Decoding Trace Peak Behaviour – A Neuro-Fuzzy Approach

David Thornley, *Member, IEEE,* and Stavros Petridis

***Abstract*— DNA sequence basecalling is commonly regarded as a solved problem, despite significant error rates being reflected in inaccuracies in databases and genome annotations. This has made measures of confidence of basecalls important, and fuzzy methods have recently been used to approximate confidence by responding to data quality at the calling position. We have demonstrated that variation in contextual sequencing trace data peak heights actively encodes novel information which can be used for basecalling and confidence estimation. Using neuro-fuzzy classifiers we are able to decode much of the hidden contextual information in two fuzzy rules per base and partially reveal its underlying behaviour. Those two fuzzy rules can satisfactory explain over 74% of data samples. The error rate is 6-7% higher on individual bases than when using classification trees, but the number of rules is reduced by a factor of 100. Compact comprehensible knowledge representation is achieved with the use of SANFIS which allows us to easily interpret the embedded knowledge. Finally, we propose a hybrid architecture based on SANFIS which achieves slightly better performance than a classification tree with significantly improved knowledge representation.**

I. INTRODUCTION

After 30 years, the Sanger method remains the dominant DNA sequencing approach [1,2,3], but is subject to errors [4] and restricted read lengths. Up to 1% errors in the "high confidence" region of a trace are not uncommon. Miscalls or indels are commonly recovered in genomic sequencing by multiple coverage of the region of DNA being sequenced. Estimates of confidence in individual basecalls are important. In 1994, Lipshutz [17] proposed a successful classification-based method for confidence estimation using contextual data quality. More recently Varghese, Musavi and Ressom applied fuzzy methods to a different set of features to good effect [12]. These methods assess data quality.

A new means for reducing error rates or increasing read lengths will reduce the depth of coverage required in sequencing applications, and hence reduce resource consumption. A novel approach was proposed by Thornley [5] in 1997 in which the inherent sequence dependent peak height variation – rather than data quality – is exploited through abduction of the sample sequence composition against a model of trace data behaviour prediction. This uses a model of sequence-specific enzyme activity in the Sanger reaction, which enables prediction of signal variation. While development of a phenomenological model continues, we have used machine learning to demonstrate clearly that the variation in peak heights encodes information which can be used for basecalling [6]. We used neural networks and classification trees, singly and in ensembles, responding to measurements in the context of the basecalling position, *i.e.* calling bases without reference to the peak heights at the basecalling position itself. The result of that work was a success rate of 78%, which compares well with the 34% achievable using only organism specific base composition information (data for these experiments comes from the human X chromosome).

In this paper, we extend our previous work [6] to provide an efficient and compact machine learned interpretation of the information. The neural networks used previously are "black-boxes" models, not providing clear information about the underlying process. The classification (and regression) trees (CART) [7] performed less impressively than neural networks in this domain, but they have the advantage that rules can be easily extracted. The main drawback is that crisp trees result in hundreds of rules in this application. We currently believe that the reasons for this include the non-linearity of the system being modeled, and the existence of two main modes of behaviour which we explain later. The neural network, which uses non-linear relationships but crisp decisions, required 70 nodes to achieve a better error rate than approximately 200 nodes in the classification tree. In the present work we use a neuro-fuzzy network approach, SANFIS [8], to acquire a compact rule base for interpreting the embedded knowledge.

Neuro-fuzzy networks are multilayered connectionist networks that realize the component functions of fuzzy logic decision systems. We chose SANFIS over ANFIS [9], FALCON [10] or NEFCLASS [11] for our experiments due to its immunity from the curse of dimensionality, hence addressing our requirement for a smaller, comprehensible rule set to guide our research. A mapping-constrained agglomerative (MCA) clustering algorithm is used to initialize the network architecture, which results in a number of rules (clusters) similar to the number of classes. The present approach consists of two stages as shown in Fig. 1. In the first stage four SANFIS classifiers are used, each attempting to recognize a single class (True or False). If they do not call a base (all False or more than one True) then the second stage uses an alternative classifier, in this case a neural network. This yields accuracy comparable to the use of CART [6], but with the added advantage that we are able to extract two simple IF-THEN fuzzy rules for each class

D. Thornley was employed during this work on EPSRC Grant GR/S60266/01 in the Department of Computing, Imperial College London, UK. (e-mail: djt@doc.ic.ac.uk).

S.Petridis is with the Department of Computing, Imperial College London, UK. (e-mail: sp104@doc.ic.ac.uk).

(basecalls A, C, G and T) which describe the majority of cases. Thus the influence of the context for each base is largely encoded in only 2 fuzzy rules. We therefore have a combination of practical efficacy and valuable information about the system.

Experiments in over 600,000 samples show that 2 fuzzy rules explain (*i.e.* classify correctly) approximately 74%, 77%, 79% and 83% of the samples for bases T, A, C and G respectively. The full classifier achieves a success rate of 68.77%. We emphasize that in all experiments we only use contextual information as shown in Fig. 2, excluding the data at the basecall position which would be used in traditional basecalling. In this work, we are exploring in isolation the contextual information which is currently not utilized in basecallers.

The rest of the paper is organized as follows. In section II we briefly describe the SANFIS architecture we use. In section III we provide some background on DNA sequence analysis and describe our proposed method for efficient representation of contextual information by extraction of fuzzy rules. In section IV we describe the results obtained using our approach and finally section V summarizes our conclusions.

II. SANFIS

The recently introduced SANFIS architecture comprises five or six layers, each implementing an operation of the fuzzy inference system. In this work we restrict ourselves to a type III SANFIS (Takagi Sugeno model commonly referred to as TSK), and further define that the consequent linear functions cannot be shared by multiple rules. This leads to five layers as described below.

The output of each rule j is a linear combination of its inputs, with the following format:

$$\begin{aligned} &\textit{Rule } j: \textbf{IF } x_1 \textit{ is } A_1^{(j)} \textit{ and} \ldots \textit{and } x_n \textit{ is } A_n^{(j)} \\ &\textbf{THEN } y_1 \textit{ is } f_1^{(j)} \textit{ and} \ldots \textit{and } f_n^{(j)} \\ &\textit{where } f_k^{(j)} = b_{0k}^{(j)} + b_{1k}^{(j)} x_1 + \ldots + b_{nk}^{(j)} x_n \end{aligned}$$

Each node in the network consists of an input combination function f, and an output activation function α. The input function is expressed as:

$$node_{in} = f\left(u_1^{(l)}, u_2^{(l)}, \ldots, u_p^{(l)}; w_1^{(l)}, w_2^{(l)}, \ldots, w_p^{(l)}\right)$$

Where $u_1^{(l)}, \ldots u_p^{(l)}$ are the inputs to the node, $w_1^{(l)}, \ldots w_p^{(l)}$ are the associated weights and a superscript always indicates the layer number. The output activation function is expressed as:

$$node_{out} = \alpha^{(l)}\left(node_{in}\right) = \alpha^{(l)}(f) = u^{(l+1)}$$

Links which are not described by a particular function have weight 1. The functions of the nodes from layer 1 to 5 are defined as follows:

Layer 1: This is the *input* layer. The nodes in this layer convert the input values to a fuzzy type – a *fuzzy singleton* in this case - for the next layer. Thus, the functions of the *i*th node are defined as

$$f_i^{(1)} = u_i^{(1)} = x_i \textit{ and } \alpha_i^{(1)} = f_i^{(1)}$$

Layer 2: This *fuzzification* layer calculates the degrees of membership of each term in the fuzzy rule. The nodes represent Gaussian membership functions defined as:

$$f_{ij}^{(2)} = -\frac{1}{2}\left(\frac{u_i^{(2)} - m_i^{(j)}}{\sigma_i^{(j)}}\right)^2 \textit{ and } \alpha_{ij}^{(2)} = e^{f_{ij}^{(2)}}$$

where $m_i^{(j)}$ and $\sigma_i^{(j)}$ are the center and width of the Gaussian membership function of the j^{th} term of the i^{th} input respectively. The link weight, represented as $[m_i^{(j)}, \sigma_i^{(j)}]$, provides the Gaussian parameters. These are initialized by the MCA algorithm and optimized during the learning phase.

Layer 3: This layer implements the *antecedent* or IF part of the fuzzy rules and calculates the firing strength of each rule. Each node is equivalent to a rule (cluster) in the input space. The rule nodes perform a fuzzy AND operation, yielding input nodes and output functions of the j^{th} node as follows:

$$f_j^{(3)} = \prod_{i=1}^{n} u_i^{(3)} \textit{ and } \alpha_j^{(3)} = f_j^{(3)}$$

Layer 4: This is the *rule strength normalization* layer. The number of nodes in layers 3 and 4 are equal to the number of rules J.

$$f_j^{(4)} = \sum_{j=1}^{J} u_j^{(4)} \textit{ and } \alpha_j^{(4)} = \frac{u_j^{(4)}}{f_j^{(4)}}$$

Layer 5: This layer implements the *consequent* part of the fuzzy rules to form the output layer. It integrates the information from layer 5 with link weights [1 b_1 b_2 … b_n] and acts as a defuzzifier. The functions of the k th output node are

$$f_k^{(5)} = \sum_{j=1}^{J}\left(\sum_{i=1}^{n} b_{ki}^{(j)} x_i + b_{k0}^{(j)}\right) u_j^{(5)} \textit{ and } \alpha_k^{(5)} = f_k^{(5)}$$

The learning algorithm comprises two phases. In the first, we take the mean and variance of the clusters generated by the MCA clustering algorithm to provide the initial membership functions for the network. The main advantage of this MCA clustering algorithm is that the number of

clusters (rules) is similar to the number of classes and does not depend on the number of inputs, and hence does not suffer from the curse of dimensionality. This property results in a compact knowledge representation. In the second step a recursive linear/nonlinear least squares optimization algorithm is applied to tune the weights of the system to achieve a better performance. A detailed description of the architecture, clustering algorithm and the optimization algorithm can be found in [7].

III. DNA SEQUENCING – PROPOSED FRAMEWORK

The Sanger method produces four traces (see Fig. 2), one for each base, which are used for basecalling. Traditional basecallers perform pre-processing steps to simplify the data. This 'analysed' trace data enables a basecalling method based on finding the largest peak at each position which motivates confidence approximation using data quality. Later methods, such as PHRED [3,4] calibrate a model of the general layout of peaks along the trace to assess proposed signal peaks against expected positions.

No currently available method uses the information encoded in the contextual peak height variation identified by Thornley [13]. The presence of contextual information can be identified using a "blind-spot" analysis, *i.e.* calling a base excluding the peak heights at the calling position and only using information from the context, first introduced in [5]. Integrating this information into a full basecaller which can address the lower quality data later in the trace is a research question that we are currently addressing.

In [6], we performed a feature selection procedure which identified the peak height previous to the basecall and in the next three base positions, together with the spacing between the surrounding base positions as the most efficient features for blind calling a base (labelled in Fig. 2). We might assume that the surrounding basecalls should be included to complete the description, but these are implicit in the provision of peak heights in the correct lanes. Feature selection is sometimes considered outmoded given the availability of processing power, but when relating experiments to other modeling work, it forms part of the process of knowledge discovery.

In this framework we propose to use 19 features: 16 peak heights (f1-f16) surrounding the base to be called (since there are four peak heights in each base position) and 3 spacing values (f17-f19). The output is the class (base) in the blind spot position. We work with tuples constructed over 5 base positions, but information from only 4 of these is included, with the output base to be located in the second position as shown in Fig. 2.

The architecture we use is shown in Fig. 1. In the first stage there are 4 neuro-fuzzy classifiers which are trained to recognize a single class. The output of each classifier is True if the input pattern belongs to the class that the classifier is trained to recognize, or False otherwise. Unfortunately, it is common that two or more components vote for True or, in the vast majority of undecidable cases, all of them vote for False. To demonstrate the utility of the fuzzy basecaller in cooperation with other techniques, we use a neural network in stage two which is trained on the subset of the data which the stage one classifier does not produce a conclusive result.

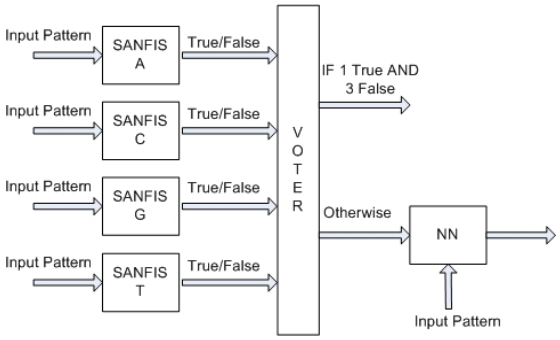

Fig. 1: Simple hybrid architecture. Initially the pattern is fed into the 4 SANFIS classifiers. Patterns that cannot be classified in stage 1 are passed to stage 2 where the neural network predicts a class for the input pattern.

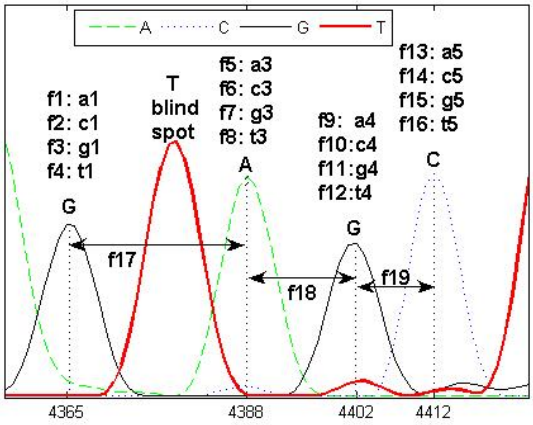

Fig. 2: Example of a 5-base window showing the 4 traces and 19 features

The main advantage of this approach is that the use of SANFIS allows us to extract two fuzzy rules for each class while maintaining use of context information in a valid, informative manner. Therefore the static influence of the context in each base A, C, G, T can generally be encoded in those two fuzzy rules. From other work we deduce that it will not be possible to encode contextual information in a strictly local, memoryless manner due to emergent dynamic behaviour [16]. Examination of the successes and failures of classification attempts will shed further light on these issues. Our analysis of the base contexts failing with the present SANFIS implementation show a bias toward homogeneous runs of bases which we propose lead to dynamic behaviour. Further work will look at iteratively identifying those data which are and are not susceptible to static classification.

Apart from extracting the embedded knowledge in the data for recognizing an individual base we go a step further and try to construct a classifier that recognizes all the bases. So we combine the four components as shown in Fig. 1 and when the data falls outside the scenarios addressable using the fuzzy classifier we use a neural network. In that way we are able to combine the interpretability of neuro-fuzzy systems and the detailed response of neural networks.

IV. RESULTS

A. Datasets

The data we use come from a genomic sequencing trace data set used in quality control testing at the Wellcome Trust Sanger Institute and are taken from a 182 kilobase contig in the human X chromosome. For our experiments we used processed output from ABI equipment.

Initially, we perform a skyline normalization [6], [14] of the data to remove the overall decay in peak heights, and the initial rise thought to be due to length dependent loading efficiency, by fitting a quadratic curve to the apexes of the peaks and dividing through by the value of the quadratic at each peak sample point.

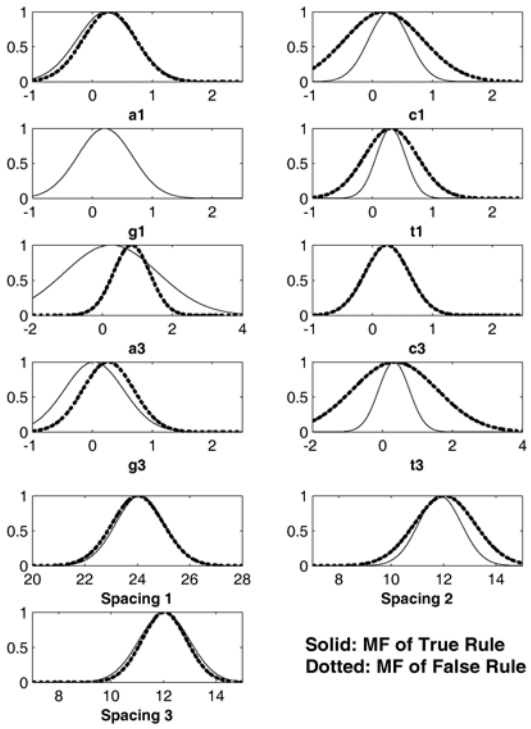

Fig. 3: Example membership functions for each feature of base G. Dotted MF belong to the False rule whereas solid MF belong to the True rule.

Since we have access to over 10000 files with sequencing traces and we consider overlapping 5-base windows in our experiments we have tens of millions of 5-base samples. Diminishing returns on training set size dictate that we use several subsets of data for training, validation and testing which in addition allow us to examine reproducibility. We use the same data as in [6], in which the training set consists of 62173 samples and the validation set contains 55782 samples. We also created ten independent test sets, to enable estimate the accuracy of the classifiers, which contain from 29757 to 113090 samples making a total of 603011 samples.

It is common in pattern classification applications to consider an equal number of training samples per output class when the data are uniformly distributed. However, as we showed in [6] the distribution of the 5-base windows is far from uniform, so each data set created contains at least two instances of all possible 5-base groups.

B. Single class neuro-fuzzy classifiers

Initially we trained four SANFIS classifiers to recognize a single class each. Each SANFIS was trained for 120 epochs. Two membership functions (MFs) are merged if their similarity measure calculated as in [9] is higher than 0.9. The first 16 features are the skyline normalized trace heights as described in section III and they only take positive values. The last 3 features are the peak spacing between surrounding peak heights and they also take positive values. We tested performance on the ten independent test sets as described in the previous section. The mean error rates together with the number of rules generated are shown in Table I.

TABLE I
MEAN % ERROR RATES OF RECOGNISING A SINGLE CLASS OVER 10 TEST SETS

	A	C	G	T
CART	15.84	15.45	10.05	18.31
(num rules)	(215)	(264)	(305)	(474)
SANFIS	23.19	20.74	16.62	25.88
(num rules)	(2)	(2)	(2)	(2)

We see an increase in the error rate from 5.3% (class C) to 7.57% (class T) compared to classification trees. However we notice a dramatic reduction in the number of rules generated. The minimum number of rules obtained by trees is 215 and in the worst case we have 474 rules which obscure the main advantage of trees in their interpretability. SANFIS creates only 2 fuzzy rules in all cases, which are capable of explaining over 74% of the samples.

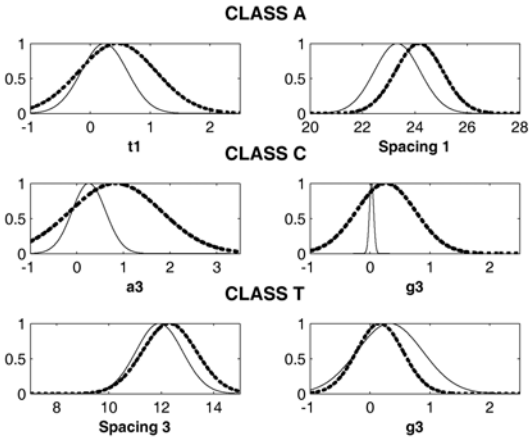

Fig. 4: Membership functions for each feature of base G. Dotted MF belong to the False rule whereas solid MF belong to the True rule.

This is a significant improvement in interpretability over the plethora of crisp rules generated by non-fuzzy systems. Such simplification is often desirable in bioinformatics applications where complexity must be peeled away methodically

The format of the two fuzzy rules is as follows:

$R1$: **IF** f_1 *is* S_1 *and*…*and* f_{19} *is* S_{19} **THEN** y_1 *is* f_1 (*TRUE*)

$R2$: **IF** f_1 *is* D_1 *and*…*and* f_{19} *is* D_{19} **THEN** y_1 *is* f_2 (*FALSE*)

where f_1, f_2 are linear combinations of the inputs since we only use TSK models and S, D are the solid and dotted MF respectively. Fig. 3 shows examples from the membership functions MFs for inputs (f1-f19) from class G. Solid MFs belong to rule 1 and dotted to rule 2.

TABLE II
FEATURES

	Most informative features	Least informative features (Merged MF)
Class A	a1, t1, g3, Spacing1	t3, c4, g4, t4,Spacing2
Class C	a3, c3, t3, g3,	g1, t1, a4, g4, a5, c5, g5, t5, Spacing 3
Class G	c1, t1, a3, g3, t3, Spacing2	g1, a4, c5, t5,
Class T	g3, t3, t4, Spacing3	a3

This indicates that the contribution of those features is the same for both rules, they achieve the same degree of activation, so they are not that important for discrimination. On the other hand, the MFs of some other features are quite different and they are useful for discrimination since they activate both rules to a different degree, for example a3, t3. Table II shows the most informative features together with the least informative features for each class. We see that the first position to the right of the blind spot, *i.e.* the third position in the 5-mer, seems the most important since some of its features are present in all cases. Particularly, g3 is present in all classes and t3 is present in three classes. On the other hand features from the fifth position do not seem to have significant contribution in the inference process and their MFs are often merged as shown in the second column of Table II. These observations agree with our previous work, but are made clearer by the simpler form of the rules.

Fig. 4 shows two characteristic MFs from the other classes A, C, T which are not presented for lack of space.

C. Full classifier

In an attempt to create a classifier that is capable of directly predicting a base in the blind spot given only contextual information and not just giving a True/False answer we use the architecture of Fig. 1. As explained in section III ideally we would like only one True and three False. To deal with scenarios outside the remit of direct classification, we use a neural network similar to that in [6] as a second stage in the classifier. Initially, we tested the four SANFIS in the validation set of section IV.A and then the patterns that were not able to classified (led to contradictory True/False classifications) were gathered and used to train the neural network. We used a feed-forward network with 70 neurons. The resilient backpropagation training algorithm was used for its speed of operation [15] and the network was trained for 4000 epochs and the learning rate was set to 0.005.

We tested the full hybrid classifier again over the ten independent test sets and the results are presented in Table III. We compare the results to a classification trees and neural networks trained using the whole training set which predict the output class directly from [6]. We also quote the best results obtained using an ensemble of ten neural networks with averaging as the combination rule.

TABLE III
VARIATION IN ERROR RATES

	CART	NN	NN Ensemble	SANFIS + NN
Test1	31.01%	23.52%	21.10%	30.55%
Test2	30.69%	23.92%	21.14%	30.38%
Test3	30.14%	22.27%	20.80%	30.40%
Test4	31.41%	23.79%	21.20%	30.20%
Test5	31.99%	24.52%	22.16%	30.60%
Test6	30.30%	22.77%	19.89%	29.26%
Test7	31.63%	24.13%	21.84%	31.47%
Test8	34.99%	27.91%	26.18%	34.65%
Test9	33.07%	25.54%	23.50%	32.92%
Test10	31.92%	24.66%	22.19%	31.87%
Mean	31.72%	24.30%	21.998%	31.23%
Variance	2.09	2.47	3.10	2.48
N° Rules	853	-	-	8

A single neural network and the ensemble of neural networks clearly out-perform the proposed architecture, but they do not provide any insight into the underlying mechanism. The proposed 2-stage classifier has better performance than CART, and it offers eight fuzzy rules that explain satisfactory a significant portion of the samples. Of course, they are not as general as the CART rules since they are applied to each base individually and output only True/False whereas a CART rule outputs directly one of the four classes. However, while the overall performance is comparable, the gain in interpretability is significant. It is tractable for humans to understand 8 fuzzy rules, whereas 853 crisp rules would require further processing to be of use. Compact comprehensible knowledge representation is an important advantage of SANFIS, and the reduction of rules by a factor of 100 here confirms this.

We also tried a hybrid classifier with stage one comprising tree classifiers, resulting in an error rate of around 30%, which is slightly better than SANFIS performance but in this case we have 1269 rules (see Table I), which further reduces the interpretability of the system.

Table IV presents the error rates in the two stages of the classifiers. The first column presents the number of samples per test set. The second column shows the percentage of samples that were classified by the combination of the four SANFIS in the first stage. The remaining samples are passed on stage two. The third column shows the error rate of the combination of the four SANFIS classifiers. Here we only consider the patterns that are classified in stage one and

therefore are not passed on stage two. Finally, the last column gives the error rate of the neural network on the patterns that are not classified in the first stage and are fed in the network.

TABLE IV
DECOMPOSITION OF HYBRID ERROR RATE

	No Samples	Classified in Stage 1	Stage 1 Error Rate	NN Error Rate
Test1	74629	44.79%	34.88%	27.03%
Test2	54859	44.43%	33.56%	27.83%
Test3	59926	46.54%	33.11%	28.04%
Test4	75056	44.82%	34.78%	26.47%
Test5	30589	43.35%	34.67%	27.50%
Test6	34572	44.21%	34.60%	25.04%
Test7	113090	45.35%	34.10%	29.29%
Test8	29757	45.31%	34.52%	34.76%
Test9	88492	44.54%	34.83%	31.39%
Test10	42041	44.05%	35.05%	29.36%
Mean	-	44.74%	34.41%	28.67%
Variance	-	0.76	0.40	7.62

We see that just under 45% of the patterns can be classified by the four SANFIS with an accuracy of 65.59% (*c.f.* 25% random, or 34% with organism bias). The remaining 55% percent is classified by the neural network which achieves an accuracy of 71.33%. The accuracy of the neural network is lower than the one shown in Table III since it only classifies samples which are rejected in the first stage and therefore a significant proportion of "difficult" patterns is included.

Finally, we note the variance of the SANFIS classifiers: their performance is remarkably consistent between different test sets (the majority of the variance in the error rate of the hybrid classifier is due to the neural network in stage two). This confirms that the compact knowledge representation consistently captures the behaviour of the system.

V. Conclusions

DNA sequence basecalling is at the heart of modern genomics, which is already contributing to healthcare innovation. This work with fuzzy classifiers is part of our ongoing investigation of the use of contextual information to enhance basecalling.

We have demonstrated that with the use of SANFIS we are able to achieve the same level of performance as CART, but with a dramatically more compact rule base for a significant subset of the data. This allows us to interpret the embedded knowledge in data and explain static effects of contextual information in DNA sequencing trace data.

Our further work will draw from modern, emerging concepts in fuzzy systems directed to achieving a performance level comparable to neural networks while maintaining interpretability. This includes the evolution of constructed features to capture the relationships in the Sanger reaction. The present results also motivate investigation of fuzzy classification trees to build directly on [6] with the expectation of enhancing accuracy over our previous results while reducing the rule count.

As well as providing information for a basecalling application, the fuzzy classifier finds compact information about the contributing features which will be used to guide development of a phenomenological model of the underlying process to be used in an optimal abduction-based caller [13]. Fuzzy reasoning will therefore play an essential part in the development and implementation of a basecaller which uses sequence-dependent context information.

References

[1] F. Sanger, S. Nicklen and A.R. Coulson. DNA sequencing with chain terminator inhibitors. *Proc. Natl. Acad. Sci*. 74, 5463 5467, 1977d ed. vol. 3, pp. 15–64 J. Peters, Ed. New York: McGraw-Hill, 1964,.

[2] C. Connel, S. Fung, C. Heiner, J. Bridgham, V. Chakerian, E. Heron, B. Jones, S. Menchen, W. Mordan, M. Raff, M. Recknor, L. Smith, J. Springer, S. Woo and M. Hunkapiller. (1987) Automated DNA Sequence Analysis *BioTechniques* **5**, 342-348.

[3] Ewing B, Hillier L, Wendl MC, Green P, "Basecalling of automated sequencer traces using phred. I. Accuracy assessment", GENOME RESEARCH 8 (3): 175-185 MAR 1998

[4] Ewing B, Green P, "Basecalling of automated sequencer traces using phred. II. Error probabilities", GENOME RESEARCH 8 (3): 186-194 MAR 1998

[5] D. J. Thornley. "Analysis of trace data from fluorescence based Sanger sequencing". PhD thesis, University of London, Imperial College of Science, Technology and Medicine, Department of Computing, 1997.

[6] D. Thornley, S. Petridis, "Machine Learning in Basecalling – Decoding Trace Peak Behaviour" *Proc. IEEE CIBCB, Toronto, 2006*

[7] L. Breiman, J. H. Friedman, R.A. Olshen and C. J. Stone. "Classification and regression trees", Wadsworth, Inc., Belmont, California, 1984.

[8] J. S. Wang, C. S. G. Lee, "Self-Adaptive Neuro-Fuzzy Inference Systems for Classification Applications", *IEEE Trans. Fuzzy Systems*, Vol.10, No 6, pp.790-802, Dec 2002

[9] J. S. R. Jang, "ANFIS: Adaptive-network-based fuzzy inference system", *IEEE Trans. Syst. Man. Cyber*., vol.23, pp.665–685. June 1993

[10] C.T. Lin, C.S.G. Lee, *Neural Fuzzy Systems:A neuro-Fuzzy Synergism to Intelligent Systems*. Upper Saddle River, NJ: Prentice Hall, 1996

[11] D. Nauck, U. Nauck. R. Kruse, "A Neuro-Fuzzy Approach for the Classification of Data", *Proc. 1995ACM Symposium on Applied Computing*, p.461-465, ACM Press, New York, February 1995

[12] R.S.Varghese, M.T. Musavi and H. Ressom. A Fuzzy Confidence Value for DNA Bases, FUZZ-IEEE 2004, Budapest

[13] International Patent Application WO96/20286 July 4, 1996, European Patent EP0799320 Mar. 7 2001 and US Patent 6,090,550, Jul. 18, 2000

[14] Lucio Andrade, Elias S. Manolakos. Skyline Normalization of DNA Chromatograms by Regression, *in Workshop On Genomic Signal Processing and Statistics (GENSIPS)*, 2002, pp.CP2—7:1-4

[15] Riedmiller, M., and H. Braun, "A direct adaptive method for faster backpropagation learning: The RPROP algorithm", Proceedings of the IEEE International Conference on Neural Networks, 1993

[16] D.J.Thornley. Modeling along the DNA template in the Sanger method: inhibition through competition and form, Process Algebra and Stochastically Timed Activities, June 2006

[17] Robert J. Lipschutz, Fred Taverner, Kevin Hennesy, George Hartzell and Ron Davis. DNA sequence confidence estimation. *Genomics* 19, 417 – 424 1994