-dependent.

It has also been shown that the number of epochs of *MLP* base classifiers may be selected from the training set using third and first order coefficients, and therefore a validation set is not required. However, the results are based on averages and individual runs can be noisy, so that it may be necessary to introduce filtering to facilitate a practical design method.

REFERENCES

[1] D. Brzezinski and J. Stefanowski, Reacting to different types of concept drift, *IEEE Trans Neural Networks and Learning Systems*, vol. 25, no. 1, pp. 81-95, 2014.

[2] S. Ren et al., Knowledge-maximised ensemble algorithm for different types of concept drift, Information Sciences 430-431, pp. 261-281, 2018.

[3] Y Sun et al., Concept drift adaptation by exploiting historical knowledge, *IEEE Trans Neural Networks and Learning Systems*, to be published, DOI 10.1109/TNNLS.2017.2775225.

[4] B. Krawczyk et al., Ensemble learning for data stream analysis,: A survey, *Information Fusion*, vol. 37, pp132-156, 2017.

[5] R. M. O. Cruz, R. Sabourin and G. D. C. Cavalcanti, Dynamic classifier selection: Recent advances and perspectives, *Information Fusion* vol. 41, pp 195-216, 2018.

[6] T. Windeatt, "Accuracy/ Diversity and ensemble classifier design," *IEEE Trans Neural Networks*, vol. 17, pp. 1194- 1211, Sept. 2006.

[7] A. L. Brun et al., A framework for dynamic classifier selection oriented by the classification problem difficulty, *Pattern Recognition*, vol. 76, pp. 175-90, 2018.

[8] R.P.W. Duin, The combining classifier; to train or not to train Proc. 16th ICPR, 2002, pp. 765-770.

[9] R. K. Sevakula and N. K. Verma, Assessing generalisation ability of majority vote point classifiers, *IEEE Trans Neural Networks and Learning Systems*, vol. 28, no. 12, pp. 2985-2997, 2017.

[10] Z. Yu et al., Progressive subspace ensemble learning, *Pattern Recognition*, vol. 60, pp. 692-705, 2016.

[11] G. Fumera and F. Roli, A theoretical and experimental analysis of linear combiners for multiple classifier systems, *IEEE Trans. On Pattern Analysis and Machine Intelligence*, vol. 27, no. 6, 2005.

[12] L. I. Kuncheva, Combining Pattern Classifiers: Methods and algorithms, Wiley, 2004.

[[13] T. Windeatt and C. Zor "Ensemble Pruning using spectral coefficients," *IEEE Trans Neural Networks and Learning Systems*, vol. 24, no. 4, pp. 673 - 678, 2013.

[14] B. Krawczyk and M. Wozniak, Untrained weighted classifier combination with embedded ensemble pruning, *Neurocomputing*, vol. 196, pp. 14-22, 2016.

[15] A. N. Tikhonov and V. A. Arsenin, Solutions of Ill-posed Problems, Winston & Sons, Washington, 1977.

[16] J.T Tou and R. C. Gonzales, *Pattern Recognition Principles*, Addison-Wesley, 1974, pp. 151-4.

[17] T. Windeatt and C. Zor, "Minimising Added Classification Error using Walsh Coefficients", *IEEE Trans Neural Networks*, vol. 22 no. 8, pp. 1334- 1339, 2011.

[18] T. Windeatt, "Vote Counting Measures for Ensemble Classifiers," *Pattern Recognition, vol.* 36, no. 12, pp. 2743-2756, 2003.

[19] L. Hurst, D. M. Miller, and J. Muzio, *Spectral Techniques in Digital Logic*, Academic Press, 1985.

[20] K. Tumer and J. Ghosh, "Error correlation and error reduction in ensemble classifiers," *Connection Science*, vol. 8, no. 3, pp. 385-404, 1996.

[21] C. J. Merz and P. M. Murphy, UCI repository of ML databases, [Online], http://www.ics.uci.edu/~mlearn/MLRepository.html.

[22] R.P.W. Duin, P. Juszczak, P. Paclik, E. Pekalska, D. de Ridder, D.M.J. Tax, S. Verzakov PRTools4.1, A Matlab Toolbox for Pattern Recognition, Delft University of Technology, 2007.

[23] T. G. Dietterich and G. Bakiri, Solving Multi-class Learning Problems via Error-Correcting Output Codes, *J. Artificial Intelligence Research vol.* 2, 1995, 263-286.

[24] A. Krizhevsky, and G. Hinton, Learning multiple layers of features from tiny images. Master's thesis, Dept. of Computer Science, U. of Toronto, 2009.