8

Reputation-Based Neural Network Combinations

Mohammad Nikjoo, Azadeh Kushki, Joon Lee, Catriona Steele and Tom Chau
Bloorview Research Institute, Holland Bloorview Kids Rehabilitation Hospital
Toronto Rehabilitation Institute
Institute of Biomaterials and Biomedical Engineering
Electrical and Computer Engineering, University of Toronto
Canada

1. Introduction

The exercise of combining classifiers is primarily driven by the desire to enhance the performance of a classification system. There may also be problem-specific rationale for integrating several individual classifiers. For example, a designer may have access to different types of features from the same study participant. For instance, in the human identification problem, the participant's voice, face, and handwriting provide different types of features. In such instances, it may be sensible to train one classifier on each type of feature (Jain et al., 2000). In other situations, there may be multiple training sets, collected at different times or under slightly different circumstances. Individual classifiers can be trained on each available data set (Jain et al., 2000; Xu et al., 1992). Lastly, the demand for classification speed in online applications may necessitate the use of multiple classifiers (Jain et al., 2000).

Optimal combination of multiple classifiers is a well-studied area. Traditionally, the goal of these methods is to improve classification accuracy by employing multiple classifiers to address the complexity and non-uniformity of class boundaries in the feature space. For example, classifiers with different parameter choices and architectures may be combined so that each classifier focuses on the subset of the feature space where it performs best. Well-known examples of these methods include bagging (Breiman, 1996a) and boosting (Bauer & Kohavi, 1999).

Given the universal approximation ability of neural networks such as multilayer perceptrons and radial basis functions(Haykin, 1994), there is theoretical appeal to combine several neural network classifiers to enhance classification. Indeed, several methods have been developed for this purpose, including, for example, optimal linear combinations (Ueda, 2002) and mixture of experts (Jacobs et al., 1991), and negative correlation (Chen & Yao, 2009) and evolving neural network ensembles (Yao & Islam, 2008). In these methods, all base classifiers are generally trained on the same feature space (either using the entire training set or subsets of the training set). While these methods have proven effective in many applications, they are associated with numerical instabilities and high computational complexity in some cases (Bauer & Kohavi, 1999).

Another approach to classifier combination is to train the base classifiers on different feature spaces. This approach is advantageous in combating the undesirable effects of associated with high-dimensional feature spaces (curse of dimensionality). Moreover, the feature sets can be chosen to minimize the correlation between the individual base classifiers to further improve the overall accuracy and generalization power of classification (Ueda, 2002). These methods are also highly desirable in situations where heterogeneous feature combinations are used.
Combination of classifiers based on different feature has been generally accomplished through fixed classification rules. These rules may select one classifier output among all available outputs (for example, using the minimum or maximum operator), or they may provide a classification decision based on the collective outputs of all classifiers (for example, using the mean, median, or voting operators)(Bloch, 2002; Kuncheva, 2002). Among the latter, the simplest and most widely applied rule is the majority vote (Hull et al., 1988; Suen et al., 1990). Many authors have demonstrated that classification performance improves beyond that of the single classifier scenario when multiple classifier decisions are combined via a simple majority vote (Nadal et al., 1990; Xu et al., 1992). Xu et al. (1992) further introduced the notion of weighted majority voting by incorporating classifier-specific beliefs which reflect each classifier's uncertainty about a given test case. Unfortunately, this method does not deal with the degenerate case when one or more beliefs are zero - a situation likely to occur in multi-class classification problems. Moreover, this classifier relies on the training data set to derive beliefs values for each classifier. This approach, therefore, risks overfitting the classifier to the training set and a consequent degradation in generalization power.
In this chapter, we describe a method for combining several neural network classifiers in a manner which is computationally inexpensive and does not demand additional training data beyond that needed to train individual classifiers. Our reputation method will build on the ideas of (Xu et al., 1992). In the following, we present notation that is used throughout the manuscript and detail the majority voting algorithm using this notation. The following presentation is adapted from (Nikjoo et al., 2011).

1.1 Notation

Assume the time series, S, is the pre-processed version of an acquired signal. Also let $\Theta = \{\theta_1, \theta_2, ..., \theta_L\}$ be a set of $L \geq 2$ classifiers and $\Omega = \{\omega_1, \omega_2, ..., \omega_c\}$ be a set of $c \geq 2$ class labels, where $\omega_j \neq \omega_k$, $\forall j \neq k$. Without loss of generality, $\Omega \subset \mathbb{N}$. The input of each classifier is the feature vector $x \epsilon \Re^{n_i}$, where n_i is the dimension of the feature space for the i^{th} classifier θ_i, whose output is a class label $\omega_j, j = 1, ..., c$. In other words, the i^{th} classifier, $i = 1, \ldots, L$, is a functional mapping, $\Re^{n_i} \rightarrow \Omega$, which for each input x gives an output $\theta_i(x) \in \Omega$. Generally, the classifier function could be linear or non-linear. It is assumed that for the i^{th} classifier, a total number of d_i subjects are assigned for training. The main goal of combining the decisions of different classifiers is to increase the accuracy of class selection.

1.2 Majority voting algorithm

In a multi-classifier system, the problem is to arrive at a global decision $\Theta^*(x) = \omega_j$ given a number of local decisions $\theta_i(x) \in \Omega$, where generally (Xu et al., 1992):

$$\theta_1(x) = \theta_2(x) = ... = \theta_L(x). \tag{1}$$

In the literature, a classical approach for solving this problem is *majority voting* (Hull et al., 1988) (Suen et al., 1990). To express this idea mathematically, we define an indicator function

$$I_i(x, \omega_j) = I(\theta_i(x) - \omega_j) = \begin{cases} 1, & \text{when } \theta_i(x) = w_j, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Now, using (2), the majority voting rule can be expressed as follows:

$$\Theta^*(x) = \begin{cases} \omega_{max}, & \text{if } \max_{\omega_j} I_\Omega(x, \omega_j) > L/2, \\ Q, & \text{otherwise,} \end{cases} \tag{3}$$

where $\omega_{max} = \arg\max_{w_j} I_\Omega(x, \omega_j)$, $I_\Omega(x, \omega_j) = \sum_{i=1}^{L} I_i(x, \omega_j), j = 1, ..., c$, and $Q \notin \Omega$ is the rejection state. In other words, given a feature vector, each classifier votes for a specific class. The class with the majority of votes is selected as the *candidate class*. If the candidate class earns more than half of the total votes, it is selected as the final output of the system. Otherwise, the feature vector is rejected by the system.

The majority voting algorithm is computationally inexpensive, simple to implement and applicable to a wide array of classification problems (Lam & Suen, 1997) (Jain et al., 2000). Despite its simplicity, majority voting can significantly improve upon the classification accuracies of individual classifiers (Lam & Suen, 1997). However, this method suffers from a major drawback; the decision heuristic is strictly *democratic*, meaning that the votes from different classifiers are always equally weighted, regardless of the past performance of individual classifiers. Therefore, votes of *weak classifiers*, i.e., classifiers whose performance only slightly exceeds that of the random classifier, can diminish the overall performance of the system when they have the majority. To exemplify this issue, consider a classification system with $c = 2$ classes, $\Omega = \{\omega_1, \omega_2\}$, and $L = 3$ classifiers, $\Theta = \{\theta_1, \theta_2, \theta_3\}$, where two are weak classifiers with 51% average accuracy while the remaining one is a strong classifier with 99% average accuracy. Now assume that for a specific feature vector both the weak classifiers vote for ω_1 but the strong classifier votes for ω_2. Based on the majority voting rule, ω_1 is preferred over ω_2, which is mostly likely an incorrect classification.

As it is discussed in (Tresp & Taniguchi, 1995) (Jacob et al., 1991) (Jain et al., 2000), in practice, there are various situations in which the majority vote may be suboptimal. Motivated by the above, we propose a novel algorithm for combining several classifiers based on their individual *reputations*, numerical weights that reflect each classifier's past performance. The algorithm is detailed in the next section and again is adapted from Nikjoo et al. (2011).

2. Reputation-based voting algorithm

To mitigate the risk of the overall decision being unduly influenced by poorly performing classifiers, we propose a novel fusion algorithm which extends the majority voting concept to acknowledge the past performance of classifiers. To measure the past performance of the i^{th} classifier, we define a measure called *reputation*, $r_i \in \Re$, $0 \leq r_i \leq 1$. For a classifier with high performance, the reputation is close to 1 while a weak classifier would have a reputation value close to 0. For each feature vector, both the majority vote and the reputation of each classifier contribute to the final global decision. The collection of reputation values for L classifiers constitutes the reputation set, $R = \{r_1, r_2, ..., r_L\}$. Each classifier is mapped to a real-valued reputation, r_i, namely,

$$r_i = r(\theta_i) = \alpha, i = 1, ..., L, \tag{4}$$

where $r : \Theta \rightarrow [0,1]$, $\alpha \in \Re$ and $0 \leq \alpha \leq 1$. To determine the reputation of each classifier, we utilize a *validation set* in addition to the classical training and test sets. Specifically, the performance of the trained classifiers on the validation data determines their reputation values. Now, we have all the necessary tools to explain the proposed algorithm.

1. For a classification problem with $c \geq 2$ classes, we design and develop $L \geq 2$ individual classifiers. The proposed algorithm is especially useful if the individual classifiers are independent. This condition can be guaranteed by using different training sets or using various resampling techniques such as bagging (Breiman, 1996b) and boosting (Schapire, 1990). Unlike some of the previous work (Lee & Srihari, 1993) (Hull et al., 1988), there is no restriction on the number of classifiers L and this value can be either an odd or an even number. Also, it should be noted here that, in general, the feature space dimension, n_i, of each classifier could be different and the number of training exemplars, d_i, for each classifier could be unique.
2. After training the L classifiers individually, the respective performance of each is evaluated using the validation set and a reputation value is assigned to each classifier. The validation sets are disjoint from the training sets. It is important to note that here we use two different validation sets. The first one is the traditional validation set which is used repeatedly until the classifier is satisfactorily trained (Duda et al., 2000). In contrast, the second validation set is used to calculate the reputation values of individual classifiers. The accuracy of each classifier is estimated with the corresponding validation set and normalized to $[0,1]$ to generate a reputation value. For instance, a classifier, θ_i, with 90% accuracy (on the latter validation set mentioned above) has a reputation $r_i = 0.9$.
3. Now, for each feature vector, x, in the test set, L decisions are made using L distinctive classifiers:

$$\Omega(x) = \{\theta_1(x), \theta_2(x), ..., \theta_L(x)\}. \tag{5}$$

4. To arrive at a final decision, we consider the votes of the classifiers with high reputations rather than simply selecting the majority class. First, we sort the reputation values of the classifiers in descending order,

$$R^* = \{r_{1^*}, r_{2^*}, ..., r_{L^*}\}, \tag{6}$$

such that $r_{1^*} \geq r_{2^*} \geq ... \geq r_{L^*}$. Then, using this set, we rank the classifiers to obtain a reputation-ordered set of classifiers, Θ^*.

$$\Theta^* = \begin{pmatrix} \theta_{1^*} \\ \theta_{2^*} \\ \vdots \\ \theta_{L^*} \end{pmatrix}. \tag{7}$$

The first element of this set corresponds to the classifier with the highest reputation.

5. Next, we examine the votes of the first m elements of the reputation-ordered set of classifiers, with

$$m = \begin{cases} \frac{L}{2}, & \text{if } L \text{ is even,} \\ \frac{L+1}{2}, & \text{if } L \text{ is odd.} \end{cases} \tag{8}$$

If the top m classifiers vote for the same class, ω_j, we accept the majority vote and take ω_j as the final decision of the system. However, if the votes of the first m classifiers are not equal, we consider the classifier's individual reputations (Step 2).

6. Let $p(\omega_j)$ be the *prior probability* of class ω_j. As before, $\Theta(x) = \{\theta_1(x), \theta_2(x), ..., \theta_L(x)\}$ represents the local decisions made by different classifiers about the input vector x. The probability that the combined classifier decision is ω_j given the input vector x and the individual local classifier decisions is denoted as the *posterior probability*,

$$p(\omega_j|\theta_1(x), \theta_2(x), ..., \theta_L(x)) \tag{9}$$

Clearly, we should choose the class which maximizes this probability.

$$\arg\max_{\omega_j} p(\omega_j|\theta_1(x), \theta_2(x), ..., \theta_L(x)), \quad j = 1, ...c. \tag{10}$$

To estimate the posterior probability we use *Bayes formula*. For notational simplicity we drop the argument x from the local decisions.

$$p(\omega_j|\theta_1, ..., \theta_L) = \frac{p(\theta_1, ..., \theta_L|\omega_j)p(\omega_j)}{p(\theta_1, ..., \theta_L)}, \tag{11}$$

where $p(\theta_1, ..., \theta_L|\omega_j)$ is the *likelihood* of ω_j and $p(\theta_1, ..., \theta_L)$ is the *evidence* factor, which is estimated using the *law of total probability*

$$p(\theta_1, ..., \theta_L) = \sum_{j=1}^{c} p(x, \theta_1, ..., \theta_L|\omega_j)p(\omega_j). \tag{12}$$

By assuming that the classifiers are independent of each other, the likelihood can be written as

$$p(\theta_1, ..., \theta_L|\omega_j) = p(\theta_1|\omega_j)...p(\theta_L|\omega_j). \tag{13}$$

Substituting (12) into the Bayes rule (11) and then replacing the likelihood term with (13), we obtain,

$$p(\omega_j|\theta_1, ..., \theta_L) = \frac{\prod_{i=1}^{L} p(\theta_i|\omega_j)p(\omega_j)}{\sum_{t=1}^{c}\prod_{i=1}^{L} p(\theta_i|\omega_t)p(\omega_t)}. \tag{14}$$

To calculate the local likelihood functions, $p(\theta_i|\omega_j)$, we use the reputation values calculated in Step 6. When the correct class is ω_j and classifier θ_i classifies x into the class ω_j, i.e., $\theta_i(x) = \omega_j$, we can write

$$p(\theta_i = \omega_j|\omega_j) = r_i. \tag{15}$$

In other words, $p(\theta_i = \omega_j|\omega_j)$ is the probability that the classifier θ_i correctly classifies x into class ω_j when x actually belongs to this class. This probability is exactly equal to the reputation of the classifier. On the other hand, when the classifier categorizes x incorrectly, i.e., $\theta_i(x) \neq \omega_j$ given that the correct class is ω_j, then

$$p(\theta_i \neq \omega_j|\omega_j) = 1 - r_i. \tag{16}$$

When there is no known priority among classes, we can assume equal prior probabilities. Hence,

$$p(\omega_1) = p(\omega_2) = \ldots = p(\omega_c) = \frac{1}{c}. \tag{17}$$

Finally, for each class, ω_j, we compute the a posteriori probabilities as given by (14) using (15), (16), and (17). The class with the highest posterior probability is selected as the final decision of the system and the input subject x is categorized as belonging to this class.

The advantage of the reputation-based algorithm over the majority voting algorithm lies in the fact that the former has a higher probability of correct consensus and a faster rate of convergence to the peak probability of correct classification (Nikjoo et al., 2011).

3. Discriminating between healthy and abnormal swallows

We apply the proposed algorithm to the problem of swallow classification. Specifically, the problem is to differentiate between safe and unsafe swallowing on the basis of dual-axis accelerometry (Damouras et al., 2010; Sejdic, Komisar, Steele & Chau, 2010). The basic idea is to decompose a high dimensional classification problem into 3 lower dimensional problems, each with a unique subset of features and a dedicated classifier. The individual classifier decisions are then melded according to the proposed reputation algorithm.

3.1 Signal acquisition and pre-processing

In this chapter, we consider a randomly selected subset of 100 healthy swallows and 100 dysphagic swallows from the larger database originally introduced and analyzed in (Lee et al., 2009). Briefly, dual-axis swallowing accelerometry data were collected at 10 kHz from 24 adult patients (mean age 64.8 ± 18.6 years, 2 males) with dysphagia and 17 non-dysphagic persons (mean age 46.9 ± 23.8 years, 8 males). Patients provided an average number of 17.8 ± 8.8 swallows while non-dysphagic participants completed 19 swallow sequences each. Both groups swallowed boluses of different consistencies. For more details of the instrumentation and swallowing protocol, please see (Lee et al., 2009). It has been shown in (Lee et al., 2008) that the majority of power in a swallowing vibration lies below 100Hz. Therefore, we downsampled all signals to 1KHz. Then, individual swallows were segmented according to previously identified swallow onsets and offsets (Lee et al., 2009). Each segmented swallow was denoised using a 5! -level discrete Daubechies-5 wavelet transform. To remove low-frequency motion artifacts due to bolus intake and participant motion, each signal was subjected to a 4th-order highpass Butterworth filter with a cutoff frequency of 1Hz.

3.2 Feature extraction

Let S be a pre-processed acceleration time series, $S = \{s_2, s_2, ..., s_n\}$. As in previous accelerometry studies, signal features from three different domains were considered (Lee et al., 2010; 2009). The different genres of features are summarized below.

1. **Time-Domain Features**
   - Mean: The sample mean of a distribution is an unbiased estimation of the location of that distribution. The sample mean of the time series S can be calculated as

$$\mu_s = \frac{1}{n} \sum_{i=1}^{n} s_i. \tag{18}$$

- Variance: The variance of a distribution measures its spread around the mean and reflects the signal's power. The unbiased estimation of variance can be obtained as

$$\sigma_s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(s_i - \mu_s)^2. \tag{19}$$

- Skewness: The skewness of a distribution is a measure of the symmetry of a distribution. This factor can be computed as follows

$$\gamma_{1,s} = \frac{\frac{1}{n}\sum_{i=1}^{n}(s_i - \mu_s)^3}{(\frac{1}{n}\sum_{i=1}^{n}(s_i - \mu_s)^2)^{1.5}}. \tag{20}$$

- Kurtosis: This feature reflects the peakedness of a distribution and can be found as

$$\gamma_{2,s} = \frac{\frac{1}{n}\sum_{i=1}^{n}(s_i - \mu_s)^4}{(\frac{1}{n}\sum_{i=1}^{n}(s_i - \mu_s)^2)^{2}}. \tag{21}$$

2. **Frequency-Domain Features**
   - The peak magnitude value of the Fast Fourier Transform (FFT) of the signal S is used as a feature. All the FFT coefficients are normalized by the length of the signal, n.
   - The centroid frequency of the signal S (Sejdic, Komisar, Steele & Chau, 2010) can be estimated as

$$\hat{f} = \frac{\int_0^{f_{max}} f|F_s(f)|^2 df}{\int_0^{f_{max}} |F_s(f)|^2 df}, \tag{22}$$

     where $F_s(f)$ is the Fourier transform of the signal S and f_{max} is the Nyquist frequency (5KHz in this study).
   - The bandwidth of the spectrum can be computed using the following formula

$$BW = \sqrt{\frac{\int_0^{f_{max}} (f-\hat{f})^2|F_s(f)|^2 df}{\int_0^{f_{max}} |F_s(f)|^2 df}}. \tag{23}$$

3. **Information-Theory-Based Features**
   - Entropy Rate (Porta et al., 2001): Porta et al. (2001) introduced a new method for measuring the entropy rate in a signal which quantifies the extent of regularity in that signal. They showed that this rate is useful for signals with some relationship among consecutive signal points. Lee et al. (Under review) used the entropy rate for the classification of healthy and abnormal swallowing. Following their approach, we first normalized the signal S to zero-mean and unit variance. Then, we quantized the normalized signal into 10 equally spaced levels, represented by the integers 0 to 9, ranging from the minimum to maximum value. Now, the sequence of U consecutive points in the quantized signal, $\hat{S} = \{\hat{s}_1, \hat{s}_2, ..., \hat{s}_3\}$, was coded using the following equation

$$a_i = \hat{s}_{i+U-1}.10^{U-1} + ... + \hat{s}_i.10^0, \tag{24}$$

with $i = 1, 2, ..., n - U + 1$. The coded integers comprised the coding set $A_U = \{a_1, ..., a_{n-U+1}\}$. Using the Shannon entropy formula, we estimated entropy

$$E(U) = -\sum_{t=0}^{10^U-1} P_{A_U}(t) \ln P_{A_U}(t), \tag{25}$$

where $p_{A_U}(t)$ represents the probability of observing the value t in A_U, approximated by the corresponding sample frequency. Then, the entropy rate was normalized using the following equation

$$NE(U) = \frac{E(U) - E(U-1) + E(1).\beta}{E(1)}, \tag{26}$$

where β was the percentage of the coded integers in A_L that occurred only once. Finally, the regularity index $\rho \in [0, 1]$ was obtained as

$$\rho = 1 - \min NE(U), \tag{27}$$

where a value of ρ close to 0 signifies maximum randomness while ρ close to 1 indicates maximum regularity.

- Memory (Lee et al., 2010): To calculate the memory of the signal, its autocorrelation function was computed from zero to the maximum time lag. Then, it was normalized such that the autocorrelation at zero lag was unity. The memory was estimated as the time duration from zero to the point where the autocorrelation decays to $1/e$ of its zero lag value.
- Lemple-Ziv (L-Z) complexity (Lempel & Ziv, 1976): The L-Z complexity measures the predictability of a signal. To compute the L-Z complexity for signal S, first, the minimum and the maximum values of signal points were calculated and then, the signal was quantized into 100 equally spaced levels between its minimum and maximum values. Then, the quantized signal, $B_1^n = \{b_1, b_2, ..., b_n\}$, was decomposed into T different blocks, $B_1^n = \{\psi_1, \psi_2, ..., \psi_T\}$. A block ψ was defined as

$$\Psi = B_j^\ell = \{b_j, b_{j+1}, ..., b_\ell\}, 1 \leq j \leq \ell \leq n. \tag{28}$$

The values of the blocks can be calculated as follows:

$$\Psi = \begin{cases} \psi_m = b_1, & \text{if m=1,} \\ \psi_{m+1} = B_{h_m+1}^{h_{m+1}}, & m \geq 1, \end{cases} \tag{29}$$

where h_m is the ending index for ψ_m, such that ψ_{m+1} is a unique sequence of minimal length within the sequence $B_1^{h_{m+1}-1}$. Finally, the normalized L-Z complexity was calculated as

$$LZ = \frac{T \log_{100} n}{n}. \tag{30}$$

3.3 Classification

We trained 3 separate back-propagation neural network (NN) classifiers (Duda et al., 2000), one for each genre of signal feature outlined above. Hence, the feature space dimensionalities

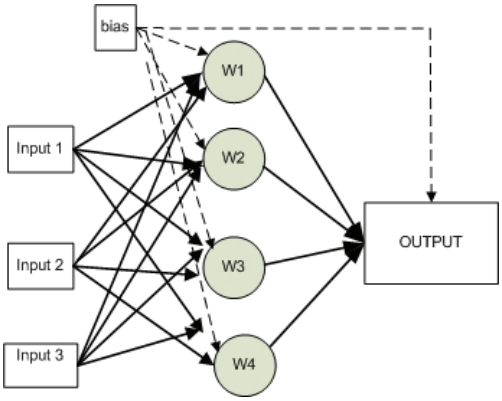

Fig. 1. The schematic of back-propagation neural network with 3 inputs and 4 hidden layers

for the classifiers were 4 (NN with time features), 3 (NN with frequency features) and 3 (NN with information-theoretic features). Each neural net classifier had 2 inputs, 4 hidden units and 1 output. Figure 1 shows the schematic of one NN classifier used in our work. Although it is possible to invoke different classifiers for each genre of signal feature, we utilized the same classifiers here to facilitate the evaluation of local decisions. The use of different feature sets for each classifier ensures that the classifiers will perform independently (Xu et al., 1992).
Figure 2 is a block diagram of our proposed algorithm. First, the three *small* neural networks, classify their inputs independently. Then, using the outputs of these classifiers and their respective reputation values, the reputation-based algorithm determines the correct label of the input.
Classifier accuracy was estimated via a 10-fold cross validation with a 90-10 split. However, unlike classical cross-validation, we further segmented the 'training' set into an actual training set and a validation set. In other words, in each fold, 160 (80%) swallows were used for training, 20 (10%) for validation and 20 (10%) reserved for testing. Among the 20 swallows of the validation set, 10 were used as a traditional validation set and 10 were used for computation of the reputation values. After training, classifier reputations were estimated using this second validation set. Classifiers were then ranked according to their reputation values. Without loss of generality, assume $r_1 \geq r_2 \geq r_3$. If θ_1 and θ_2 cast the same vote about a test swallow, their common decision was accepted as the final classification. However, if they voted differently, the a posteriori probability of each class was computed using (14) and the maximum a posteriori probability rule was applied to select the final classification.
To better understand the difference between the multiple classifier system and a single, all encompassing classifier, we also trained a multilayer neural network via back-propagation with all 10 features, i.e., using the collective inputs of all three smaller classifiers. This all encompassing classifier, from hereon referred to as the *grand* classifier, also had 4 hidden units. We also statistically compared the accuracies of the individual classifiers against

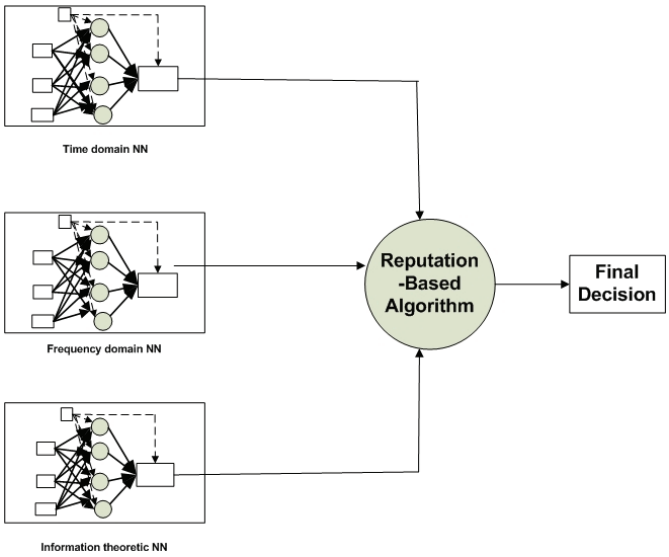

Fig. 2. The block diagram of the proposed algorithm

those of a majority vote classifier combination and a reputation-based classifier combination (Section 4.1).

To understand the knowledge representation of the individual small classifiers, we plotted Hinton diagrams for the input-to-hidden unit weight matrices (Section 4.2). Subsequently, we qualitatively compared the training performance of the small and grand classifiers to ascertain potential benefits of training a collection of small classifiers (Section 4.3). Through a systematic perturbation analysis, we quantified the local robustness of the reputation-based neural network combination (Section 4.4). In particular, we qualitatively examined the change in the classifier output and accuracy as the magnitude of the input perturbation increased. Finally, we estimated the breakdown point of the reputation-based neural network combination using increasing proportion of contaminants in the overall data set (Section 4.5).

4. Results and discussion

4.1 Classification accuracy

Table 1 tabulates the local and global classification results. On average, the frequency domain classifier appears best among the individual NNs while the information-theoretic NN fairs worst. Also, it is clear from this table that by combing the local decisions of the classifiers, using reputation based algorithm, the overall performance of the system increases dramatically. The result of the grand classifier is statistically the same as the small classifiers. However, training this classifier is more difficult and requires more time. Hence, there appears to be no justification of considering an all encompassing classifier in this application. Collectively, these results indicate that there is merit in combining neural network classifiers

Classifier	Average Performance (%)
Time domain NN	67.5 ± 12.5
Frequency domain NN	69.5 ± 10.3
Information theoretic NN	65.5 ± 11.2
Grand classifier	70.0 ± 8.5
Majority vote	74.5 ± 8.9
Combined classifier decision	78.0 ± 8.2

Table 1. The average performance of the individual classifiers and their reputation-based combination.

in this problem domain. The accuracy of the majority vote neural network combination did not significantly differ from that of the individual ($p > 0.11$) and grand classifiers ($p = 0.16$). On the other hand, the reputation-based combination led to further improvement in accuracy over the time domain ($p = 0.04$) and information-theoretic ($p = 0.05$) classifiers, but did not significantly surpass the grand ($p = 0.09$) and frequency domain networks ($p = 0.09$).
The reputation-based scheme yields accuracies better than those reported in (Lee et al., Under review) (74.7%). Moreover, in (Lee et al., Under review), the entire database was required and the maximum feature space dimension was 12. Here, we only considered a fraction of the database and no classifier had a feature space dimensionality greater than 4. Therefore, our system offers the advantages of computational efficiency and less stringent demands on training data.

4.2 Internal neural network representations

Figures 3, 4 and 5 are the Hinton graphs for the input to hidden layer weight matrices for the time, frequency, and information theoretic domain neural networks, respectively. In these figures, the weight matrix of each classifier is represented using a grid of squares. The area of each square represents the weight magnitude, while the shading reveals the sign. Shaded squares signify negative weights. The first column denotes the hidden unit biases while the subsequent columns are the weights on the connections projecting from each input unit. For instance, the frequency domain neural network uses 3 input features and 1 bias, resulting in 4 columns. Given that there are 4 hidden units, the weight matrix is represented as a 4×4 grid. In Figure 3, we see that the first neuron has a very large negative weight for kurtosis and a sizable one for variance. This suggests that this neuron represents swallows with low variance and platykurtic distributions. The second neuron seems to primarily represent swallows with leptokurtic distributions given its positive weight on the kurtosis input. By the same token, then third neuron appears to internally denote swallows with large positive means and leptokurtic distributions. Finally, the last neuron captures swallows primarily with high variance. Overall, the strongest weights are found on variance and kurtosis features, suggesting that they play the most important role in distinguishing between healthy and unhealthy swallows in our sample. Resonating with our findings here, Lee et al.

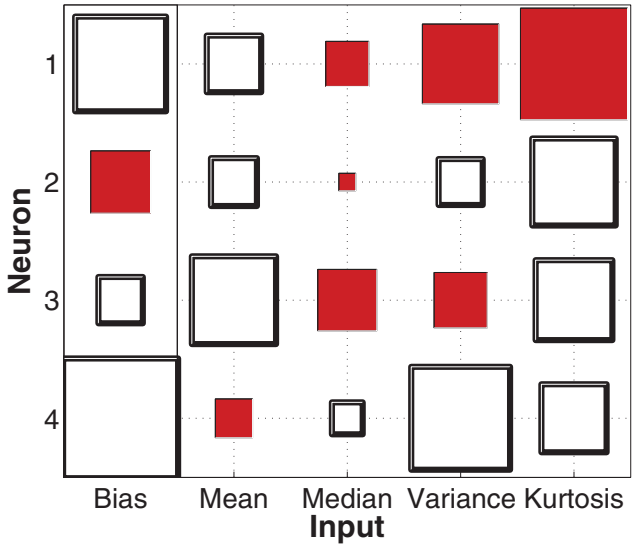

Fig. 3. The Hinton graph of the weight matrix for the time domain classifier

(2006) identified a dispersion-type measure as discriminatory between healthy swallows and swallows. Similarly, Lee et al. (2009) determined that the kurtosis of swallow accelerometry signals tended to be axial-specific and thus potentially discriminatory between different types of swallows.

Moving on to Figure 4, we notice that neurons one and two seem to have captured inverse dependencies of spectral centroid and bandwidth features. While neuron one embodies swallows with lower spectral centroid but broad bandwidth, neuron two captures swallows high frequency narrow band swallows. The peak FFT feature seems to be the least important spectral information, which is consoling in some senses as this suggests that decisions are not based upon signal strength, which may vary greatly across swallows regardless of swallowing health.

In the information theoretic neural network (Figure 5), we find that the memory feature seems to have a distributed representation across the 4 neurons, with three favoring weak memory or rapidly decaying autocorrelations. Neuron one almost uniformly considers all three information theoretic features, specifically epitomizing swallows with low complexity, low entropy rate and minimal memory. This characterization might reflect 'noisy' swallows. Interestingly, neuron three focuses on positive complexity and memory. We can interpret this neuron as representing swallows which have strong predictability and hence longer memory. In short, it appears that each individual neural network has internally represented some unique flavors of swallows. This apportioned representation across neural networks suggests that there is indeed sound rationale to combine classifiers, in order to comprehensively characterize the diversity of swallows.

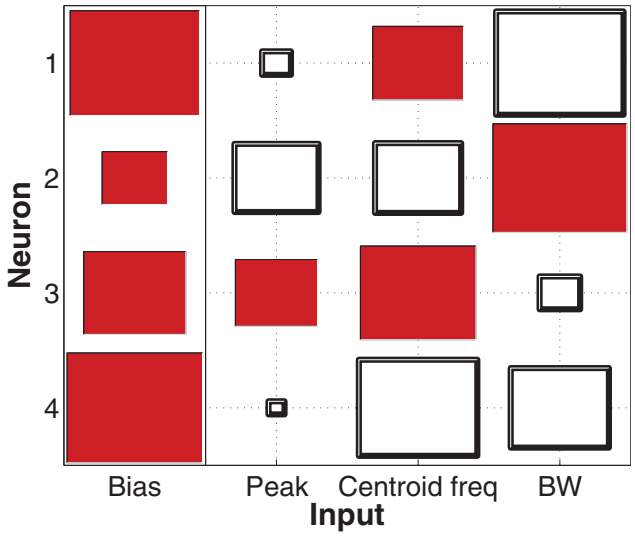

Fig. 4. The Hinton graph of the weight matrix for the frequency domain classifier (Peak - peak value of FFT; BW - bandwidth)

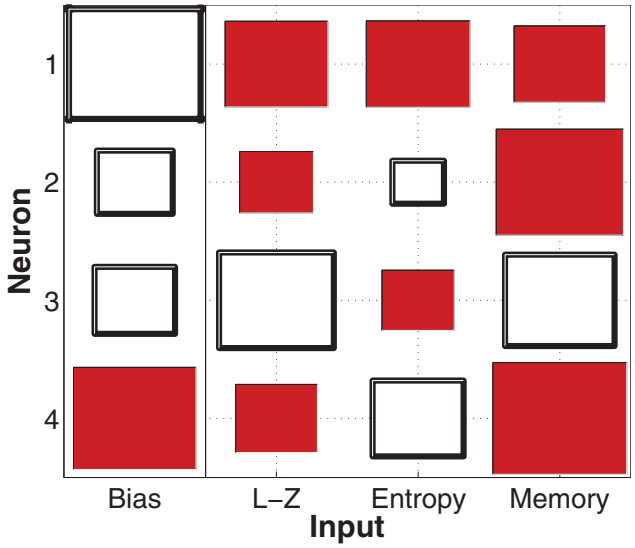

Fig. 5. The Hinton graph of the weight matrix for the information theoretic classifier.

4.3 Training error and convergence

Figure 6 pits the training performance of the small classifiers against the grand classifier as the number of training epochs increase. After 12 epochs, the small classifiers have lower

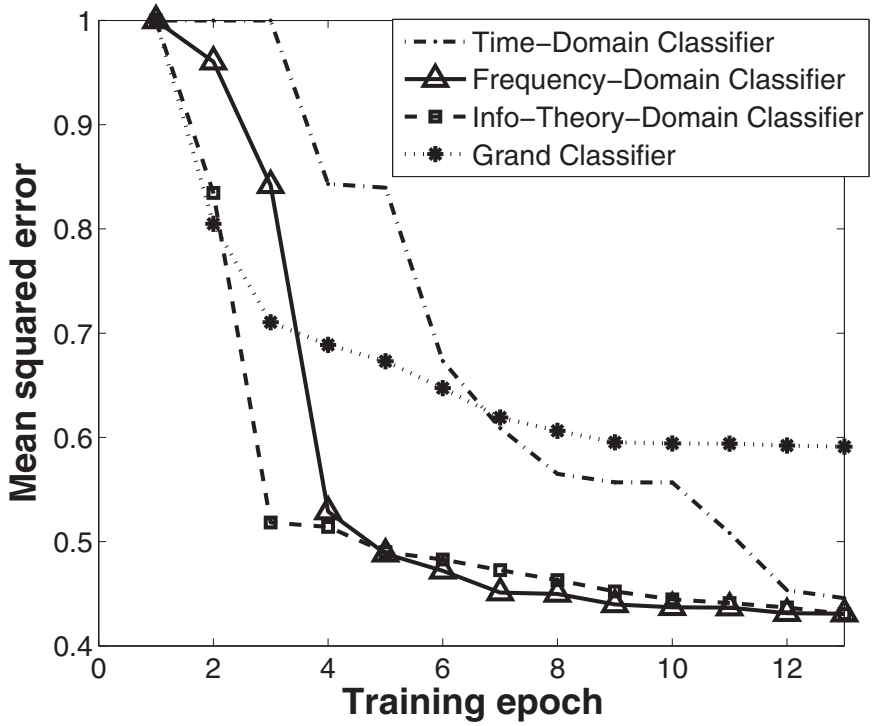

Fig. 6. The training error of the different neural network classifiers versus the number of training epochs

normalized mean squared errors than the grand classifier. This is one of the main advantages of using a multiple classifier system over a single all encompassing classifier; the rate of convergence during training is often faster with smaller classifiers, i.e., those with fewer input features, and in many cases lower training error can be achieved.

4.4 Local robustness

To gauge one aspect of the local robustness of the proposed neural network combination, we measured the sensitivity of the system to a local perturbation of the input. Recall that the reputation algorithm yields a class label rather than a continuous number. Thus, to facilitate sensitivity analysis, we calculated the reputation-weighted average of the outputs of the small classifiers for a specific input. For semantic convenience, we will just call this the reputation-weighted output. The unperturbed input sample was the mean vector of all the features in the test set. Perturbed inputs were created by adding varying degrees of positive and negative offsets to every feature of the mean vector. The sensitivity of the system to a given perturbation was defined as the difference between the reputation-weighted output for the unperturbed input and that for the perturbed input. At each iteration, the amount of perturbation was proportional to the range of the features in the test set. For instance, in the first iteration, 5% of the range of each feature in the test set was added to the respective

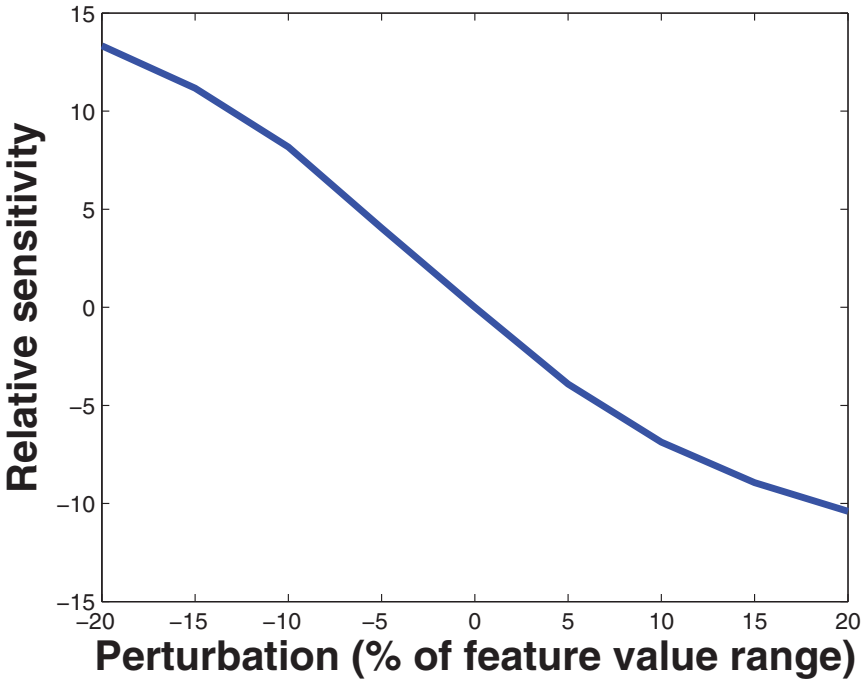

Fig. 7. Sensitivity to varying magnitudes of perturbation of the input vector

feature. Figure 7 shows the relative sensitivity of the system versus the magnitude of input perturbation. Between $\pm 10\%$, the relative sensitivity is less than 10% of the output value, suggesting that the reputation-based classifier while not robust in the strict statistical sense, can tolerate a modest level perturbation at the inputs.

To examine the effect of a local perturbation on the final decision of our algorithm, we again added/subtracted noise to the mean input vector and computed the output label using the reputation-based algorithm. For the present problem, the output was binary and without loss of generality, denoted arbitrarily as '1' or '2'. The unperturbed input belonged to class 1. As portrayed in Figure 8, the decision of the proposed algorithm is robust to negative perturbations up to 20% of the range of the features and positive perturbations up to 10% of the range of the features. However, for a positive perturbation higher than 10% of the range of the features, the reputation algorithm misclassifies the input. For practical purposes, this means that the reputation-based neural network combination can tolerate a simultaneous 10% perturbation in all its input features before making a wrong decision in the binary classification case. In the specific domain of dual-axis accelerometry, head movement induces high magnitude perturbations Sejdic, Steele & Chau (2010) which, according to our analysis here, will likely cause classification errors.

We also investigated the effect of local perturbations on the accuracy of the proposed algorithm. We perturbed all 20 samples in the test set. The amount of perturbation ranged from 0 to 100% of the range of the features in the test set. For each contamination value,

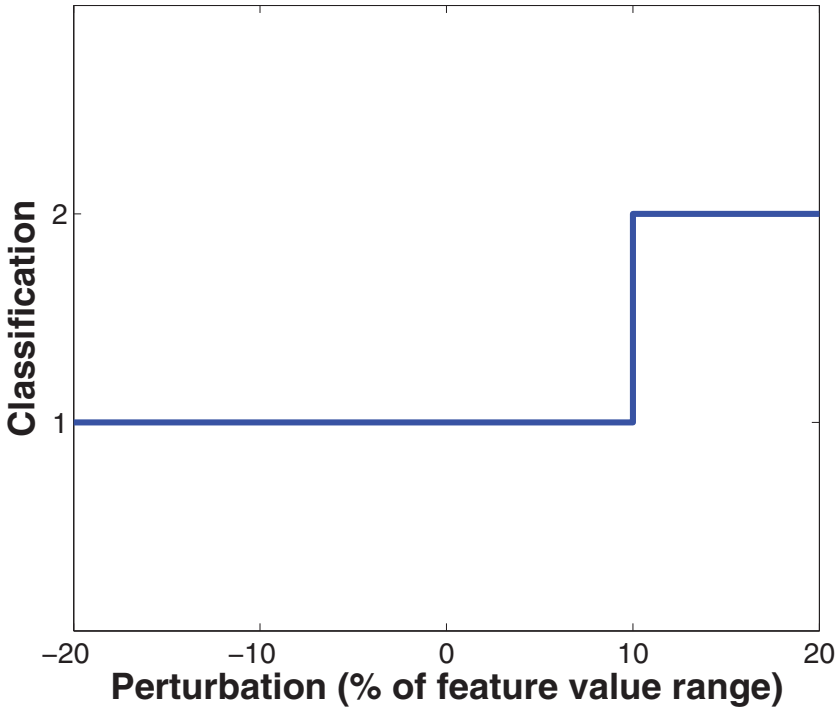

Fig. 8. The output decision versus magnitude of input perturbation

we calculated the accuracy of the proposed algorithm using the perturbed test set. Figure 9 illustrates the effect of varying levels of perturbation on the accuracy of the proposed algorithm. Based on this figure, the accuracy of the proposed algorithm decreased with increasing magnitude of perturbation in the test set. The initial accuracy of the proposed algorithm, for the unperturbed test set, was 78% and decreased to 50% for full-range (100% of the range of the features) perturbations. It is interesting to note that the decay in accuracy is quite steep for the first 20%, indicating that accuracy will take a hit with any non-zero amount of perturbation. Intuitively, this finding makes sense as the resemblance between test and training data diminishes as the magnitude of perturbation increases.

4.5 Global robustness

The sensitivity curve only offers local information about the robustness of the classifier. To measure the robustness of the system globally, we estimated the *breakdown point* for the proposed algorithm. For this matter, we substituted some feature vectors from among the 200 initial samples with contaminated versions. Contaminated feature vectors were created by sampling from a normal distribution with mean equal to that of the feature vector but with 3 times the standard deviation. The number of contaminants ranged from 20 to 100, i.e., 10 to 50% of the original data set. Using 10-fold cross validation, we divided the samples into 3 sets: training, testing, and validation. Therefore, it was possible that contaminations

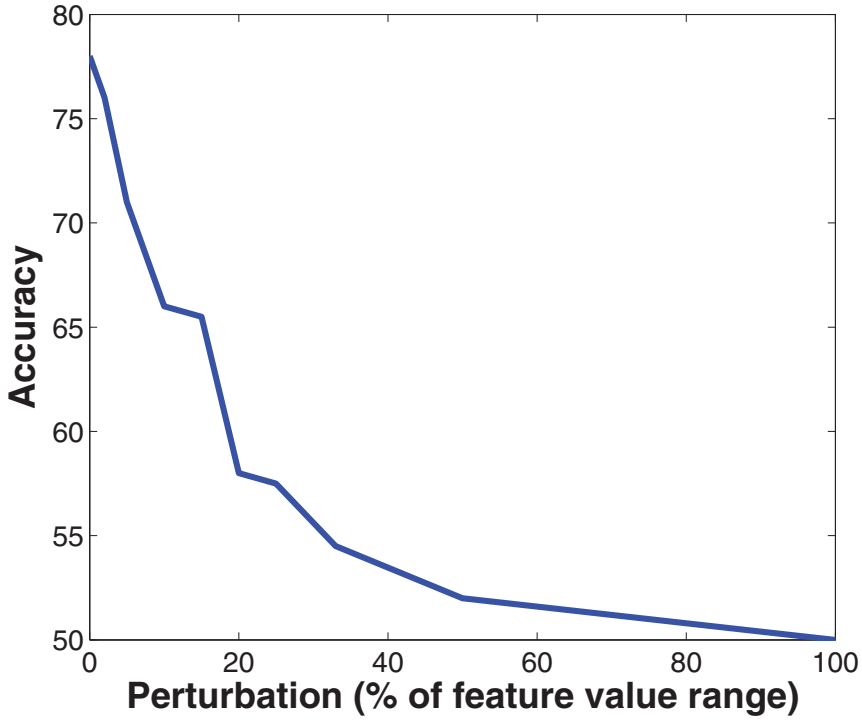

Fig. 9. The accuracy of the proposed classifier with increasing magnitude of perturbation in the test set

appeared in any or all of the training, testing, and validation sets. Figure 10 plots the accuracy of the proposed algorithm for different numbers of contaminated samples. Error-bars in this figure depict the standard deviation of each accuracy obtained from the cross-validation. To estimate the breakdown point for this system, we used the *rank sum test* to test for a significant difference between accuracies with and without varying levels of contamination. At a 5% significance level, we identified the first significant departure from the uncontaminated distribution of accuracy at 80 contaminated samples ($p = 0.043$). Given that there were 200 samples, the breakdown point was thus identified as $\frac{80}{200} = 0.4$.

5. Conclusion

We have presented the formulation of a reputation-based neural network combination. The method was demonstrated using a dysphagia dataset. We noted that generally the reputation-based classifier either achieved higher accuracies than single classifiers or exhibited comparable accuracies to the best single classifiers. Interpreting the weight matrices of the neural networks, we observed that many different aspects of time, frequency and information-theoretic characteristics of swallows were encoded. Finally, we empirically characterized the local and global robustness of the reputation-based classifier, showing that

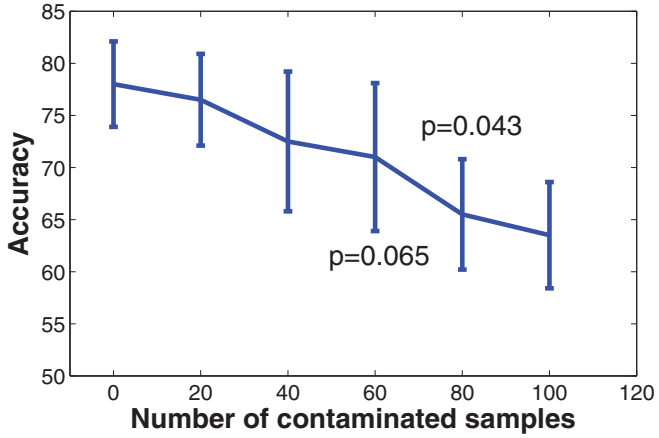

Fig. 10. The accuracy of the proposed classifier as the number of contaminated samples increase. The p-values arise from a comparison of the accuracy between the uncontaminated sample and samples with varying levels of contamination.

there exists a certain tolerance (approximately 10% of the range of a feature value) to input perturbations. However, large magnitude perturbations, such as those observed in head movement, would likely lead to erroneous classification of the swallowing accelerometry input feature vector.

6. References

Bauer, E. & Kohavi, R. (1999). An empirical comparison of voting classification algorithms: Bagging, boosting, and variants, *Machine learning* 36(1): 105–139.

Bloch, I. (2002). Information combination operators for data fusion: A comparative review with classification, *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on* 26(1): 52–67.

Breiman, L. (1996a). Bagging predictors, *Machine learning* 24(2): 123–140.

Breiman, L. (1996b). Bagging predictors, *Machine Learning* 24(2): 123–140.

Chen, H. & Yao, X. (2009). Regularized negative correlation learning for neural network ensembles, *Neural Networks, IEEE Transactions on* 20(12): 1962–1979.

Damouras, S., Sejdic, E., Steele, C. & Chau, T. (2010). An on-line swallow detection algorithm based on the quadratic variation of dual-axis accelerometry, *IEEE Transactions on Signal Processing* 58(6): 3352–3359.

Duda, R., Hart, P. & Stork, D. (2000). *Pattern Classification*, 2nd edn, Wiley-Interscience.

Haykin, S. (1994). *Neural Netowrks*, Macmillan College Publishing Company, New York.

Hull, J., Srihari, N., Cohen, E., Kuan, C., Cullen, P. & Palumbo, P. (1988). A blackboard-based approach to handwritten zip code recognition, *Procedings of the US Postal Service Advanced Technology Conference*, Washington, DC, pp. 1018–1032.

Jacob, R., Jordan, M., Nowlan, S. & Hinton, G. (1991). Adaptive mixtures of local experts, *Nueral Computation* 3(5): 79–87.

Jacobs, R., Jordan, M., Nowlan, S. & Hinton, G. (1991). Adaptive mixtures of local experts, *Neural computation* 3(1): 79–87.

Jain, A., Duin, R. & Mao, J. (2000). Statsitical pattern recognition: A review, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22(1): 4–37.

Kuncheva, L. (2002). A theoretical study on six classifier fusion strategies, *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 24(2): 281–286.

Lam, L. & Suen, C. (1997). Application of majority voting to pattern recognition: An analysis of its behavior and performance, *IEEE Transactions on Systems, Man, and Cybernetics* 27(5): 553–568.

Lee, D. & Srihari, S. (1993). Handwritten digit recognition: A comparison of algorithms, *3rd Intl. Workshop Frontiers Handwriting Recognition*, pp. 153–162.

Lee, J., Blain, S., Casas, M., Berall, G., Kenny, D. & Chau, T. (2006). A radial basis classifier for the automatic detetion of aspiration in children with dysphagia, *Journal of Neuroengineering and Rehabilitation* 3(14): 1–17.

Lee, J., Sejdic, E., Steele, C. & Chau, T. (2010). Effects of liquid stimuli on dual-axis swallowing accelerometry signals in a healthy population, *Biomedical Engineering OnLine* 9(7): 10 pp.

Lee, J., Steele, C. & Chau, T. (2008). Time and time-frequency characterization of dual-axis swallowing accelerometry signals, *Physiological Measurement* 29(9): 1105–1120.

Lee, J., Steele, C. & Chau, T. (2009). Swallow segmentation with artificial neural networks and multi-sensor fusion, *Medical Engineering & Physics* 31(9): 1049–1055.

Lee, J., Steele, C. & Chau, T. (Under review). Classification of healthy and abnormal swallows based on accelerometry and nasal airflow signals, *Artificial Intelligence in Medicine* xx(yy): zz.

Lempel, A. & Ziv, J. (1976). On the complexity of finite sequences, *IEEE Transactions on Information Theory* 22: 75–81.

Nadal, C., Legault, R. & Suen, C. (1990). Complementary algorithms for the recognition of totally unconstrained handwritten numerals, *10th Int. Conf. Pattern Recognition* A: 434–449.

Nikjoo, M., Steele, C. & Chau, T. (2011). Automatic discrimination between safe and unsafe swallowing using a reputation-based classifier, *IEEE Transactions on Biomedical Engineering* submitted(?): ?

Porta, A., Guzzetti, S., Montano, N., Furlan, R., Pagani, M., Malliani, A. & Cerutti, S. (2001). Entropy, entropy rate, and pattern classification as tools to typify complexity in short heart period variability series, *IEEE Transactions on Biomedical Engineering* 48(11): 1282–1291.

Schapire, R. (1990). The strength of weak learnability, *Machine Learning* 5(2): 197–227.

Sejdic, E., Komisar, V., Steele, C. & Chau, T. (2010). Baseline characteristics of dual-axis cervical accelerometry signals, *Annals of Biomedical Engineering* 38(3): 1048–1059.

Sejdic, E., Steele, C. & Chau, T. (2010). The effects of head movement on dual-axis cervical accelerometry signals, *BMC Research Notes* ?(?): in press.

Suen, C., Nadal, C., Mai, T., Legault, R. & Lam, L. (1990). Recognition of totally unconstrained handwritten numerals based on the concept of multiple experts, *in* C. Suen (ed.), *Proceedings of the International Workshop on Frontiers in Handwriting Recognition*, pp. 131–143.

Tresp, V. & Taniguchi, M. (1995). Combining estimators using non-constant weighting functions, *in* G. Tesauro, D. Touretzky & T. Leen (eds), *Advances in Neural Information Processing Systems*, Vol. 7, MIT Press, Cambridge, MA, pp. 418–435.

Ueda, N. (2002). Optimal linear combination of neural networks for improving classification performance, *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 22(2): 207–215.

Xu, L., Kryzak, A. & Suen, C. (1992). Methods of combining multiple classifiers and their applications to handwriting recognition, *IEEE Transactions on Systems, Man and Cybernetics* 22(3): 418–435.

Yao, X. & Islam, M. (2008). Evolving artificial neural network ensembles, *IEEE Computational Intelligence Magazine* 3(1): 31–42.