# A Familiarity Discrimination Algorithm Inspired by Computations of the Perirhinal Cortex

Rafal Bogacz[1], Malcolm W. Brown[2], and Christophe Giraud-Carrier[1]

[1] Dept. of Computer Science, University of Bristol, Bristol BS8 1UB, UK
{bogacz,cgc}@cs.bris.ac.uk
[2] Dept. of Anatomy, University of Bristol, Bristol BS8 1TD, UK
M.W.Brown@bristol.ac.uk

**Abstract.** Familiarity discrimination, i.e. the ability to recognise previously experienced objects is important to the survival of animals, but it may also find practical applications in information technology. This paper describes the *Fam*iliarity discrimination based on *E*nergy algorithm (FamE) inspired by the computations of the perirhinal cortex - the area of the brain involved in familiarity discrimination. In FamE the information about occurrences of familiar records is encoded in the weights of a neural network. Using the network, FamE can discriminate whether a given record belongs to the set of familiar ones, but cannot retrieve the record. With this restriction, the network achieves much higher storage capacity for familiarity discrimination than other neural networks achieve for recall. Therefore, for a given number of familiar records, the description of the weights of the network occupies much less space in memory than the database containing the records itself. Furthermore, FamE can still classify a record as familiar even if it differs in a substantial proportion of its bits from its previous representation. FamE is also very fast. Preliminary results of simulations demonstrate that the algorithm may be applied to real-world problems.

## 1 Introduction

Animal and humans possess the ability to discriminate familiarity for an impressive number objects. Human subjects, after seeing thousands of different pictures once, can still recognise the individual pictures as familiar [21]. This ability to recognise previously seen objects is important to the survival of animals, but it may also find practical applications in information technology. For example, the ability to determine whether a web site has been visited or not is very useful to a software agent searching the web. Familiarity discrimination is analogous to checking whether a record belongs to a certain set (i.e., the set of familiar records). From this point of view there exists a wide range of possible applications. One example could be a program controlling a security camera at the entrance of a building - it does not need to identify a person (e.g., retrieve its name), but only discriminate whether this person belongs to the set of persons authorised to enter the building. Another example could be checking whether a record is stored in a database without actually searching the database.

S. Wermter et al. (Eds.): Emergent Neural Computational Architectures, LNAI 2036, pp. 428–441, 2001.

To solve problems related to familiarity discrimination, current methods create a database of familiar records and use it to check whether a given record is contained in the database. Storing the database allows retrieval of information so this approach is not specialised for familiarity discrimination and hence is far from optimal. It is used because nowadays disk space is cheap and database searching is relatively fast. However, in the case when a user does not look for an exact match in the database but for something similar, database searching is much slower because one cannot use standard searching techniques such as indexing or hashing.

This paper describes the recently developed algorithm, *Fami*liarity discrimination based on *E*nergy (FamE) [5] and shows its applicability. In FamE the information about occurrences of familiar records is encoded in the weights of a neural network. Using the network, FamE can discriminate whether a given record belongs to the set of familiar ones, although it cannot retrieve the record. For example, it cannot retrieve the record after being given only partial information (e.g., a key). With this restriction, the network achieves much higher storage capacity for familiarity discrimination than other neural networks achieve for recall. Therefore, for a given number of familiar records, the description of the weights of the network occupies much less space in memory than the database containing the records itself. Furthermore, FamE can still classify a record as familiar even if it differs in a substantial proportion of its bits from its previous representation. FamE is also very fast.

FamE is inspired by the computations performed by the perirhinal cortex. Work in amnesic patients and in animals has established that discrimination of the relative familiarity or novelty of visual stimuli is dependent on the perirhinal cortex [1,2,8,17]. Damage to the perirhinal cortex results in impairments in recognition memory tasks that rely on discrimination of the relative familiarity of objects [16]. Within the monkey's perirhinal cortex, ∼25% of neurons respond strongly to the sight of novel objects but respond only weakly or briefly when these objects are seen again [8,22]. We have created a model of the perirhinal cortex which is consistent with many experimental observations [6]. Since the perirhinal network has the properties mentioned in the previous paragraph for FamE, the perirhinal cortex alone may rapidly discriminate the familiarity of many more stimuli than current neural network models indicate could be recalled (recollected) by all the remaining areas of the cerebral cortex. This efficiency and speed of detecting novelty provides an evolutionary advantage, thereby giving a reason for the existence of a familiarity discrimination network in addition to networks used for recollection.

FamE differs in important respects from the artificial neural networks used for familiarity discrimination in industrial applications [19,10]. In these approaches familiarity discrimination is regarded as detecting typical patterns of device behaviour, since atypical (i.e., novel) patterns may be a sign of malfunction. Hence such models assume that familiar patterns create clusters in representation space (and the synaptic weights of their neurons often encode prototypes of familiar patterns, e.g., see [10]). In contrast, the model outlined here does not require any

assumptions concerning the distribution of patterns and it discriminates whether a particular pattern was presented previously rather than whether the pattern is typical. The information processing of FamE is somewhat similar to that in a novelty detector [14,15], but the novelty detector is an abstract model of a single neuron, with correspondingly limited storage capacity. The proposed model is a network of neurons constructed as to have a very large storage capacity.

The description of the algorithm is given in Section 2. In section 3 the storage capacity for familiarity discrimination is investigated. Section 4 discusses implementation issues which must be considered before the application of FamE. Section 5 shows how the algorithm performs on real data. Finally, section 6 discusses the relation of FamE to other techniques.

## 2 Algorithm Description

FamE stores information about the familiar patterns in the weights of a Hopfield network. The Hopfield network provides a simple model of associative memory [12]. It is a fully connected recurrent neural net consisting of $N$ neurons, whose activations are denoted by $x_i$. The active state of a neuron is represented by 1, and the inactive state by -1. The patterns stored by the network are denoted by $\xi^\mu$ and the number of these patterns by $P$. The weight of the connection between neurons $j$ and $i$ is denoted by $w_{ij}$ and computed according to Hebb rule [12]:

$$w_{ij} = \begin{cases} \frac{1}{N}\sum_{\mu=1}^{P} \xi_i^\mu \xi_j^\mu & \text{if } i \neq j \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

The energy of the Hopfield network is defined by [12]:

$$E(x) = -\frac{1}{2}\sum_{i=1}^{N} x_i \sum_{j=1}^{N} x_j w_{ij} \tag{2}$$

The value of the energy function is usually lower for stored patterns and higher for other patterns [3]. Therefore, the value of the energy may be used for familiarity discrimination, which in this context corresponds to checking whether a pattern is stored in the Hopfield network [5]. Normally, the Hopfield network is used for retrieval of information by updating one-by-one the activities of the neurons (a process called relaxation). In FamE, the neurons do not perform any computations (i.e., there is no relaxation), but familiarity discrimination is achieved by checking the value of the energy function after delivery of a pattern. In other words, the discrimination is done not by the Hopfield network itself, but by an external entity which sets up the activations of the neurons according to a discriminated pattern and calculates the network's energy for this pattern.

We have already showed that the average value of the energy for stored patterns is $-N/2$, while for novel patterns it is 0 [5]. Therefore, by taking as a threshold the middle value of $-N/4$, we can define a familiarity discrimination

criterion, namely, if $E < -N/4$, then the pattern is classified as familiar, and as novel otherwise.

In [6] we showed that the neural network designed to mimic neuronal activity in the perirhinal cortex performs similar computations during familiarity discrimination. The energy of the Hopfield network is an artificial function whose value is calculated by a double summation (see Equation 2). The model perirhinal network effectively calculates a similar function also by a double summation implemented by two layers of neurons - the first layer performs the first summation and the second layer the second summation. For details see [6].

## 3 Storage Capacity

Using signal-to-noise analysis we have established that the FamE algorithm using a network of $N$ neurons can discriminate familiarity with 99% reliability for $0.023N^2$ uncorrelated patterns [5]. This capacity is much greater than the standard capacity of the Hopfield network for retrieval, namely $0.145N$ [3]. If a higher reliability is required, the familiarity discrimination capacity decreases slightly but it is still of order $N^2$, e.g., for an error probability of $10^{-4}$ the capacity is about $0.009N^2$ and for an error probability of $10^{-6}$, about $0.006N^2$.

If the human perirhinal cortical network operates on similar principles, its theoretical capacity may be estimated on the assumption that it contains $\sim 10^7$ pyramidal neurons [13], 25% of which discriminate familiarity, each with $\sim 10^4$ synapses. With a probability of error of $10^{-6}$, the perirhinal network could store $\sim 10^8$ patterns, each with up to $0.25 \times 10^7$ bits. A pile of books storing these patterns would be $\sim$7000 km high (equal to the Earth's radius). The speed of searching this database is also impressive. It would take 20 ms for light to traverse the pile, while discriminating familiarity in the model of the perirhinal cortex takes only $\sim$10 ms.

FamE demonstrates generalisation and is resistant to disruption by noise - a pattern will still be classified as familiar even if it differs in a substantial proportion of its bits from its previous representation. More precisely, a pattern will be classified as familiar if the Hamming distance (number of different bits) between the pattern and one of the stored patterns is small. Therefore, before applying the methods one should ensure that data is represented in such a way that similar pieces of information (from the point of view of the user) have similar representations in Hamming space.

The above considerations concerning capacity assume that the weights of the network are represented as real numbers. When in the Hopfield network the weights are replaced by binary values, i.e., positive weights by 1 and negative weights by -1, then the capacity drops from $0.145N$ to about $0.1N$, i.e., it decreases by a factor of 0.69 [11]. A similar decrease is observed in the case of FamE. For example, for a probability of error of 1%, the capacity decreases from $0.023N^2$ to $0.016N^2$. After the conversion of the weights to binary values, the average value of the energy for stored patterns is not $-N/2$ anymore. Hence, the energy discrimination threshold (below which patterns are classified as familiar)

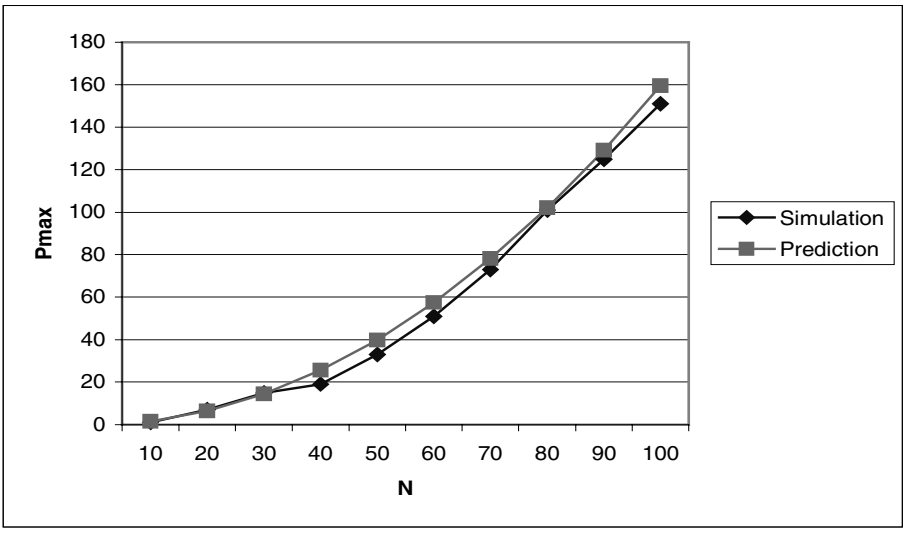


**Fig. 1.** Comparison of the simulated familiarity discrimination capacity of the network with binary weights, with theoretical predictions

cannot be taken as $-N/4$. The threshold should be found empirically. First, one should compute what the average energy value is for patterns stored in the network (by delivering stored patterns one by one to the network and calculating the energy without modifying the weights). Then, one should compute the average energy for a number of patterns which are not stored in the network: the discrimination threshold should be taken as the mean of these two averages.

Figure 1 compares the simulated familiarity discrimination capacity of the network with binary weights, with theoretical predictions. For each number of inputs $N$, and for each number of stored patterns $P$, the behaviour of the network was tested on 1000 random patterns. Among these patterns, 500 were stored patterns and 500 were random patterns for which the absolute value of the correlation with each stored pattern was less than 0.5. For each number of inputs $N$, $P_{max}$ is taken as the maximum number of stored patterns for which the error rate is $\leq 1\%$.

In the case of traditional databases, a database storing $K$ records of length $N$ bits occupies $NK$ bits of memory (e.g., on the disk). The number of bits occupied on average by one record may be defined as the memory occupied by the database divided by the number of records, that is, $NK/K = N$. In the case of a network with $N$ neurons and binary weights, the database occupies $N^2/2$ bits, because each weight is represented by one bit and the weights are symmetrical, so only half of them need be stored. Assuming a probability of error of 1%, the number of records for which FamE may discriminate familiarity is $0.023N^2 \times 0.69$. Hence, the number of bits occupied on average by one record

is equal to $(N^2/2)/(0.023N^2 \times 0.69) = 31.51$ bits (i.e., 3.94 bytes). In contrast to traditional databases, here the space occupied by a record is constant and does not depend on the number of bits in the record. For a probability of error of $10^{-4}$ the space occupied by a record is 80.52 bits (10.6 bytes) and for a probability of error of $10^{-6}$ the space is 131.75 bits (16.47 bytes). These data show that FamE is especially useful for databases containing large records. For example, for a database with records of 1.5KB, the weights of the network allowing FamE to discriminate familiarity with a probability of error of $10^{-6}$, would occupy only about 1% of the space taken by the database with the records.

## 4 Implementation

Section 2 describes how the FamE algorithm works. However, before using it for practical applications, some implementation issues need to be considered.

Every Hopfield network has limited capacity for familiarity discrimination. The size of the network required by a particular application (i.e., the number of neurons) is determined by the number of bits in the records. The capacity of the network is determined by its size and required probability of error. Hence, the network capacity may differ from the capacity required by the application (e.g., the number of records in the database). If the capacity required is smaller, then one can use a network with sparse connections - we have showed that if connections are removed from the network, the storage capacity decreases in proportion to the number of connections removed [6]. If the capacity required is larger than can be achieved using a single network, then one can use a number of Hopfield networks and discriminate familiarity by checking the energy of every network one-by-one.

There are two types of errors the FamE algorithm may make: to classify a novel pattern as familiar (false-recognition) and to classify a familiar pattern as novel (non-recognition). Although, as we showed in [5] false-recognition errors cannot be avoided, non-recognition errors can be eliminated. There are two methods of eliminating or reducing non-recognition errors. First, in the case where many networks are used (to accommodate a database larger than the capacity of a single network), the patterns written to the first network and not recognised by it may be included in the set of patterns being written to the second network, and so on. Second, the number of non-recognition errors may also be reduced by using a different algorithm to modify weights. Instead of presenting all the patterns to the network and modifying the weights only once for each pattern according to Equation 1, one can present patterns in a number of epochs (i.e., show each pattern once and modify the weights, then show the patterns again and so on). One should start with all the weights initialised to zero and for each pattern modify the weights by an amount $\delta$ depending on how confident the network is of the novelty of the pattern:

$$\Delta w_{ij} = \frac{1}{N}\delta x_i x_j, \text{ where } \delta = \begin{cases} \dfrac{N/2 + E(x)}{N/2} & \text{for } E(x) > -N/2 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

The coefficient $\delta$ is large for patterns having high energy (and thus classified as novel). Hence for such patterns the magnitude of weight modification is high. On the other hand, $\delta$ is small for patterns having low energy, so for these patterns the weights are not changed significantly. Equation 3 is a generalisation of Equation 1, since when $\delta = 1$, they are equivalent. We have showed in [7] that this algorithm always finds values of weights resulting in the correct classification of stored patterns as familiar if such values exist. In practice this means that in many cases it eliminates non-recognition errors completely. In [7] we also proposed a biological mechanism which may implement a similar mechanism in the perirhinal cortex.

The results presented in section 3 concerned the case where patterns are not correlated. The condition of uncorrelation is likely to be satisfied by the patterns of neuronal activity for which the network of the perirhinal cortex discriminates familiarity because, as suggested in [4], the activity of sensory neurons encodes independent features.

The storage capacity of the Hopfield network decreases when the stored patterns are correlated [11]. A similar decrease is also observed in the case of FamE. For many real-world applications, the binary representations fo records are correlated. For example when the records contain text in ASCII code, then the most significant bit of each byte is equal to zero because in ASCII code because letters have code values less than 128 in ASCII. As demonstrated in section 5, the correlation in data decreases the capacity dramatically, hence normally some form of pre-processing will be required before FamE is used. In the remainder of this section a few such pre-processing techniques are suggested, but usually, specialised pre-processing will need to be developed for each application.

When the records in a particular application contain text, the simplest method for removing correlation is to code the text into smaller numbers of bits, or remove the most significant bit from each byte. We will refer to this method as *7-bit coding*. Another simple technique consists of using a code in which each alphanumerical symbol is assigned a random 8-bit code instead of its standard ASCII code. We will refer to this method as *random coding*. Since this code is generated randomly, it possesses much less structure than ASCII and the text records coded by random coding are less correlated than those coded in ASCII. However, these two simple operations can only go so far in reducing correlation.

The method most commonly used for removing correlation is Principal Component Analysis (PCA) [20]. Although PCA is normally used for data having continuous values, it may also be used for binary-valued data. The number of inputs to PCA may be set as the number of bits. The number of outputs should be smaller since PCA also removes redundancy. PCA transforms the records to vectors of real values and those real-valued vectors can be simply converted to binary records by replacing positive values by 1 and negative values by -1.

Removing correlation is one element of pre-processing, but one should also ensure that similar pieces of information (from the point of view of the user) have similar binary representations in Hamming space. This pre-processing is specific

**Table 1.** Meaning of Bits in Phonetic Coding

| vowels | | | consonants | | | | |
|---|---|---|---|---|---|---|---|
| lips pos. | tongue pos. | | voicing | pos. of articulation | | manner of articulation | |
| spread | front, middle | back, middle | unvoiced | bilabial, labiodental, alvedar | alvedar palatal, velar, grottal | plosive fricative | fricative |

for each application, but here we show an example for record de-duplication - an application on which we are currently testing FamE. This work is in collaboration with Optima, a company that produces database management software for storing customer data for marketing purposes. These databases may contain several millions of records and the records are quite long. The databases also often contain duplicates of records, e.g., the same person can be included several times with a misspelled name or an address in a different format. The duplicates should be removed because sending marketing information to the same person twice increases the costs and is not good for the company image. Currently, during de-duplication, a special key for each record is generated and each key is compared with all the others. So de-duplication of a database containing $K$ records requires $K^2/2$ comparisons, which is inefficient, especially considering the size of the databases. Hence, de-duplication is an appropriate application for FamE.

Since duplicates are often the result of misspelling, pre-processing should ensure that letters with similar pronunciation have similar representation. In different languages particular letters are pronounced in different ways. Furthermore, in many languages, pairs of letters represent only one sound (e.g., 'th' in English). For simplicity, we assume one sound per letter and Latin pronunciation.

We code each letter into one byte, where each bit represents a certain property of pronunciation. We refer to this coding as *phonetic coding*. The sounds of speech may be classified based on the way in which the air stream is modified by the vocal tract. The two basic groups of sounds are vowels and consonants. In our phonetic coding, we use the first 3 bits to encode vowels and the last 5 bits to encode consonants (i.e., for each vowel, the last 5 bits are set to 0, and for each consonant, the first 3 bits are set to 0).

The first bit of a vowel's code represents lips position during articulation (i.e., spread or round), and the second and third bits represent tongue position (i.e., front, middle or back). The first bit of a consonant's code (i.e., fourth bit of the phonetic code) represents voicing (i.e., voiced or unvoiced), the second and third bits represent the position of articulation (i.e., bilabial, labiodental, alvedar, palatal, velar or grottal), and the fourth and fifth bits represent the manner of articulation (i.e., plosive, fricative, nasal, lateral or aproximal) [18]. The detailed meaning of each bit is shown in Table 1.

**Table 2.** Comparison of Errors Made by FamE on Text with Different Pre-processing Methods

| Number of patterns per network | 447 | 175 | 106 |
|---|---|---|---|
| Theoretical prediction of error | 1% | $10^{-4}$ | $10^{-6}$ |
| Error without pre-processing | 67.44% | 62.16% | 54.89% |
| Error with 7-bit coding | 67.25% | 51.74% | 41.85% |
| Error with random coding | 26.19% | 9.49% | 1.86% |
| Error with PCA | 7.2% | 0.37% | 0.03% |
| Error with random coding and PCA | 1.16% | 0.05% | 0.03% |

In phonetic coding, the codes of letters with similar pronunciation, such as 't' and 'd', and 'm' and 'n', differ only in one bit, while the codes of letters with different pronunciations are very different. Other methods of pre-processing suitable to text are detailed in [9].

## 5 Simulation Results

Section 3 compares the storage capacity obtained in simulation with theoretical predictions for random, uncorrelated data. However, as we mentioned in the previous section in most applications the representation of records are correlated. This section shows how FamE performs for correlated data. We describe tests carried on two sets of data, one containing text and one made up of customer records.

We tested FamE on records containing text by taking a large amount of text, dividing it into parts of equal length, and building a database with records containing these parts of the text. As the text we chose part of The Holy Bible, Luke's Gospel, chapters 1-10. The text occupied 65390 bytes. Different methods of pre-processing were tested and the size of the records was chosen in such a way that each record after pre-processing occupied 21 bytes. Hence, the size of the network was 21x8 = 168 neurons. The number of networks used to accommodate the whole database depended on the choice of probability of error. The algorithm was tested for each network storing 447, 175 and 106 patterns - these numbers were selected because they yield theoretical predictions of error of 1%, $10^{-4}$ and $10^{-6}$, respectively. To eliminate non-recognition errors, the weights were modified according to Equation 3 in two epochs, and any record still not recognised was written to the next network (see section 4). After writing information about the records to the network, the weights of the neurons were converted to binary values. Then, the performance of the algorithm was tested by checking the familiarity of all records written to the network and an equal number of records from the database which had not been written to the network. The errors were averaged over all the networks used to accommodate the database. The errors obtained for different methods of pre-processing are shown in Table 2.

**Table 3.** Comparison of Errors Made by FamE on Customer Database with Different Pre-processing Methods

| Number of patterns per network | 228 | 89 | 54 |
|---|---|---|---|
| Theoretical prediction of error | 1% | $10^{-4}$ | $10^{-6}$ |
| Error without pre-processing | 52.09% | 45.39% | 41.96% |
| Error with random coding | 47.46% | 30.49% | 20.77% |
| Error with phonetic coding | 66.82% | 54.59% | 47.46% |
| Error with PCA | 11.08% | 0.51% | 0.05% |
| Error with random coding and PCA | 0.74% | 0.0% | 0.0% |
| Error with phonetic coding and PCA | 5.63% | 0.19% | 0.03% |

When FamE is applied to text without any pre-processing, it performs even worse than chance. When one observes the results for particular networks, one can see, that for the first network the error is about 50% (i.e., it performs at random) and for the following networks the error increases. This increase comes from the fact that records not recognised by one network are in the set for the next network. The non-recognised records are the ones which are particularly difficult for FamE, so they increase the error of the following networks.

When 7-bit coding is used (before pre-processing the records had a length of 24 bytes), the error is still very large. When random coding is used, the error decreases slightly but is still unacceptably large.

The error is strongly reduced by using PCA pre-processing. Before pre-processing the records had a length of 35 bytes. They were divided into 7 chunks of 5 bytes, and PCA was applied to each chunk separately to reduce the size of the PCA network. Each chunk was reduced to 3 bytes so the records were reduced to 3x7 = 21 bytes. When PCA was applied to text coded in ASCII, the error decreased to reasonable values (row 'Error with PCA' in Table 2). It is still far from the theoretical prediction because the correlation in ASCII code is very large. Although the outputs from PCA are always uncorrelated [20], the patterns delivered to the familiarity discrimination network after PCA pre-processing do not have to be, since they are created by converting real-valued PCA outputs to binary patterns. When one first applies random coding, and then PCA, the actual error approaches the predicted one (last row of Table 2).

The performance of FamE was also tested on the records from a customer database provided by Optima. For simplicity, only three fields, namely first name, last name and city, were selected. Different pre-processing methods were tested and the size of the records was chosen in such a way that each record occupied 15 bytes following pre-processing. Hence, the size of the network was 15x8 = 120 neurons. The algorithm was tested for each network storing 228, 89 and 54 patterns - corresponding to theoretical predictions of error of 1%, $10^{-4}$ and $10^{-6}$, respectively. Non-recognition errors were avoided using the same method as in the previous experiment. The errors obtained for different pre-processing methods are shown in Table 3

**Table 4.** Energy Function for Records with Similar Pronunciations for Different Types of Coding

| | | | Energy Value | |
|---|---|---|---|---|
| First Name | Last Name | City | Phonetic | Random |
| Vincent | Hopfield | Bristol | -600 | -656 |
| Fynsind | Habfield | Bryztol | -424 | -4 |
| James | Bond | London | -156 | 24 |
| Anakin | Skywalker | Deathstar | -36 | -40 |

The second, third and fourth rows of Table 3 show errors obtained with different types of coding without PCA. In these experiments, each recorded consisted of 15 bytes, where each group of 5 bytes encoded the first five letters of the first name, last name and city, respectively. Table 3 shows the error is very large without PCA.

The last three rows of Table 3 show the errors when PCA pre-processing was used. Before applying PCA, each record consisted of 21 bytes, where 7 bytes were used for each field of the record. The size of each record was reduced to 15 bytes using a single PCA network. The error obtained for random coding is lower than the one obtained with phonetic coding, because phonetic coding gives more correlated records. When one looks for an exact match, phonetic coding is the most appropriate, but in the case of de-duplication we want records with similar pronunciations to be classified as familiar as well.

We performed another experiment to check how random and phonetic codings work for records with similar pronunciations. We added another record to the Optima database: "Vincent Hopfieldi, Bristol". From each record, the first seven letters of the first name, last name and city were encoded using random coding and phonetic coding. The size of each record was then reduced from 21 to 10 bytes using PCA. The database consisted of 54 records and was written to a single Hopfield network of 100 neurons. Then, the energy of the network was checked for the stored record "Vincent Hopfiled", a misspelled version "Fynsind Habfield" and two other records not stored in the database. The values of the energy function are given in Table 4. none of the names in Table 4 were in the original Optima database.

Table 4 shows that for both types of coding, the energy is low for the stored record "Vincent Hopfield, Bristol", and is much closer to zero for records which have not been stored, namely "James Bond, London" and "Anakin Skywalker, Deathstar". However, the energy for the record "Fynsind Habfield, Bryztol" is low for phonetic coding and close to zero for random coding. That is because the representations of "Fynsind Habfield, Bryztol" and "Vincent Hopfield, Bristol" are very similar for phonetic coding, and completely different for random coding. Hence, the phonetic coding is more appropriate when one requires records with similar pronunciation to be classified as familiar.

After analysing the binary representations of "Fynsind Habfield, Bryztol" and "Vincent Hopfield, Bristol" created by phonetic coding and PCA, one can observe that the first bits are mostly the same, while the later bits are different. This is due to the fact that the first principal components are directions in which the data have the highest variance, so the first outputs from PCA carry the most information about the record [20]. The later outputs carry much less information and hence a small change in the record may change their values very much. This property is undesirable, because even very similar records may have many different bits in their representations. Furthermore, when the first outputs of PCA are transformed to binary values, much information is lost by conversion from real to binary numbers. These properties show that PCA, although very simple to implement, is not an ideal method of pre-processing for this application and it would be interesting to investigate other methods of feature extraction.

## 6 Discussion and Conclusion

It is interesting to compare FamE with other algorithms which may potentially be used for familiarity discrimination, e.g., hashing. Hashing is used in databases, where records are often divided into a large number $B$ of buckets. A hashing function takes a record's key and produces an integer between 0 and $B$-1, determining in which bucket the record should be stored.

Let us consider the following algorithm for familiarity discrimination. Instead of storing records let us just store the values of the hashing function for these records. To check whether a record is stored in the database, we simply compute the value of the hashing function and check whether this value is stored in the database.

The above algorithm seems to be very naive and vulnerable to errors. However, with a good hashing function, whose outputs have a large enough number of bits, the probability of error is very small. For example, if the database has $10^6$ records and the value of the hashing function is encoded on 5 bytes, the probability of error is less than 10−6. Thus, great "compression" of information may be achieved, as in FamE. In addition, it is possible (but with very low probability) that a new record maps to the same value of the hashing function as one of the stored records. Hence, it would be classified as a stored record and the algorithm would make false-recognition errors. It is interesting that non-recognition errors will never be made by this hashing-based algorithm, analogously to the fact that non-recognition errors may be eliminated from FamE.

Familiarity discrimination using the hashing function is very vulnerable to noise, however. If two patterns differ even in a single bit, their values under a hashing function may be very different. On the other hand, FamE is very robust to noise, since it classifies a pattern as familiar even if it differs from the stored one in several bits.

The FamE algorithm is suitable for hardware implementation. Firstly, all the internal summations in Equation 2 may be done in parallel for each $i$. Hence, calculating energy could be done in just two processing steps. Secondly, FamE

is very robust to damage - loss of connections between neurons or even of whole neurons causes only a decrease in capacity proportional to the damage [5].

If such fast implementation were available, FamE could be used for searching massive databases, when one does not require an exact match. Currently, searching databases for near-exact matches is slow because normal searching techniques such as indexing or hashing are not applicable. Large databases could be divided into parts and records from each part encoded in the weights of the Hopfield network thus creating a kind of "neural index". In order to find a record one can check the energy in each network and in this way identify whether a similar record is stored in the database and in which part. One may then search only the identified part using other techniques. The property of FamE that non-recognition errors can be eliminated guarantees that if the record exists in the database it will always be found. The false-recognition errors do not affect this application because, even if the algorithm falsely indicates that the record is stored in one part of the database, the further search of this part will show that it is not stored.

This paper discusses the Familiarity discrimination based on Energy (FamE) algorithm, inspired by the presumed computations of the perirhinal cortex. The algorithm allows fast and accurate familiarity discrimination with high storage capacity. The initial experiments demonstrate that FamE may be applied to real-world problems. Many new applications for FamE are likely to emerge in the future, especially due to the increasing sizes of Internet databases.

**Acknowledgements.** We are grateful to Steven Cole for useful comments and help in implementation, and to Optima for providing the database and support. This work is supported in part by ORS and MRC grants.

## References

1. Aggleton, J.P. and Shaw, C. (1996). Amnesia and recognition memory: a re-analysis of psychometric data. *Neuropsychologia*, **34**:51-62.
2. Aggleton, J.P. and Brown, M.W. (1999). Episodic memory, amnesia and the hippocampal-anterior thalamic axis. *Behavioral Brain Science*, **22**:425-498.
3. Amit, D.J. (1989). *Modelling Brain Function*. Cambridge University Press, Cambridge, UK.
4. Barlow, H.B. (1989). Unsupervised Learning. *Neural Computation*, **1**:295-311.
5. Bogacz, R., Brown, M.W. and Giraud-Carrier, C. (1999). High capacity neural networks for familiarity discrimination. In *Proceedings of the International Conference on Artificial Neural Networks (ICANN'99)*, Edinburgh, UK, 773-776.
6. Bogacz, R., Brown, M.W, and Giraud-Carrier, C. (2000). Model of familiarity discrimination in the perirhinal cortex. Submitted.
7. Bogacz, R., Brown, M.W. and Giraud-Carrier, C. (2000). Frequency-based error back-propagation in a cortical network. To appear in *Proceedings of the International Joint Conference on Neural Network (IJCNN'00)*, Como, Italy.
8. Brown, M.W. and Xiang, J.Z. (1998). Recognition memory: Neuronal substrates of the judgement of prior occurrence. *Progress in Neurobiology*, **55**:149-189.

9. Cole, S. (2000). Record de-duplication using a familiarity discrimination neural network. Final Year Project, University of Bristol, Department of Computer Science.
10. Granger, E., Grossberg, S., Rubin, M.A. and Streilein, W.W. (1998). Familiarity discrimination of radar pulses. *Advances in Neural Information Processing Systems*, **11**:875-881.
11. Hertz, J., Krogh, A. and Palmer, R.G. (1991). *Introduction to the Theory of Neural Computation*. Addison Wesley.
12. Hopfield, J.J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Science*, **79**:2554-2558.
13. Insausti, R., Juottonen, K., Soininen, H., Insausti, A.M., Partanen, K., Vainio, P., Laakso, M.P. and Pitkanen, A. (1998). MR volumetric analysis of the human entorhinal, perirhinal and temporopolar cortices. *American Journal of Neuroradiology*, **19**:659-671.
14. Kohonen, T., Oja, E. and Ruohonen, M. (1974). Adaptation of a linear system to a finite set of patterns occurring in an arbitrarily varying order. *Acta Polytechnic Scandinavian Electric Engineering*, **25**.
15. Kohonen, T. (1989). *Self-organisation and Associative Memory*. Springer-Verlag, Heidelberg, Third Edition.
16. Murray, E.A. (1996). What have ablation studies told us about the neural substrates of stimulus memory? *Seminars in Neurosciences*, **8**:13-22.
17. Murray, E.A. and Bussey, T.J. (1999). Perceptual-mnemonic functions of the perirhinal cortex. *Trends in Cognitive Science*, **3**:142-151.
18. Owens, F.J. (1993). *Signal Processing for Speech*. Macmillan Press, London.
19. Roberts, S. and Tarassenko, L. (1995). A probabilistic resource allocating networks for novelty detection. *Neural Computation*, **6**:270-284.
20. Sanger, T.D. (1989). Optimal unsupervised learning in a single-layer feedforward neural network. *Neural Networks*, **2**:59-473.
21. Standing, L. (1973). Learning 10,000 pictures. *The Quarterly Journal of Experimental Psychology*, **25**:207-222.
22. Xiang, J.Z. and Brown, M.W. (1998). Differential neuronal encoding of novelty, familiarity and recency in regions of the anterior temporal lobe. *Neuropharmacology*, **37**:657-676.