Multi-speaker voice cryptographic key generation

L. Paola García-Perera, J. Carlos Mex-Perera and Juan A. Nolazco-Flores
Computer Science Department, ITESM, Campus Monterrey
Av. Eugenio Garza Sada 2501 Sur, Col. Tecnológico
Monterrey, N.L., México,C.P. 64849
paola.garcia@itesm.mx, carlosmex@itesm.mx, jnolazco@itesm.mx

Abstract

In this work we present a procedure for generating binary vectors, which can be used as keys for cryptographic purposes. This research is based on Automatic Speech Recognition Technology and Support Vector Machines. Keys bits are produced by making a distinction among the phoneme features of the users employing hyperplanes. The implementation of the method is described and the statistics are computed for several number of users. The results obtained show that the proposed method is sufficiently robust to reliably regenerate the key.

1 Introduction

The necessity of having a cryptographic key is an important issue for nowadays secure systems. The objective of a key is to maintain the information secret from any unauthorised person; then, it should be very difficult to guess or obtain. Typically, secure systems are designed based on cryptographic primitives, where its security relies on the privacy of the keys.

Lately, one of the problems of typical authentication systems is that even for the simplest algorithms the authorised user should remember the correct password; otherwise, the system would deny access to the information. However, remembering strong passwords is difficult for humans. Furthermore, if the chosen password is in a dictionary it is extremely vulnerable to cracking [8, 10]. The goal of the biometric key is to produce a password using the intrinsic characteristics of some specific user distinguishing quality (voice, iris, fingerprint and face) with the consequence that remembering a complex password would not be necessary. Since voice is a natural process and produces different utterances in short time, it gives advantages among other biometrics allowing to assure the user's identity. For instance, the system's interface can always request to him a random passphrase, avoiding the use of recorded voice by an unauthorised person. In this research we focus on voice, and its characteristics, to properly obtain a reliable cryptographic key. This type of key generation has several applications in security; for example, access control to networks either to a computer system or to a restricted area. It can also be useful for remote access control by telephone.

Since speech is an important element in this research, we have the advantage of the extensive research done in the field. Our cryptographic scheme is based on the knowledge of the phonemes, then it is important to know the starts and ends of the phonemes given a speech signal. Fortunately, the output of an Automatic Speech Recognition (ASR) gives this information. The ASR primary task is to obtain the transcription for a given speech signal. Moreover, if properly configured the ASR can also obtain the starts and ends of the phonemes.

The main challenge of our research is to produce a cryptographic key that should repeatedly be the same for one user under certain conditions. An experiment by Monrose *et. al* [9] showed a method to produce this key. In their research an acoustic space was formed derived from the utterances of a significant group of people. A vectorial quantization was applied and a set of 20 centroids was obtained. The raw voice signal of each user was transformed into 12-Cepstral Coefficients frame sequences. Afterwards, those sequences were segmented. To optimise the segmentation for their purposes, the likelihood function between subsequences and the matching centroids needed to be maximised. An iterative process using the Viterbi algorithm was used. From each final segment a media was obtained. The substraction of the media of a segment from the value of the matching centroid resulted in a set of vectors to be mapped into a binary class. A partition plane for the vector space was suggested, and a binary biometric key was conformed. The last step of the proposal of Monrose *et. al* referred to the hardening of the key using shared schemes.

The underlying part of that research was the generation of the descriptor, since the classification planes decide the final key. However, it has a serious drawback, the search

0-7803-8735-X/05/$20.00©2005 IEEE

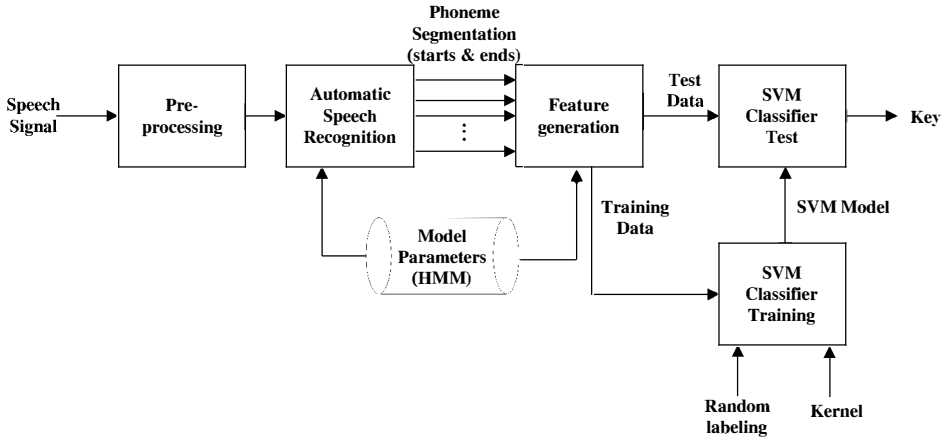

Figure 1. System architecture

of the partition plane is difficult due to the fact that infinite planes are possible. The key depends strictly on the chosen plane, this means that the key of the user will be unknown until the plane is set. A more flexible way to produce a key is always desirable. A control on the assignation of the key values can enhance the cryptographic performance. It is also interesting to explore the possibility to produce a key by the use of an algorithm that can handle all the voice important characteristics without discarding any of them.

In the next section we present our proposal. Section 3 through 5 describes the fundamental elements of ASR, the feature descriptor formation and SVM techniques. Section 6 examines the experimental methodology and the final section presents the results and discusses future research problems.

2 Proposal

The purpose of this work is to produce a cryptographic key given the voice signal and the spoken user passphrase. The imperative challenge of this goal lies in the search of suitable set of phoneme planes that can significantly partition the users' data.

A general view of the mechanism is displayed in Figure 1. In the first stage, the speech signal is pre-processed and the Automatic Speech Recognition (ASR) technique is used to extract the phoneme model and the phoneme starts and ends. Using the last new sets of data, a resulting set of features is conformed. This set of features are the input of the Support Vector Machine (SVM) and will produce a users' phoneme classification according to a kernel and bit specifications. Finally, using the SVM model the key is generated. Each part will be discussed in the following sections.

3 Automatic speech recognition

The ASR primary task is to obtain the transcription of a given speech signal. In addition, the ASR can also obtain the starts and ends of the phonemes if properly configured, see Figure 2.

The speech signal is pre-processed by calculating its Mel Frequency Cepstral Coefficients (MFCC) [15]. MFCCs are a common transformation that has shown to be very robust. To obtain the MFCC (Figure 3), each speech signal is divided into short frames. For each frame the amplitude spectrum is obtained using the Discrete Fourier Transform (DFT). Afterwards, the spectrum is converted to the log-scale. Then, the Filter Banks are used to smooth the scaled spectrum. Finally, the discrete cosine transform is applied to eliminate the correlation between components. The result is a 13-dimension vector, each dimension corresponding to one parameter. To emphasise the dynamical features of the speech in time, the time-derivative (Δ) and the time-acceleration (Δ^2) of each parameter are calculated. Then, the final representation is a 39 dimension vector formed by 12-dimension MFCC, followed by 1 energy coefficient,

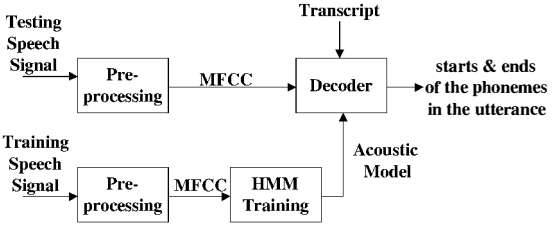

Figure 2. Speech recognition architecture

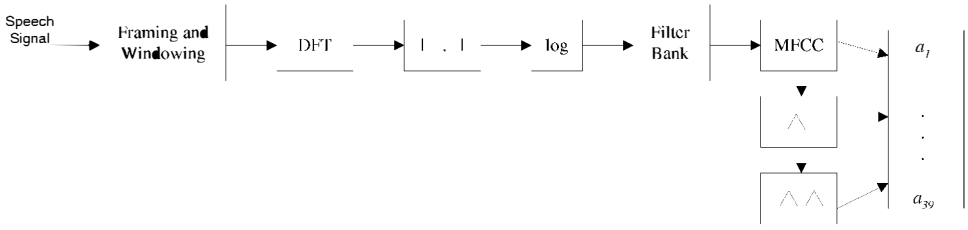

Figure 3. Pre-processor

13Δ and 13 Δ^2 coefficients.

The training branch leads to a production of an acoustic model. Given the acoustic model, the transcription of the utterance, and the parametrised speech a decoder produces the phonemes' starts and ends. This process is known as *speech forced alignment.*

In this training phase the system learns the patterns that represent the speech sound. Depending on the application the units can be words, phonemes, or syllables. The Hidden Markov Model (HMM) is the leading technique for acoustic modelling [14]. An HMM is characterised by the following, see Figure 4:

$A = \{a_{ij}\}, a_{ij} = Prob\{q_j \text{ at } t+1 | q_i \text{ at } t\}$ state transition probability distribution

$B = \{b_j(O_t)\}, b_j(O_t) =$ observation probability distribution

$\pi = \{\pi_i\} = Prob\{q_i \text{ at } t = 1\}$ initial state distribution

$O = \{O_1, O_2, ..., O_T\} =$ observation sequence (input sequence)

$T =$ length of observation sequence

$Q = \{q_1, q_2, ..., q_N\}$ hidden states in the model

$N =$ number of states

The compact notation $\lambda = (A, B, \pi)$ is used to represent an HMM [4]. The parameter set N, M, A, B, and π is calculated using the training data and it defines a probability measure $Prob(O|\lambda)$.

The resulting model has the inherent characteristics of real speech. The output distributions of the HMM are commonly represented by Gaussian Mixture Densities with means and covariances as important parameters, see Figure 5. Depending on the application one or more gaussians can be included per state. But also, one or more states are also possible for a given reference sound. To determine the parameters of the model and reach convergence it is necessary to first make a guess of their value. Then, more accurate results can be found by optimising the likelihood function and using Baum-Welch re-estimation algorithm.

For the purpose of this research, the HMM will model phonemes. Let, P be the set of phonemes.

$$P = \{x | x \text{ is a phoneme}\}$$

The states and the gaussians will make reference to the set P. The phonemes were chosen as acoustic model because it is possible to generate larger keys compared with the length of the key that could be generated using complete words.

Assuming the phonemes are modelled with a three-state left-to-right HMM, and assuming the middle state is the most stable part of the phoneme representation, $\mathcal{C}_P = \{C_i\}$ will denote the set of means of central gaussian of the middle state vectors.

In the decoding phase (Figure 2), the phonemes' starts and ends of each utterance are obtained using the Viterbi algorithm.

4 Feature generation

After obtaining the starts and ends of the phonemes of the speech signal, the MFCC's for each phoneme in the utterances are arranged forming the sets $R^u_{i,j}$, where i is the index associated to each phoneme, j is the j-th user, and u is an index that starts in zero and increments every time the user utters the phoneme i. Then, the feature vector is defined as

$$\psi^u_{i,j} = \mu(R^u_{i,j}) - C_i$$

where $\mu(R^u_{i,j})$ is the mean vector of the data in the MFCC set $R^u_{i,j}$, and $C_i \in \mathcal{C}_P$ is known as the matching phoneme mean vector of the model. Let us denote the set of vectors,

$$D_p = \{[\psi^u_{p,j}, b_{p,j}] \mid \forall\, u, j\}$$

where $b_{p,j} \in \{-1, 1\}$ is the key bit or class assigned to the phoneme p of the j-th user.

D_p is partitioned into two sets, those vectors in D_p to be used for the training phase are grouped in set D^{train}_p and those for the test phase form D^{test}_p.

5 Support vector machine

The SVM is a method widely used for classification, derived by Vapnik and Chervonenkis [1, 3]. The goal of basic

SVM is to obtain a model to perform vector classification in one of two classes. Two stages conform the SVM process: training and testing.

The first step in the training process is to format the data in vectors, and each vector is labelled according to its class. Afterwards, the following set of pairs are defined $\{x_i, y_i\}$; where $x_i \in \mathbb{R}^n$ are the training vectors and $y_i = \{-1, 1\}$ are the labels. The SVM learning algorithm finds an hyperplane (w, b) such that,

$$\min_{x_i,b,\xi} \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \tag{1}$$

$$\text{subject to } y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i \tag{2}$$

$$\xi_i \geq 0$$

where ξ_i is a slack variable and C is a positive real constant known as a tradeoff parameter between error and margin. For a better understanding, see Figure 6.

Equations 1 and 2 can be transformed into a dual problem represented by the Lagrange multipliers α_i. Thus,

$$\sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{l} \alpha_i \alpha_j y_i y_j \langle \phi(x_i), \phi(x_j) \rangle \tag{3}$$

$$\sum_{i=1}^{l} \alpha_i y_i = 0, C \geq \alpha_i \geq 0. \tag{4}$$

α_i can be solved as a quadratic programming (QP) problem. The resulting values $\alpha_i \in \mathbb{R}$ have a close relation with the training points x_i; each α_i represents the strength with which a point x_i is associated in the final decision function. Only a subset of the points will be associated with a non-zero α_i, called *support vectors* (SV). Those SV are the points that lie closest to the decision boundary and therefore will conform the decision hyperplane.

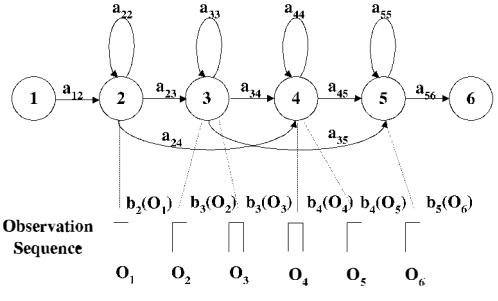

Figure 4. Left-to-right HMM, *1...6* states, a transition probabilities, b output probabilities, O observation sequence

To extend the linear method to a nonlinear technique, the input data is mapped into a higher dimensional space by function ϕ. However, exact specification of ϕ is not needed: instead, the expression known as kernel $K(x_i, x_j) \equiv \phi(x_i)^T \phi(x_j)$ is defined. There are different types of kernels, but the most common are

- linear $K(x_i, x_j) = x_i^T x_j$
- polynomial $K(x_i, x_j) = (\gamma x_i^T x_j + r)^d, \gamma > 0$
- Radial Basis Function (RBF) $K(x_i, x_j) = e^{(-\gamma ||x_i - x_j||^2)}, \gamma > 0$
- Sigmoid function $K(x_i, x_j) = tanh(\gamma x^T x_j + r)$

where γ, r and d are parameters.

The decision function is expressed as

$$y(x) = sign[\sum_{k=1}^{\#SV} \alpha_k y_k K(x_i, x) + b]. \tag{5}$$

For better results, the parameters of the kernels should be tunned to find the best classification of the training data. When convergence is reached, the kernel and its final parameters are evaluated using Equation 5 and the test data. The statistics can give a guide on the performance of the classifier and decide to tune or stop.

The SVM has been used for several applications, including biometrics [12, 13]. In this research we employ it for searching a function that can be able to transform a feature vector into a binary number (key bit) arbitrarily selected. In this research we will just focus on the RBF kernel since it can model closed decision surfaces [11], and if one class is encapsulated by another class it can still produce a reliable model. Two parameters need to be tunned: γ and C. After obtaining the best parameters, the test data is ready to be classified.

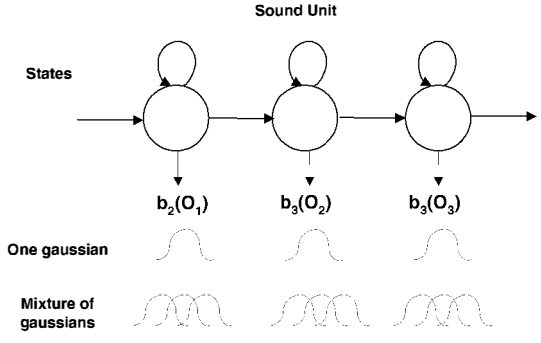

Figure 5. HMM for a sound unit

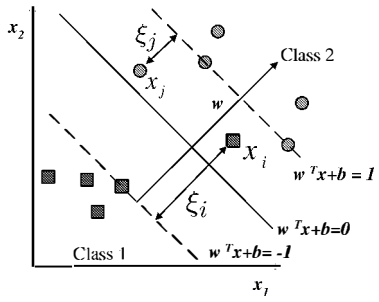

Figure 6. Support vector machine

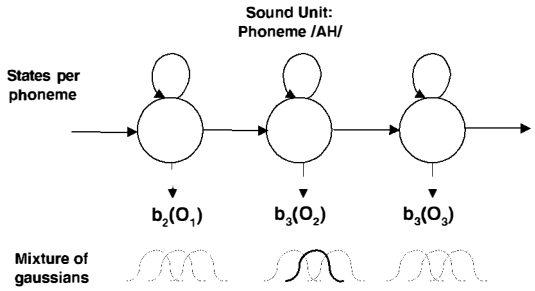

Figure 7. HMM model

For our research the methodology used to implement the SVM training is as follows. Suppose that */AH/* is the unique phoneme that can be uttered from all users. The one bit label (-1 or 1) is assigned to each $D_{/AH/}$. However, if one user utters the phoneme */AH/* several times, the value of the bit will remain the same. For instance, $f(D_{/AH/}) = 1$ will be the same for all the utterances of the same user. The procedure is similar for the rest of the users, but the value of the bit $f(D_{/AH/})$ can change from user to user. The procedure is extended to all the phonemes. Afterwards the parameters for C and γ are stablished.

In the test stage, we evaluate the model produced by the SVM using the test features. The statistics can be made in terms of errors per phoneme according to certain quantity of users. The final key is obtained by concatenating the bits produced by each phoneme. For instance, if a user utters two phonemes: $/F/$ and $/AH/$, the final vector key is as follows,

$$K = [f(D_{/F/}), f(D_{/AH/}]$$

For this work, the key bit assignation is arbitrary. Thus, the keys have liberty of assignation, and the entropy for the keys can be easily maximised if they are given in a random fashion with a uniform probability distribution. This research handles just binary assignation, however, an M-ary bit assignation per phoneme might be also possible for future research.

5.1 Experimental methodology

The experiments were performed using the YOHO database [2] [5]. YOHO contains the voice utterances of 138 speakers of different nationalities speaking three pairs of numbers in each utterance. For instance, "Thirty-Two, Fourty-One, Twenty-Five". The utterances are divided in two sets enrollment (training) and verification (test). Each user produces 96 utterances for enrollment, and 40 for verification. The Hidden Markov Models Toolkit (HTK) by Cambridge University Engineering Department [6] was used for the pre-processing, decoding and training.

The important resulting parameters of the Hidden Markov model are the nineteen means and variances of the phonemes: */AH/*, */AO/* */AX/*, */AY/*, */EH/*, */ER/*, */EY/*, */F/*, */IH/*, */IY/*, */K/*, */N/*,*/R/*, */S/*, */T/*, */TH/*, */UW/*, */V/*, */W/*. The model of each phoneme is composed by three states, and three gaussians per state, see Figure 7.

For the objective of this research, the central gaussian of the middle state of each phoneme (highlighted in Figure 7) and the starts and ends of the phoneme in the utterance are the input of the feature generator. Following the method already described in section 4, the subsets D_p are formed. It is important to note that the cardinality of each D_p can be different since the number of the same phoneme utterances can vary from user to user. For the training, the number of vectors picked for generating the model is the same. The number of test vectors that each user provided can be different.

SVMLight by Thorsten Joachims was used to implement the Classifier [7]. We explored the RBF kernel and the influence that its two main parameters, γ and C, can have in performance. Making a grid search of the parameters, we found that γ=0.01 and C=9 are values that can perform good classification among the four sets of different number of users(10, 20, 30, and 50).

To obtain reliable statistics several trials were perfomed for each set. A random assignation of the bits was perfomed for the training and its corresponding test classification results are depicted in the next section.

The complexity of the complete process to generate the key can be viewed from two perspectives: ASR and SVM. For the ASR, the complexity depends on the number of data required for the training phase, the vocabulary used, the number of states and gaussians employed in the model. The time of computation can vary depending on all of this factors. For the SVM, the formulation of the problem is based on the dimension of the kernel. Furthermore, several methods have been suggested to reduce the complexity of the

large data quadratic programming problem. The SVM algorithm employed for this research spends most of its time in iterations of the kernel evaluations, and is of the order $O(Mq)$, where M is the number of training examples and $q << M$ is a parameter [16]. The research on both fields is in constant evolution to reduce the computation time.

6 Experimental Results

The behaviour of the SVM is given in terms of the classification accuracy on the test data, computed by the ratio

$$\eta = \frac{\text{classification matches in test data}}{\text{total number of vectors in test data}}. \quad (6)$$

It is important to explore the behaviour for different types of parameters. Table 1 shows the global accuracy average for different number of users: 10, 20, 30, 50 (all phonemes are included and several trials were perfomed). If the number of users is incremented the value of η decreases, then the search of the plane should be more robust to obtain a suitable plane. Note that the variances are very small, thus the global average statistic is closed to the values of η for individual users and phonemes.

Table 2 shows the behaviour of the global accuracy per phonemes for all the different number of users. Although there exist phonemes that give a slightly lower value of η all of them show an accuracy above 0.80.

Finally, Table 3 shows an example of the behaviour of each user from a group of 10. We calculate the average accuracy performed for all the phonemes, and also, the average of the variance of the accuracy of each phoneme per user. It is important to note that a reliable generation of the key is possible since the accuracy is sufficiently high and the variance is small.

7 Conclusion

We have presented a method to produce a cryptographic key from voice based on phoneme segmentation, where one key bit was assigned to each user phoneme. Combined techniques of ASR and Support Vector Machines have been used in this work.

Numb. of Users	*Glob. Average for* η	*variance of* η
10	.927	0.0000136843
20	.9088	0.0000133
30	.9004	0.0000142
50	.8871	0.000397

Table 1. Global average for η for different number of users

Phoneme	*Average value for* η
/AH/	.922
/AO/	.9149
/AX/	.9157
/AY/	.9508
/EH/	.9247
/ER/	.9297
/EY/	.8884
/F/	.8378
/IH/	.9001
/IY/	.8842
/K/	.8557
/N/	.9182
/R/	.8917
/S/	.8628
/T/	.8686
/TH/	.8393
/UW/	.9188
/V/	.9086
/W/	.8863

Table 2. Global η for different phonemes

These results indicate that the ability of the proposed method to distinguish phonemes of specific users is quite good and provides good results for any key and user. However, it is desirable a classification error near to zero, *i.e* error correction mechanisms might be considered in future research since the bits of the key must not present any error. Adding extra bits for error correction demands more phonemes in the passphrase but it might reduce the possibility of wrong key production.

Besides, future studies on a M-ary key can be useful to increase the number of different keys available for each user given a fixed number of phonemes in the passphrase.

user	η	*variance*
1	.90638	0.014202
2	.90513	0.015751
3	.92189	0.014548
4	.96076	0.0059596
5	.95179	0.0078196
6	.92059	0.012574
7	.92063	0.01163
8	.93509	0.0096878
9	.90166	0.017299
10	.9443	0.0088079

Table 3. η for 10 users, all the phonemes

References

[1] B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal mar-gin

classifiers. In Proceedings of the Fifth Annual Workshop on Computational Learning Theory, 1992.

[2] J. P. Campbell, J. P., Jr. Features and Measures for Speaker Recognition. Ph.D. Dissertation, Oklahoma State University, 1992.

[3] C. Cortes, and V· Vapnik. Support-vector network. Machine Learning 20, 273297, 1995.

[4] S. Furui. Digital Speech Processing, Synthesis, and Recognition. Mercel Dekker,inc. New York, 2001.

[5] A. Higgins, J. Porter and L. Bahler. YOHO Speaker Authentication Final Report. ITT Defense Communications Division, 1989.

[6] HTK Hidden Markov Model Toolkit home page. http://htk.eng.cam.ac.uk/

[7] T. Joachims, SVMLight: Support Vector Machine, SVM-Light Support Vector Machine http://svmlight.joachims.org/, University of Dortmund, November 1999.

[8] D.V· Klein, Foiling the Cracker; A Survey of, and Improvements to Unix Password Security, (original paper) Proceedings of the United Kingdom Unix User's Group, London ENGLAND, July 1990

[9] F. Monrose, M. K. Reiter, Q. Li , S. Wetzel. Cryptographic Key Generation From Voice. Proceedings of the IEEE Conference on Security and Privacy, Oakland, CA. May, 2001.

[10] R. Morris and K. Thompson. Password security: A case history. Communications of the ACM, 22(11):594–597, 1979.

[11] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. IEEE Neural Networks, 12(2):181-201, May 2001.

[12] E. Osuna, R. Freund, and F. Girosi. Support vector machines: Training and applications. Technical Report AIM-1602, MIT A.I. Lab., 1996.

[13] E. Osuna, R. Freund, and F. Girosi, Training Support Vector Machines: An Application to Face Recognition, in IEEE Conference on Computer Vision and Pattern Recognition, pp. 130-136, 1997.

[14] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE, 77(2):257–286, February 1989.

[15] L.R. Rabiner and B.-H. Juang. Fundamentals of speech recognition. Prentice-Hall, New-Jersey, 1993.

[16] B. Schölkopf, C.J.C. Burges and A.J. Smola. Advances in kernel methods: support vector learning. MIT Press, 1999.

8 Acknowledges

The authors would like to acknowledge the Catedra de Seguridad, ITESM, Campus Monterrey and the CONACyT project CONACyT-2002-C01-41372 who partially supported this work.