T.C.
Bahçeşehir Üniversitesi

SECURITY LEVEL CLASSIFICATION FOR CONFIDENTIAL DOCUMENTS BY USING ADAPTIVE NEURO-FUZZY INFERENCE SYSTEMS

Master's Thesis

Erdem ALPARSAN

İstanbul, 2010

T.C.
Bahçeşehir Üniversitesi
The Graduate School of Natural and Applied Sciences
Computer Engineering

SECURITY LEVEL CLASSIFICATION FOR CONFIDENTIAL DOCUMENTS BY USING ADAPTIVE NEURO-FUZZY INFERENCE SYSTEMS

Master's Thesis

Erdem ALPARSAN

Advisor: Assoc.Prof.Dr. Adem KARAHOCA

İstanbul, 2010

ACKNOWLEDGEMENTS

I would like to thank all people who have helped and inspired me during my study.

Especially, I offer my sincerest gratitude to my supervisor Assoc. Prof. Dr. Adem Karahoca and to my co-supervisor Dr. Hayretdin Bahşi, who have supported me, thought-out my thesis with their experience and knowledge. It would be impossible to complete this study without their encouragement, motivation and guidance.

I owe my deepest gratitude to my mother Sevim Sema Alparslan, for endless love and support throughout my life. Not only in this study but also in every moment in my life her encouragement made everything easier than it is.

Finally I would like to thank my wife, Hatice Alparslan, for her everlasting love, endless support and encouragement in every part of my life. Because of her patience and tolerance I owe her a great appreciation.

ABSTRACT

SECURITY LEVEL CLASSIFICATION FOR CONFIDENTIAL DOCUMENTS BY USING ADAPTIVE NEURO-FUZZY INFERENCE SYSTEMS

Erdem Alparslan

Computer Engineering

Supervisor: Assoc.Prof.Dr. Adem Karahoca

06 / 2010, 42 pages

In recent years, protecting secure information became a challenge for military and governmental organizations. As a result, well defined security level contents and rules are more preferable than in the past. Each piece of information has its own security level. Correct detection of this security level may lead to apply correct protection rules on information. This study aims to develop a wide perspective classification framework for security critical documents written in Turkish.

Recent studies on text classification are planned on the categorization of the news stories written in English. The framework proposed in this study aims to classify in a much fuzzier domain: security level classification. Thus using fuzzy inference systems are unavoidable to gain a meaningful success from security level classification. Adaptive Neuro-Fuzzy Inference Systems are fitting as a solution for fuzzy security level classification. The fuzzy results are discretized by using recent discretization algorithms and security labels will be extracted from security level scores.

Preprocessing phases in security level classification is nearly similar to the other document classification problems. Making structured the textual data is provided by reformatting the data in TF-IDF form. TF-IDF representation holds each document as a row and each feature (critical words for classification) as a column. Because of the high dimensionality of document classification, a feature selection task is meaningful as a preprocessing task.

Turkish document classification deals with an adscititious problem: Turkish natural language processing. Stemming process is essential for calculating the realistic TF-IDF values of features. Turkic languages are agglutinative languages. So stemming the Turkish words introduces a very difficult natural language processing problem.

The document is organized in straightforward sections. First section makes an introduction to the problem. Next chapters are explaining the algorithm and solutions that are used, experimental settings to test the new framework, solutions of experiments and giving a conclusion and final discussion.

Keywords: Document classification, ANFIS, expert systems, Turkish NLP, ROC

ÖZET

GİZLİ DOKÜMANLARIN UYUMSAL NÖRON-BULANIK ÇIKARIM SİSTEMLERİ YARDIMIYLA GÜVENLİK DERECELERİNİN SINIFLANDIRILMASI

Erdem Alparslan

Bilgisayar Mühendisliği

Danışman: Doç. Dr. Adem Karahoca

06 / 2010, 42 sayfa

Son yıllarda güvenlik derecesi yüksek bilginin korunması, askeri ve kamusal kurumlarda zorlu bir uğraş haline gelmiştir. Bunun sonucu olarak iyi tanımlanmış güvenlik derecesi bilgi ve kurallarına her zaman olduğundan daha fazla ihtiyaç duyulmaktadır. Her bilgi parçacığı kendine özgü bir güvenlik derecesi barındırmaktadır. Bilginin kendine özgü güvenlik derecesinin doğru tespit edilebilmesi, o bilgi için doğru ve uygun koruma kurallarının oluşturulabilmesini de sağlar.

Önceki çalışmalar İngilizce dilinde yazılmış haber metinlerinin sınıflandırılması üzerine hazırlanmıştır. Bu çalışmada, güvenlik derecesi sınıflandırması bulanık bir alanda gerçekleştirilmektedir. Bu nedenle, bulanık tahmin yöntemlerinin kullanılması bu çalışmada tutarlı güvenlik derecesi tespiti yapılabilmesi için kaçınılmaz olmaktadır. Uyumsal nöron-bulanık çıkarım sistemlerinin güvenlik derecesi sınıflandırması için iyi bir çözüm olabileceği üzerinde durulmuştur. Ayrıklaştırma algoritmaları yardımı ile güvenlik seviyesi skorlarından güvenlik etiketleri bulunacaktır.

Güvenlik dereceli sınıflandırmanın ön işleme aşamaları diğer doküman sınıflandırma çalışmalarına benzer şekilde gerçekleştirilir. Verinin yapısal olarak tanımlanması onun TF-IDF denen özel bir forma dönüştürülmesi ile olur. TF-IDF formu her dokümanı bir satır ve her niteleyiciyi (sınıflandırma için kritik sözcükler) bir sütun olarak tanımlar. Doküman sınıflandırmanın çok boyutlu doğasından ötürü bir niteleyicinin seçimi için ön işlemeden geçirilmesi oldukça mantılıdır.

Türkçe doküman sınıflandırmada bir ilave ön işleme problemi ile daha uğraşılmaktadır, bu da Türkçe doğal dil işlemedir. Gövde ayrıştırma işlemi niteleyicilerin gerçek TF-IDF değerlerinin hesaplanabilmesi için kaçınılmazdır. Türkçe sondan eklemeli bir dildir dolayısı ile Türkçe kelimelerin doğal dil işleme tabii tutulması daha zor olmaktadır.

Doküman birbirini takip eden bölümler halinde organize edilmiştir. İlk bölüm konuya bir giriş yapmaktadır. İlerleyen bölümlerde sırasıyla uygulanacak algoritma ve çözümler, yeni oluşturulacak sınıflandırma ana çatısı için deneysel kurgu, deney sonuçları ve son görüş ve saptamalar ele alınmaktadır.

Anahtar Kelimeler: doküman sınıflandırma, ANFIS, uzman sistemler, Türkçe DDİ

TABLE OF CONTENTS

TABLES

FIGURES

ACRONYMS

The Scientific and Technological Research Council of Turkey	:	TUBITAK
National Institute of Electronic and Cryptography	:	UEKAE
Support Vector Machines	:	SVM
Adaptive Neuro-Fuzzy Inference Systems	:	ANFIS
Naïve Bayes	:	NB
Term Frequency – Inverse Document Frequency	:	TF-IDF
Natural Language Processing	:	NLP
Class-Attribute Contingency Coefficient	:	CACC

1. INTRODUCTION

In recent years, we are witnessing the applications of expert systems in information / data security area. Protecting the content of the secure information became a new and difficult challenge because of the various new communication and interaction styles. This protection of secure information is highly related to the security level of the information. Detecting the realistic security level for a document is crucial to assign the best fitting protection rules for this information.

Determining the security level of a document by using expert systems is a document classification problem. The aim of classification of confidential documents is to assign predefined class labels to a new document that is not classified (Joachims 1998). An associated classification framework provides training documents with existing class labels. Therefore supervised, semi-supervised or unsupervised classification algorithms are fitting as a solution to the classification problem. The set of labeled and unlabeled documents for an organization may lead selecting supervised or unsupervised algorithms. For a document set which contains mostly unlabeled documents, choosing an appropriate unsupervised or semi-supervised methodology may present more accurate results. In the other hand if all of the train and test documents are labeled by a subject matter expert, and then by using a supervised algorithm seems obviously more realistic procedure (Feldman and Sanger 2007).

Classification accuracy of textual data is highly related to preprocessing tasks of training and test data (Han and Kamber 2007). These tasks become more difficult in processing unstructured textual data than in structured data. Unstructured nature of data needs to be formatted in a relational and analytical form. TF-IDF (term frequency-inverse document frequency) is preferred to represent text based contents of documents. This representation holds each word stem as an attribute for classification; and each document represents a separated classification event.

Another important task of formatting textual data is stemming. Stemming the Turkish documents is more difficult than the other studies based on English or in other Latin-based languages. Turkic languages which are agglomerative languages, involve diverse

exceptional derivation rules. Therefore stemming of Turkish terms provides some unstable rules varying from structure to structure.

Each of the distinct terms of document set is a dimension of the TF-IDF representation. Hence this representation leads very high dimensional space, more than 10000s dimensions. It is mainly noted that feature selection tasks are critical to make the use of conventional learning methods possible, to improve generalization accuracy, and to avoid over fitting (Joachims 1998).

Recent studies on document classification are performed on text datasets, especially on news stories in English (Joachims 1998; Cooley 1999). We have recently performed a classification of Turkish news stories and obtained a classification accuracy of 90% by using support vector machines (SVM) (Alparslan et al. 2009).

In this study, internal documents of TUBITAK UEKAE (National Research Institute of Electronics and Cryptology) are classified into three classification levels: "secret, restricted and unclassified" by using naïve bayes classifiers (NB), support vector machines (SVM) and adaptive neuro-fuzzy inference systems (ANFIS). Finally, a hybrid approach is proposed which use SVM and its outputs are directed as inputs to ANFIS. By this hybrid approach classification accuracy of 97% has became reached as well for Turkish confidential document set.

Also, security level classification of documents will also provide an extension point for Data Loss Prevention Solutions. Actual systems detect security leakages by using only predefined rules as keyword or regular expression search. With the help of document classification frameworks, these solutions may detect the security breaches according to the contents of documents.

2. PREPROCESSING TASKS

This section clarifies the main steps which must be handled for a document classification process. In data mining, preprocessing is the reduction of incomplete, inconsistent or noisy data from dataset (Han and Kamber 2007). Detecting data anomalies, rectifying them early, and reducing the data to be analyzed can lead to huge payoffs for data mining. But text mining or classification process is slightly different. Data anomalies, inconsistencies or lack is not in question for text mining. But in the other hand most of the classification algorithms need a structured data form to execute their categorization schema (Vapnik 2000). Getting the text based unstructured documents into structured data form is the preprocessing phase of text mining. This phase of document classification needs more preprocessing effort than in classical data mining problems.

2.1. STEMMING

Stemming is the primary and essential preprocessing task of document classification which is related to Natural Language Processing (NLP). It is essential and vital task for calculating realistic TF-IDF values of features (Berry 2003). Any words derived from same root affects the classification in the same sense, in behalf of the same feature. Therefore, effective stemming may improve classification accuracy significantly (Kao and Poteet 2007).

Stemming task for a Turkish NLP study is more difficult than in other languages because of the exceptional derivation rules of Turkish language. For the morphologically simple languages storing all the word inflectional forms in a lexicon and doing the stemming without any morphological analysis is suitable. But in the other hand, for the agglutinative languages like Turkish this approach is not sufficient where a word can take hundreds of different forms after the concatenation of affixes (Sever and Bitirim 2003).

Ex: Some of the different forms of the noun “eye” in Turkish are listed below:

Göz	Eye
Göz-lem	Observation
Göz-lem-ci	Observer
Göz-lem-ci-lik	The job of the observer
Göz-lem-le-dik-ler-im	The ones that I observed

In this study we have used a comprehensive stemmer library *Zemberek* to find out roots of unstemmed words (zemberek 2001). *Zemberek* gives us all the possible stemming structures for a term. User must select one of these stemming suggestions. Our stemming system selects the structure that has the biggest probability of semantic and morphologic patterns of Turkish language. Table 1 shows some stemming examples of Zemberek:

Table 1. Stemming examples of Zemberek

word	stem	suffixes
gözlemlediklerim	gözlem	-le -dik -ler -im
şifrelenmiş	şifrele	-n -miş
yayımladığı	yayım	-la -dık(ğ) -ı
kayıtlarının	kayıt	-lar –ı -nın

But also the problem of choosing morphologically correct stemming structure is a problem in Turkish NLP. For example the word “altıncı” may be translated in English in two forms according to stemming structures. “Altı – ncı” structure means “sixth” but also “altın – cı” means “the person who sells gold”. As one can differ easily the two meanings of word “altıncı” are irrelevant. Some stemming anomalies which are encountered in this study are shown in Table 2:

Table 2. Stemming anomalies

word	predicted stem	real stem
ulusal (national)	ulu (grand)	ulus (nation)
kullanıcılar (users)	kul (slave)	kullan (use)
çalıştırması (execution of sth)	çalışmak (execute)	çal (steal)
yüksek (high)	yük (charge)	yüksek (high)

Although these anomalies of stemming exist, our study does not aim to reach the highest stemming accuracy. Because of the reduced affect of stemming structure on document classification accuracy, we are accepting the present stemming success of the tool *zemberek*.

2.2. FEATURE SELECTION

Features are important stemmed words which have a powerful discriminative adequacy in classification. Many text classification problems struggle 10000s words or maybe more as mentioned before. This high dimensionality may probably cause over fitting and performance problems (Feldman and Sanger 2007; Berry 2003). Assigning some of these words as features may prevent the potential performance problems.

On the other hand, Support vector machines and naive Bayesian classifiers do not need to reduce dimensionality for accuracy. With their ability in effective generalization of high dimensional feature spaces, these classifiers eliminate the need for feature selection, making the application of text categorization considerably easier (Feldman and Sanger 2007; Hsu et. al. 2010). However we still have a performance problem for high dimensional classification problems. A high dimensional classification task may consume lots of time and gives us the same accuracy as a classification in reduced dimensions. Because of these performance handicaps the feature dimension size is reduced by eliminating features with feature selection methods.

Feature selection methods are used to select a subset or list of attributes or variables that are used to construct models for describing data. In text classification problems feature selection means selecting stemmed words which play important roles in classification (Prado and Ferneda, 2007). The purpose of feature selection includes reducing dimensionality and the amount of data needed for learning, removing irrelevant and redundant features, improving algorithms' predictive accuracy, and increasing the constructed models' comprehensibility (Joachims, 1998).

In document classification removing the stopping words is the easiest and the most powerful feature selection / elimination task. Stopping words are the common words of the language that usually do not contribute to the semantics of the documents and have no real added value on classification (Kao and Poteet, 2007). (For example "the, and, a,

an, on, or, from" etc) For Turkish natural language processing an assembled stopping words list is used. Besides this for each document classification problem, some common words or phrases frequently appearing in all kinds of documents for that problem area are also must be added to stopping words list (Berry 2003). For example if we are classifying the military documents the word “captain” may be denoted as a stopping word and be eliminated from features. Also the features which have total count in all documents below a predefined threshold are evaluated as stopping words.

A common metric widely used for text classification feature selection is chi-square statistics. In the domain of text classification chi-square test is applied to test the independence of two random variables; the occurrence of term t and the occurrence of the class c:

$$\chi^2(t,c) = \frac{n.(AD-BC)^2}{(A+C).(B+D).(A+B).(C+D)} \quad \textbf{(2.1)}$$

where A and C denote the number of documents in class c in which term t, respectively, appears and does not appear; B and D denote the number of documents in other classes in which term t, respectively, appears and does not appear. n is the total number of documents (Feldman and Sanger, 2007).

Alternatively some studies use another common feature selection metric: information gain (IG). Information gain is entropy based metric by giving the number of bits of information yield for the prediction of the classes by knowing presence or absence of a term in a document:

$$IG(t) = -\sum_{i=1}^{m} P_r(c_i) * logP_r(c_i) + P_r(t) \sum_{i=1}^{m} P_r(c_i|t) * logP_r(c_i|t) + P_r(\bar{t}) \sum_{i=1}^{m} P_r(c_i|\bar{t}) * logP_r(c_i|\bar{t}) \quad \textbf{(2.2)}$$

where $P_r(c_i)$ is the probability of a document to have class label c_i , $P_r(t)$ is the probability of a term t to appear in a document, $P_r(c_i|t)$ is the probability of document

to have class label c_i given that term t appears in the document and $P_r(c_i|\vec{t})$ is the probability of a document to have class label c_i given that term t does not appear in the document (Han and Kamber, 2007).

Selecting both of these metrics does not reveal an evident accuracy difference. This study prefers the first feature selection metric, chi-square statistics.

2.3. TF-IDF MATRIX

Representation of unstructured data like text documents, voice or image data is one of the most critical preprocessing issues of document classification. Classification algorithms like support vector machines, naïve bayes or decision trees all need structured data to classify. Structured data is anything that has an enforced composition to the atomic data types. But a text document cannot easily be rendered in atomic data types (Han and Kamber, 2007).

Therefore, as a preprocessing task the unstructured documents are converted into more manageable and meaningful structured representation items, feature vectors (Feldman and Sanger, 2007). This feature vectors construct a structured representation matrix. A document is represented as a row vector which holds a sequence of features and their weights in each column. The weights of features can be calculated by several weighting methods. The most common method for assigning weights of features is term frequency / inverse document frequency (TF-IDF) measure (Cooley, 1999). The constructed matrix holding TF-IDF values of features is also called TF-IDF matrix.

TF-IDF value of a feature is a composition of two metrics about the numerical distribution of the feature in document set. Term frequency (TF) value represents the weight of term/feature for a given document. Also inverse document frequency (IDF) of same feature for whole corpus constructs the second part of the weight. For a term t_i let n is the number of documents in whole corpus the TF-IDF value of a term is calculated as:

$$TFIDF(t_i) = TF(t_i) * \log\left(\frac{n}{DF(t_i)}\right) \quad \textbf{(2.3)}$$

Where $TF(t_i)$ is the frequency of appearing term t_i in discussed document and $DF(t_i)$ is the frequency of appearing term t_i in whole corpus of documents. The idea is, if a feature appears in many of documents, it is less likely to be a good measure for distinguishing one document from another (Hsu et al). The discriminative power of a feature effects also distinguishing classes from each other.

3. CLASSIFICATION ALGORITHMS

3.1. SUPPORT VECTOR MACHINES (SVM)

The support vector machine is a supervised learning method introduced by Vapnik (Cortes and Vapnik, 1995). Based on Vapnik's statistical learning theory (Vapnik, 2000) and Structural Risk Minimization principle from computational theory (Cortes and Vapnik, 1995), support vector machines are independent of the dimensionality of the problem (Feldman and Sanger, 2007). Structural risk minimization principle finds a hypothesis *h* which guarantees the lowest true error. The true error discussed here is the probability of false deciding for a randomly selected example. Support vector machines focuses on finding the hypothesis *h* which (approximately) minimizes the true error (Joachims, 1998).

In its simplest linear form, an SVM is an hyperplane that separates a set of positive examples from a set of negative examples with the maximum margin possible which is denoted by $\underset{w}{\rightarrow}$. Let $\underset{x}{\rightarrow}$ be the input vector of the linear SVM, the output is formulated as $u = \underset{w}{\rightarrow} . \underset{x}{\rightarrow} - b$. The margin is defined by the distance of the hyperplane to the nearest of positive and negative examples in the linear case. Maximizing this margin is an optimization problem: minimize $\frac{1}{2} \left\| \underset{w}{\rightarrow} \right\|^2$ subject to $y_i \left(\underset{w}{\rightarrow} . \underset{x}{\rightarrow} - b \right) \geq 1, \forall i$ where x_i is the i th training example and y_i is the correct output which is related to x_i .

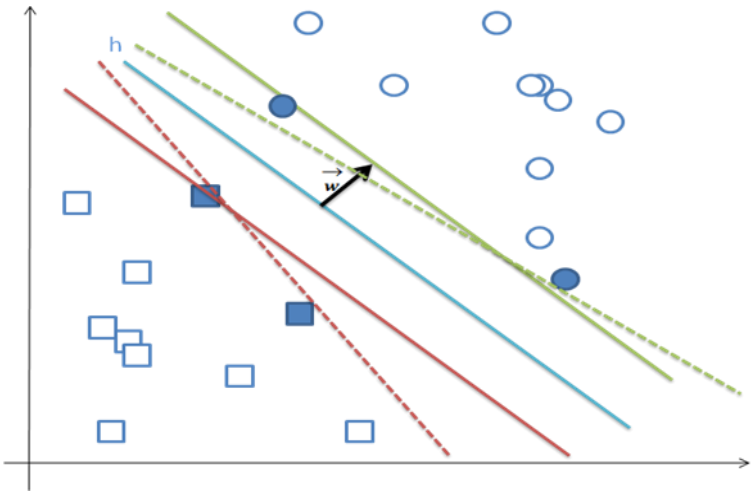


Figure 1. SVM two dimensional representations

Optimal hyperplane is denoted as the one giving the maximum margin between the training examples that are closest to the hyperplane (Cooley, 1999). Figure 1, represents an optimal hyperplane detected by support vector machine for a linearly separable problem. The representation holds squares for negative examples (vectors), circles for positive examples (vectors), green lines for possible borders of positive plane, red line for possible borders of negative plane. The filled squares and circles are support vectors which are closest to hyperplane. Aim of SVM is to find a maximal margin between green and red lines. In this figure, solid red and green lines represent maximal discrimination between positive and negative examples.

Support Vector Machines are widely preferred in text classification problems. The main factors promoting SVM in text classification are:

Text classification problems have high dimensional input spaces. SVM's are over fitting protected classifiers. Dealing with maybe more than 10000 inputs does not reduce the accuracy of support vector machines.

Other classifiers use irrelevant features to reduce the effects of overfitting cases. But text classification problems do not yields many irrelevant features. SVM's can execute with few irrelevant features.

Document vectors are sparse. For each document, the corresponding document vector contains only few entries which are not zero. SVM's can easily work with sparse input vectors.

Text classification problems are linearly separable in general. SVM's are one of the most accurate classifiers for linearly separable problems.

3.2. NAÏVE BAYESIAN CLASSIFIERS

Naive Bayes is the simplest form of Bayesian network, in which all attributes are independent given the value of the class variable. This is called conditional independence. Typically, an example E is represented by a tuple of attribute values $(x_1, x_2, \ldots\ldots, x_n)$, where x_i is the value of attribute X_i. Let C represent the classification variable, and let c be the value of C. In this paper, we assume that there are only two classes: + (the positive class) or - (the negative class).

From the probability perspective, according to Bayes Rule, the probability of an example E = $(x_1, x_2, \ldots\ldots, x_n)$ being class c is:

$$P(c|E) = \frac{P(E|c) * P(c)}{P(E)} \quad \textbf{(3.1)}$$

E is classified as the class C if and only if $\mathrm{P}(\mathrm{C} = +|\mathrm{E}) \geq \mathrm{P}(\mathrm{C} = -|\mathrm{E})$ where $f_{nb}(E)$ is called a Bayesian classifier.

Assume that all attributes are independent given the value of the class variable; that is,

$$P(E|c) = P(x_1, x_2, \ldots\ldots, x_n \mid c) = \prod_{i=1}^{n} P(X_i \mid c) \quad \textbf{(3.2)}$$

the resulting classifier is than,

$$f_{nb}(E) = \frac{P(C=+)}{P(C=-)} \prod_{i=1}^{n} \frac{P(x_i|C=+)}{P(x_i|C=-)} \quad \textbf{(3.3)}$$

The function $f_{nb}(E)$ is called a naive Bayesian classifier (Eyheramendy et. al. 2003).

3.3. ADAPTIVE NEURO-FUZZY INFERENCE SYSTEMS (ANFIS)

The notion of "fuzzy logic" was firstly proposed by Zadeh **Error! Reference source not found.** to describe complicated systems. It has became very popular and been used successfully in various problems especially on control processes such as chemical

reactors, electronic motors, automatic trains and nuclear reactors. More recently, fuzzy logic has been highly recommended for modeling data mining and knowledge engineering problems. Nevertheless, the absence of systematic procedures for the design of a fuzzy system is the main problem with this methodology. On the other hand, a neural network has the ability to learn from the environment (input–output pairs), self-organize its structure, and adapt to it in an interactive manner (Sanver and Karahoca, 2009). Because of this ability of neural networks, we prefer using adaptive neuro-fuzzy inference system for predicting the security level class labels of text documents. Also the fuzzy nature of our classification problem is making it more convenient to use fuzzy inference system. Because the discrimination between class labels in a security level classification problem may be sometimes indiscernible. Fuzzy membership of documents to more than one class is fitting better in this study.

Simply we assume the fuzzy inference system under consideration has two inputs, x and y, and one fuzzy output z within the fuzzy region specified with the fuzzy rule. For a first-order Sugeno fuzzy model, a typical rule set with two fuzzy if–then rules can be expressed as:

Rule 1: If x is A_1 and y is B_1 then $z_1 = p_1 * x + q_1 * y + r_1$

Rule 2: If x is A_2 and y is B_2 then $z_2 = p_2 * x + q_2 * y + r_2$

Where A_i and B_i are the fuzzy sets, p_i, q_i and r_i are the design parameters that are determined

during the training process (Cortes and Vapnik, 1995). The architecture of ANFIS consists of five layers shown in Figure 2. Brief introduction of the model is as follows (Shing and Janj 1993; Güler and Übeyli 2005):

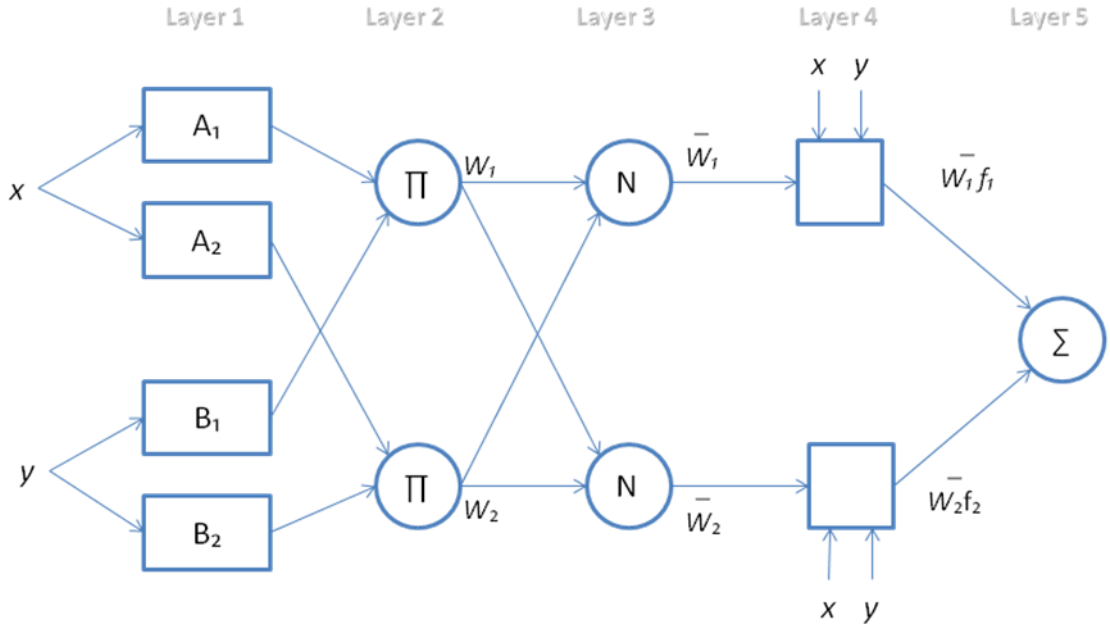


Figure 2. ANFIS structure

Layer 1: Each node of this layer generates membership grades to which they belong to each of the appropriate fuzzy sets using membership functions. All the nodes are adaptive nodes. The outputs of layer 1 are the fuzzy membership grade of the inputs, which are given by:

$$O_{1,i} = \mu_{A_i}(x) \text{ for i = 1 , 2} \quad \textbf{(3.4)}$$

$$O_{1,i} = \mu_{B_{i-2}}(y) \text{ for i = 3 , 4} \quad \textbf{(3.5)}$$

where *x*, *y* are the crisp inputs to node *i*, and A_i, B_i (small, big, etc.) are the linguistic labels characterized by appropriate membership functions μ_{A_i} , μ_{B_i} respectively.

Layer 2: In this layer, the nodes are fixed nodes which labeled with $\prod$. In its simplest form the nodes perform as a simple multiplier having the outputs:

$$O_{2,i} = w_i = \mu_{A_i}(x)\,\mu_{B_i}(y) \quad \text{for i = 1 , 2} \quad \textbf{(3.6)}$$

Layer 3: The main objective is to calculate the ratio of each *i'*th rule's firing strength to the sum of all rules' firing strength. Consequently, $\overline{w_i}$ is taken as the normalized firing strength.

$$O_{3,i} = \overline{w_i} = \frac{w_i}{w_1 + w_2} \text{ for i = 1 , 2} \quad \textbf{(3.7)}$$

Layer 4: Every node in this layer are adaptive nodes with an output function

$$O_{4,i} = \overline{w_i} f_i = \overline{w_i} (p_i x + q_i y + r_i) \text{ for i = 1 , 2} \quad \textbf{(3.8)}$$

Where $\{p_i , q_i , r_i\}$ is the parameter set which contains the consequent parameters. The adaptive learning is performed by regulating these parameters.

Layer 5: This layer consists of one single node which computes the overall output as the summation of all incoming signals as follows:

$$O_{5,1} = overall\ output = \sum_i \overline{w_i} f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \quad \textbf{(3.9)}$$

While designing a new ANFIS structure the number of fuzzy rules, the number of training epochs and the number of membership functions are important factors (Shing and Janj 1996). Over-fitting or not-fitting the data and time complexity problems may occur if the factors mentioned above are not selected appropriately.

4. DISCRETIZATION BASED ON CLASS ATTRIBUTE-CONTINGENCY COEFFICIENT ALGORITHM

In mathematics discretization is used to transfer continuous models into discrete counterparts by partitioning continuous attributes into a finite set of adjacent intervals. The main motivation behind the discretization is generating attributes with a small number of distinct values (Kurgan and Cios, 2004). Assuming that a dataset consisting of *R* examples and *S* target classes, a discretization algorithm would discretize the continuous attribute A in this dataset into n discrete intervals {$[d_0, d_1]$, $(d_1, d_2]$, . . . , $(d_{n-1}, d_n]$}, where d_0 is the minimal value and d_n is the maximal value of attribute *A*. Such a discrete result {$[d_0, d_1]$, $(d_1, d_2]$, . . . , $(d_{n-1}, d_n]$} is called a discretization scheme *D* on attribute *A*. Discretization schemes differ depending the distribution of data. Hence the adaptive discretization schemes are more powerful than static discretization methods (Tsai et. al. 2008).

The acronym CACC stands for the method Class-Attribute Contingency Coefficient. Contingency coefficient is a parameter widely used by researchers to measure the dependence between the variables. It is formulated as follows:

$$C = \sqrt{\frac{y}{y+M}} \qquad \textbf{(4.1)}$$

Where $y = M[\left(\sum_{i=1}^{S}\sum_{r=1}^{n}\frac{{q_{ir}}^2}{M_{i+}M_{+r}}\right) - 1]$, *M* is the total number of samples, *n* is the number of intervals, q_{ir} is the number of samples within the class *i* and the interval of *r-1* to *r*, M_{i+} is the number of samples belongs to class *i*, M_{+r} is the number of samples between *r-1* and *r* thresholds.

CACC methods suggests a new form of contingency coefficient by dividing y by *log(n)*. The new *c* value is represented as *cacc*:

$$cacc = \sqrt{\frac{y}{y+M'}} \qquad \textbf{(4.2)}$$

Where $y = M\left[\left(\sum_{i=1}^{S}\sum_{r=1}^{n}\frac{q_{ir}{}^{2}}{M_{i+}M_{+r}}\right) - 1\right]$ / log(n)

In Table 3 the pseudo-code representation of CACC algorithm is presented (Tsai et. al. 2008):

Table 3. CACC Discretization Algorithm

1 Input: Dataset with i continuous attribute, M examples and S target classes;
2 Begin
3 For each continuous attribute Ai
4 Find the maximum dn and the minimum d0 values of Ai;
5 Form a set of all distinct values of A in ascending order;
6 Initialize all possible interval boundaries B with the minimum and maximum
7 Calculate the midpoints of all the adjacent pairs in the set;
8 Set the initial discretization scheme as D: {[d0,dn]}and Globalcacc = 0;
9 Initialize k = 1;
10 For each inner boundary B which is not already in scheme D,
11 Add it into D;
12 Calculate the corresponding cacc value;
13 Pick up the scheme D’ with the highest cacc value;
14 If cacc > Globalcacc or k < S then
15 Replace D with D’;
16 Globalcacc = cacc;
17 k = k + 1;
18 Goto Line 10;
18 Else
19 D’ = D;
20 End If
21 Output the Discretization scheme D’ with k intervals for continuous attribute Ai;
22 End

In this study discretization is necessary to find out the correct class labels of documents after ANFIS classification. Because ANFIS classification gives us a fuzzy classification rate; for example “1.24”. This rate denotes that the document belongs a little bit to the class 2 and much more to the class 1. The discretization thresholds will be defined after the CACC discretization learning on train data.

5. EXPERIMENTAL SETTINGS

This study aims to develop a framework which has an ability to classify 222 internal documents of TUBITAK UEKAE (National Research Institute of Electronics and Cryptology) according to their security levels by using adaptive neuro-fuzzy inference systems, support vector machines and naïve bayes algorithms.

First, all of 222 internal documents are classified into correct security levels (secret, restricted, unclassified) according to the general policies of TUBITAK UEKAE with the help of a subject matter expert. (The numbers of secret, restricted and unclassified documents are 30, 165 and 27 respectively.) Then these classified documents are converted into UTF-8 encoded txt based file format. Training and test documents have totally about 2.5 millions of words except stopping words. All the documents are grouped and arranged in a relational database structure by using the open source database PostgreSQL.

These 2.5 million words are unstemmed Turkish words. A comprehensive stemmer library zemberek (zemberek 2001) is used to find out roots of unstemmed words. Zemberek gives us all the possible stemming structures for a term. Our stemming system selects the structure that has the biggest probability of semantic and morphologic patterns of the Turkish language. Stemming examples of Zemberek were given in section 2.1 in Table 1.

Performing the stemming process we obtained approximately 9000 distinct terms. These 9000 terms may cause a high dimensionality and a time consuming classification process. As we mentioned above SVM's are able to handle high dimensional TF-IDF matrix with the same accuracy; however the time complexity of the classification problem is also an important aspect of our study. Therefore a x^2 statistics with a threshold (100) was performed to select important features of classification. By the feature selection, the size of selected features is reduced from ~9000 to ~2000. This is known as the corpus of classification process.

The final task performed for the preprocessing phase was constructing a TF-IDF value matrix of all the features for all documents. The application was calculating TF-IDF values for each feature-document pair in the corpus.

Preprocessing software of this study was developed in JAVA, an open-source powerful software development language, and data were stored on PostgreSQL, an open-source professional database.

6. RESULTS

6.1. CROSS VALIDATIONS

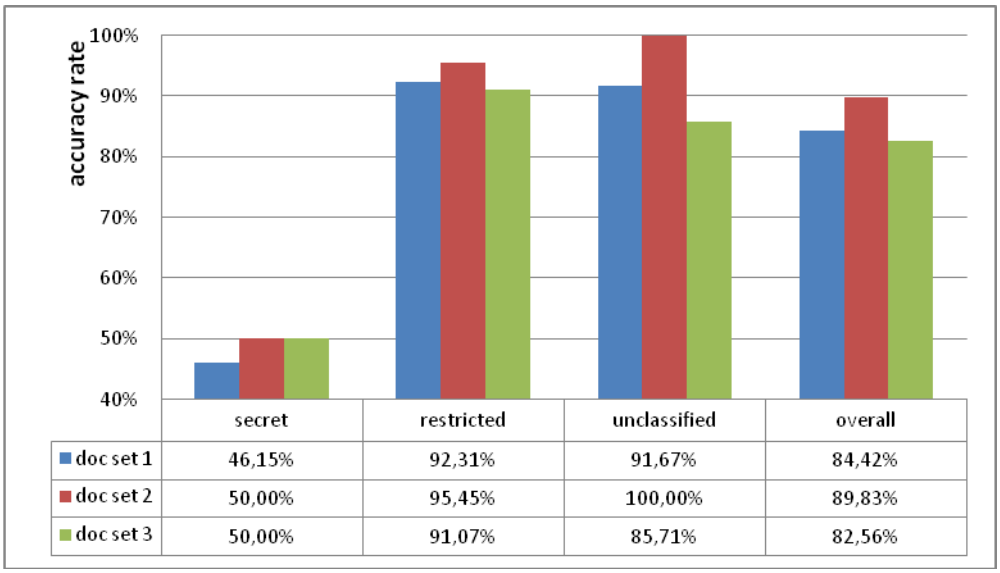
After the preprocessing tasks of data has completed some well known classification algorithms are applied on train and test document sets to compare the accuracy rates of algorithms and to construct the best fitting classification framework for the security level classification problem.

A leave-one-out cross validation was performed to confirm classification accuracy and parameter selection. All the documents have been grouped randomly into 3 train/test set pairs as in Table 4.

Table 4.Test sets applied to SVM and NB algorithms

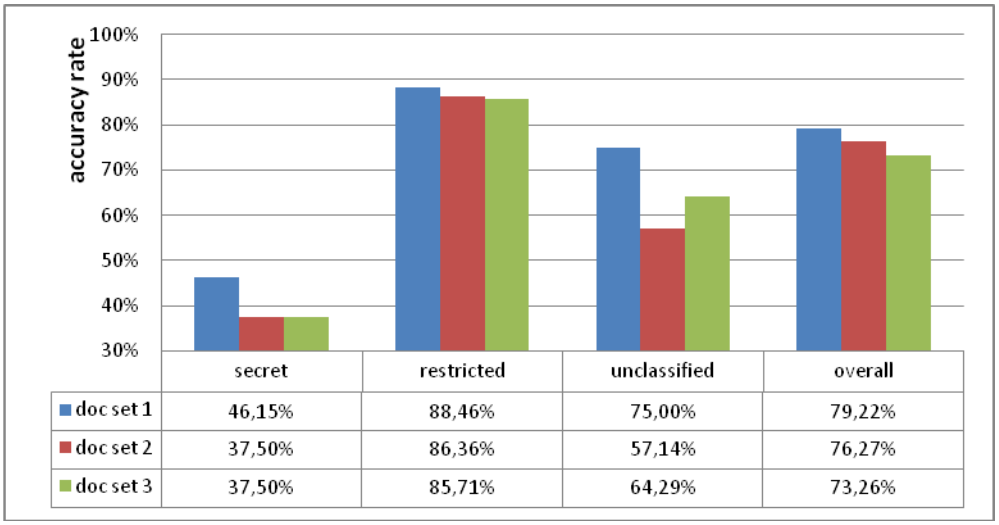
	doc set 1	doc set 2	doc set 3
number of train docs	145	163	136
number of test docs	77	59	86
total	222	222	222

Support vector machines and naïve bayes algorithms were performed on all 3 document sets which were randomly selected from 222 documents. Accuracy rates of both naïve bayes and support vector algorithms for 3 document sets are similar as shown in Figure 3 and Figure 4. Hence explaining the results of only one document set (ex doc set 2) may supply us the overall inferences for all 3 train/test pairs.



	secret	restricted	unclassified	overall
doc set 1	46,15%	92,31%	91,67%	84,42%
doc set 2	50,00%	95,45%	100,00%	89,83%
doc set 3	50,00%	91,07%	85,71%	82,56%

Figure 3. SVM results for different document sets



	secret	restricted	unclassified	overall
doc set 1	46,15%	88,46%	75,00%	79,22%
doc set 2	37,50%	86,36%	57,14%	76,27%
doc set 3	37,50%	85,71%	64,29%	73,26%

Figure 4. NB results for different document sets

Regarding the cross validation results it is obvious to see that selecting different train and test pairs does not affect the overall and partial accuracy rates of the system. Next sections will introduce the classification results of naïve bayes classification, support vector classification and a hybrid classification framework based on SVM and adaptive neuro-fuzzy inference systems.

6.2. NAÏVE BAYES CLASSIFICATION

In the first try naïve bayes classification is performed on textual data by using WEKA data mining and knowledge discovery software. With the bag of words representation of data naïve bayes classifier yields the results below:

Table 5. Naive Bayes classification results

Correctly Classified Instances	45/59 (76.27%)
Incorrectly Classified Instances	14/59 (23.73%)
Kappa Statistics	0.4083
Mean Absolute Error	0.1582
Total Number of Instances	59

These results can be detailed in class level as shown in the Table 6. Rows represent the actual/real class labels of documents and columns refer to the given class labels by classification algorithms on document set 2. For example in Table 6 we infer that document set 2 provides 8 secret documents. (3+4+1) And SVM classifier assigned 3 of them as secret, 4 as restricted and 1 as unclassified. Hence the accuracy rate for secret documents in document set 2 is 42.8%. (3/8)

Table 6. Naive Bayes classification accuracy rates

Predicted / Actual	Secret	Restricted	Unclassified	ACCR
Secret	3	4	0	42,8%
Restricted	4	38	2	86,3%
Unclassified	0	3	4	57,1%
Overall Accuracy Rate:	76,27% (45/59)			

The accuracy rates obtained from naïve bayes classification is not satisfactory at all.

6.3. SUPPORT VECTOR CLASSIFICATION

Support Vector Classifiers are very powerful and accurate classifiers one can easily use for document classification. Joachim's new classifier implementation SVM-multiclass

was used to classify the security level of internal Turkish documents. The preprocessing tasks are performed to produce the input TF-IDF matrix of SVM.

Parameter c of SVM, the trade-off between training error and margin (Cortes and Vapnik 1995) is defined as a high value like 1000. As shown in figure 1, classification accuracy varies respect to the parameter c. The best fitting c value for all these 3 document sets is 1000.

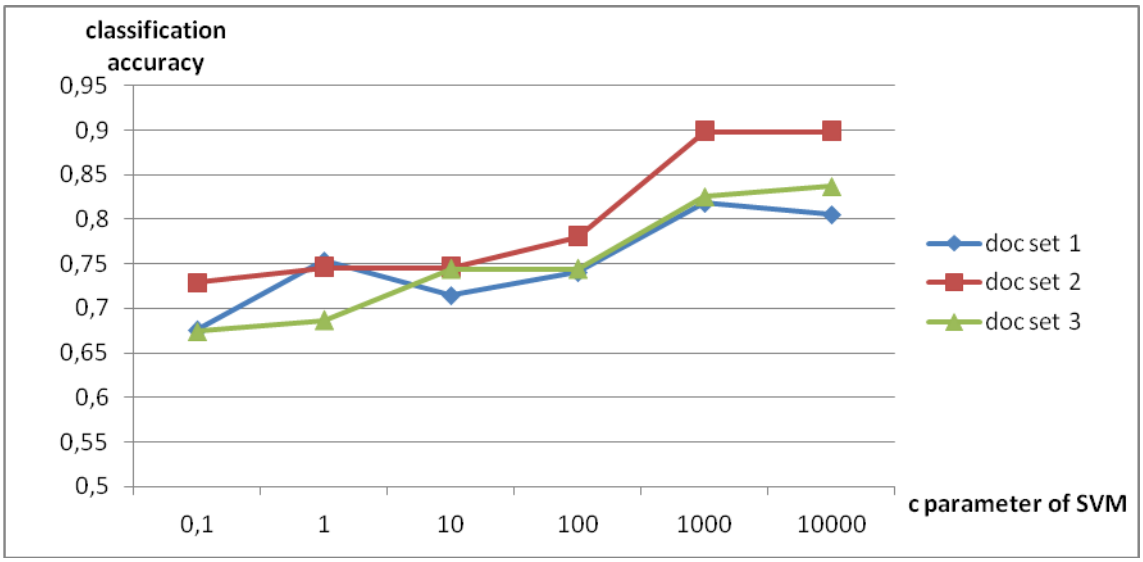


Figure 5. Classification accuracy respect to the parameter c

Setting the parameter c to the value "1000" the multiclass support vector classifier gives the results shown in Table 7:

Table 7. Support Vector classification accuracy rates

Predicted / Actual	Secret	Restricted	Unclassified	ACCR
Secret	4	3	1	50%
Restricted	0	42	2	95,5%
Unclassified	0	0	7	100%
Overall Accuracy Rate:	89,83% (53/59)			

This study also aims to state a relation between class labels and sub-class labels implicating parent label. Sub-classification areas like document type, area or format are other distinctive properties of documents. Detecting class labels of internal documents of an organization depends on some interaction rules of sub-classes. For example, document area (military, private sector, government etc...) and document type (tech test report, tech guide, meeting report, quality procedure, ITSM audit etc...) are sub-classes of security level classification (secret, restricted, unclassified) of TUBITAK UEKAE.

Some of rule based subclass – class interactions can be defined as:

If area: **military** and type: **spec. document** then level: **secret**

If area: **military** and type: **procedure** then level: **restricted**

If area: **government** and type: **travel** then level: **restricted**

If area: **general** and type: **tech. guide** then level: **unclassified**

These rules, security labels and the distribution of documents into subclasses are listed below in Table 8:

Table 8. Subclass - security level interaction rules

SECURİTY LEVEL	TYPE	AREA	TRAİN DOK #	TEST DOK #
secret	tech test report	military	*12*	*3*
secret	tech guide	military	*5*	*2*
secret	spec doc	military	*7*	*3*
restricted	quality procedure	military	*9*	*3*
restricted	meeting report	military	*6*	*3*
restricted	ITSM audit	government	*6*	*3*
restricted	tech test report	government	*12*	*7*
restricted	travel report	general	*12*	*4*
restricted	tech test report	private sector	*20*	*5*
restricted	training	general	*14*	*6*
restricted	test procedure	general	*12*	*4*
restricted	meeting report	general	*28*	*9*
unclassified	ITSM audit	general	*8*	*3*
unclassified	tech guide	general	*12*	*4*
			163	***59***

All of 59 test documents are classified with support vector machines according to their area (military, private company, government, general etc.) and their type (study report, travel report, meeting report, procedure etc.) respectively. The results of two SVM sub-classifications are merged for each test document according to the subclass-class interaction rules mentioned above. Finally we obtained a success matrix as follows:

Table 9. Subclass - class interaction methodology accuracy rates

Actual \ Predicted	Secret	Restricted	Unclassified	ACCR
Secret	3	3	2	37,5%
Restricted	0	42	2	95,5%
Unclassified	0	1	6	85,7%
Overall Accuracy Rate:	76,27% (45/59)			

As shown in Table 9 above, this document set offers a lower accuracy level for sub-classified solution than in the conventional solutions.

6.4. A HYBRID APPROACH: SVM AIDED ADAPTIVE NEURO-FUZZY INFERENCE CLASSIFICATION

Adaptive Neuro-Fuzzy Inference Systems (ANFIS) can be widely used in classification of structured data. The algorithm merges the power of a neural network algorithm with the efficiency of a fuzzy inference system. The problem of security level classification which is subject to our study, implies fuzzy class labels. By the nature, security level classification does not yield distinct and discrete security labels. For example a document labeled as “restricted” can be quite (30%) “unclassified” and more obviously (70%) “restricted”. So we label this document as “restricted”.

Thus, using ANFIS to classify security levels of documents is more suitable because of the continuous outputs of ANFIS. Having these continuous outputs a supervised discretization algorithm (CACC) can be used to detect discrete class labels of documents.

In the other hand the nature of ANFIS algorithm is not suitable to handle thousands of input parameters. So a taxonomy based pre-classification has to be performed to provide input to the ANFIS algorithm.

Regarding Table 8 a sub-level classification according to the document types (tech test report, tech guide, meeting report, quality procedure, ITSM audit etc...) is performed by using Support Vector Machines. There exist 9 different document types. SVM-multiclass scores all train and test documents according to their document types. Example SVM-multiclass scoring outputs for the first 20 documents are represented in Table 10 below:

Table 10. SVM scores for document type subclassification

TECH TEST REPORT	TECH GUİDE	SPEC DOC	QUALİTY PROCEDURE	MEETİNG REPORT	ITSM AUDİT	TRAVEL REPORT	TRANİNG	TEST PROCEDURE
9,362026	18,85194	-0,99012	-5,47238	-8,29389	-17,0853	-11,7173	48,87856	-17,614
24,28767	-17,9092	-33,8235	-13,8527	23,23329	-24,2238	38,07242	63,16465	-24,3409
-33,8924	-3,2924	13,78726	-17,0198	-14,2971	-19,9289	-8,8031	133,9987	-5,26039
8,438817	10,85226	-21,8353	-11,4376	2,460903	-30,7703	-5,59678	48,23171	20,76673
-7,92581	12,82986	-5,2941	-6,54043	-9,65095	-9,89681	-5,28716	48,52536	6,264983
-10,7447	-13,6576	-25,0162	-5,61372	11,70897	17,64177	-11,8964	62,33111	-1,86947
-18,1712	-13,8135	89,70103	-11,1795	4,822606	-20,1877	-36,8179	4,412669	-6,94369
17,92289	222,8821	15,93745	-18,6337	-9,60963	-117,71	-21,753	-8,10062	55,24706
2,140784	-0,60173	27,93732	-5,34607	12,42788	-11,9991	-13,8988	-3,51731	-8,55906
-5,38877	-6,39164	89,09076	-6,25006	-8,02262	-15,7163	-7,99134	-5,27228	-15,898
-1,60448	-22,5206	17,26676	-10,1695	19,83477	-7,99746	25,86286	-14,0172	-16,3715
112,392	55,82467	-11,9324	-28,7421	-28,932	-43,9447	-11,8435	-6,81128	-17,1679
106,5034	57,39332	-11,4532	-26,195	-27,1893	-39,7843	-14,126	-4,42984	-18,8355
100,1669	65,34735	-14,5173	-24,4422	-27,7179	-41,6507	-14,6098	-0,30571	-18,3752
-8,168	-88,1632	-26,5255	65,94036	132,0042	-7,47805	-2,56754	-19,1909	-55,5017
-10,8443	-18,586	-7,60051	93,99776	22,12609	-14,1754	-8,04292	-19,3261	-16,3951
-6,14894	-48,8829	-43,4092	114,9401	79,56547	-20,1475	-8,05776	-16,7943	-9,93762
-15,0721	-8,15236	5,390152	-9,70539	61,41209	-10,2196	-9,51811	-5,5914	-3,64481
-6,06617	-7,97041	0,072938	-7,13568	49,88315	-9,85922	-5,36499	0,083126	-4,41659
-38,06	-156,015	-215,949	-88,2234	569,6136	-67,5499	205,1715	-97,8557	-26,4203

These document type scores are the inputs of Adaptive Neuro-Fuzzy Inference System (ANFIS). ANFIS can handle less than 10 or 15 inputs because of the time complexity of the algorithm. MATLAB interface of fuzzy logic toolbox is used to predict security

levels of documents regarding the types of documents. This means the input variables of ANFIS are the document type scores represented in Table 10 and the outputs are continuous variables indicating security levels. Constructed FIS structure can be shown in Figure 6 below:

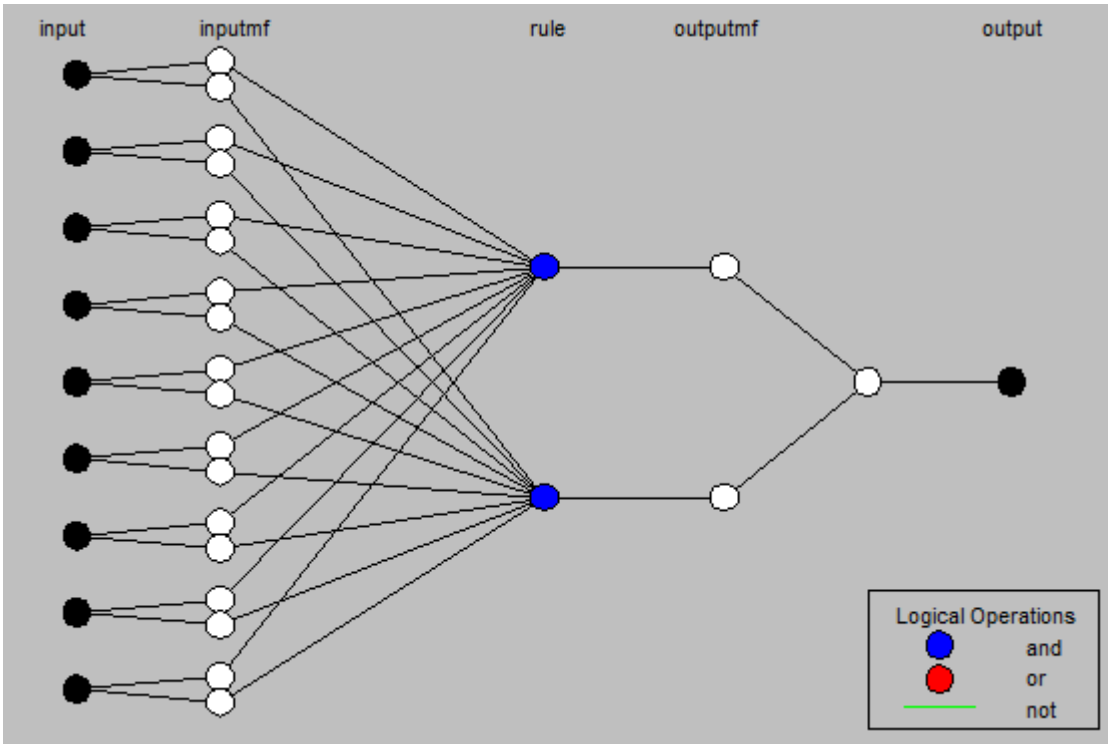


Figure 6. FIS structured used in security level classification

In this fuzzy inference structure there exist 2 membership functions for each input and also 2 membership functions for output. Increasing the number of membership functions may affect the ability of neuro-fuzzy training because of the time complexity. 2 membership functions with trimf input membership type and constant output membership type provide the most appropriate, fitting FIS structure. The resulting output of ANFIS is shown in Figure 7 below:

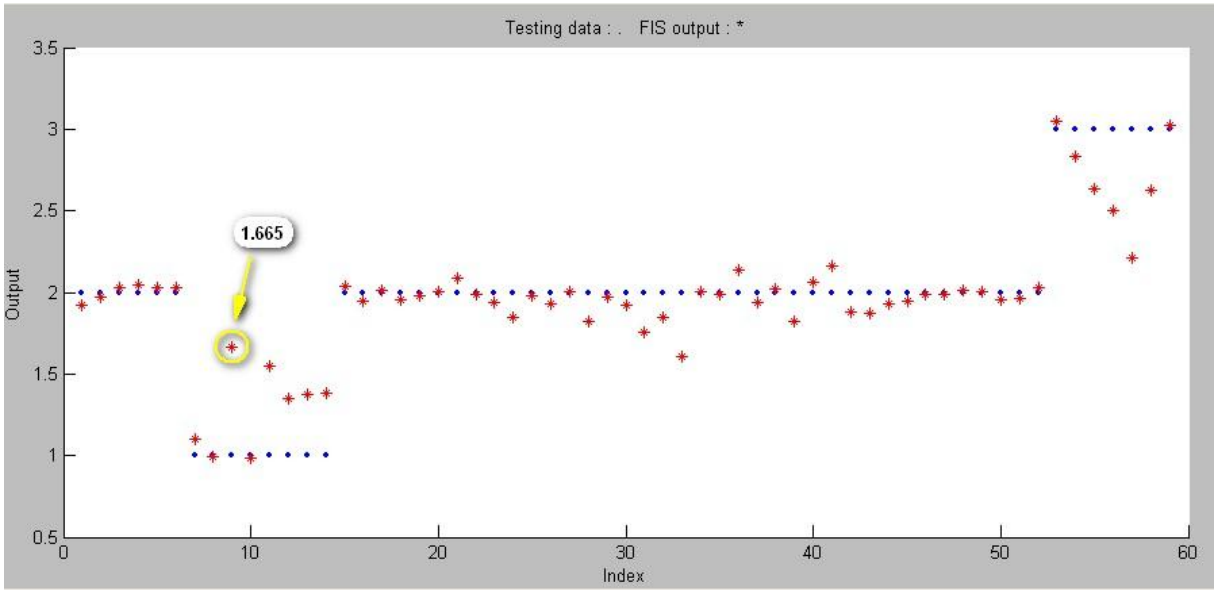


Figure 7. Continuous FIS outputs of security level classification

In this figure each dot represents a test document. Blue ones and red ones represent the real security label of documents and predicted security level of documents respectively. It is noticeable that many of the predicted values are scored between two class labels. For example the 9th sample introduces a score of 1.665 which is between 1 (secret) and 2 (restricted).

To find out the corresponding security label of this level score we have to discretize the continuous outputs to the discrete labels. After using the CACC algorithm introduced in section 0 the discrete security labels are detected. CACC algorithm suggests these thresholds for discretization:

If the ANFIS score of document is smaller than **1.68** the security label must be **1 (secret).**

If the ANFIS score of document is between **1.68** and **2.36** the security label must be **2 (restricted).**

If the ANFIS score of document is bigger than **2.36** the security label must be **3 (unclassified).**

The resulting scheme is summarized in Table 11:

Table 11. SVM-ANFIS hybrid classification accuracy rates

Predicted / Actual	Secret	Restricted	Unclassified	ACCR
Secret	8	0	0	100%
Restricted	1	43	0	97,6%
Unclassified	0	1	6	85,7%
Overall Accuracy Rate:	96,67% (57/59)			

7. DISCUSSION AND CONCLUSION

This study classifies internal Turkish documents of TUBITAK UEKAE (a military-governmental organization) according to their security levels by using 3 different approaches.

The first algorithm, naïve bayes classifier is not offering an accurate classification result with 76% of success. We believe that this inefficacy may be a result of the fuzzy nature of the security level classification and the interdependence relations between the features. In the other hand this result may be increased by using other bayesian neural algorithms.

Support vector classifiers are very standardized and ordinary classifiers for document classification. Many of the previous studies use support vector machines as a document classifier with TF-IDF values. We have also applied support vector classifiers to our security level detection problem. An accuracy rate of 90% has been reached by simple SVM methods and standard linear kernels. Besides we have developed a logic of type and area sub-classification based security level classification. Our expectation from this classification model has been resulted with a low accuracy rate, 76%.

In the third approach we have developed a hybrid solution consists of support vector machines and adaptive neuro-fuzzy inference systems. Documents are classified according to their types (tech test report, meeting report etc…) by using SVM. The outputs of SVM are also the inputs of ANFIS which is giving the security level rates of documents as output. These rates are rendered discretize by using a discretization algorithm, CACC. The accuracy rate of this constructed framework is very satisfied with 96%.

Figure 8 represents the Receiver Operating Characteristic (ROC) curve pointing out the true positive accuracy by altering the threshold levels of classification. The ROC curve indicates how the detection rate changes as the thresholds are varied to generate more or fewer false alarms. For the class label “secret” the ROC curves are defined as follows:

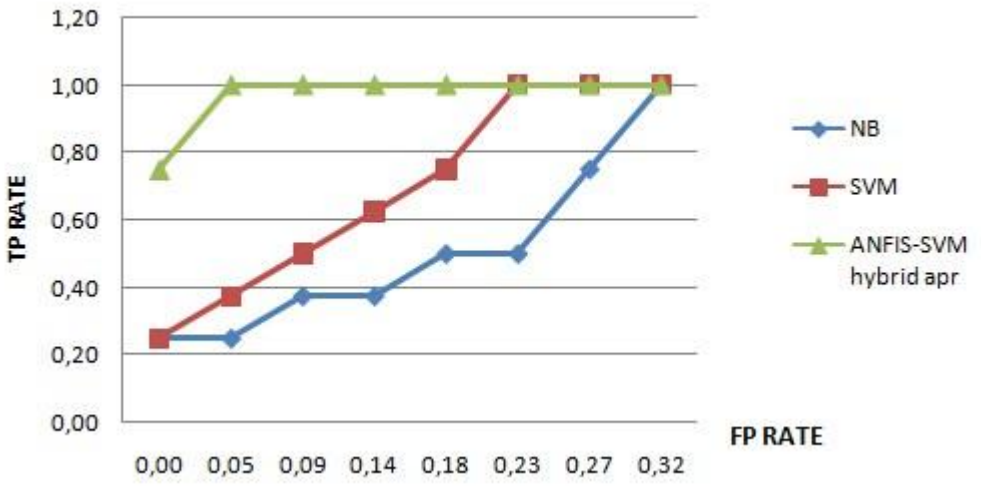


Figure 8. ROC curves for classification algorithms

Illustrated in Figure 8 the hybrid approach consisting of ANFIS and SVM outperforms the other two classification frameworks.

As we mentioned before, security level classification can constitute a basis for the extended detection capabilities of data loss / leakage prevention solutions. For each security problem, according to the nature of the business processes of documents for each organization, support vector phase of our hybrid approach may be reorganized. In TUBITAK UEKAE document types are the essential indicators to detect the security level. Therefore this study uses document types as support vector outputs and ANFIS inputs.

REFERENCES

Ageev M. & Dobrov V., 2003. *Support Vector Machine Parameter Optimization for Text Categorization*. International Conference on Information Systems Technology and its Applications

Alparslan E., Bahsi B. and Karahoca A., 2009. *Classification of Turkish News Documents Using Support Vector Machines*. International Symposium on Innovations in Intelligent Systems and Applications

Berry M., 2003. *Survey of text mining. Clustering, classification and retrieval*. New York: Springer

Cooley R., 1999. *Classification of News Stories Using Support Vector Machines.* IJCAI Workshop on Text Mining

Cortes C. & Vapnik V., 1995. *Support-vector Networks.* Machine Learning. pp 20:273-297, November

Eyheramendy S., Lewis D. and Madigan D., 2003. *On the Naive Bayes Model for Text Categorization*.

Feldman R. & Sanger J., 2007. *Text mining handbook.* Cambridge: Cambridge University Press

Güler İ. & Übeyli E., 2005. Adaptive *neuro-fuzzy inference system for classification of EEG signals using wavelet coefficients*. Journal of Neuroscience Methods vol:148 pp: 113–121

Han J.W. & Kamber M., 2007. *Data mining concept and techniques*. Second Edition. San Francisco: Elsevier

Hsu CW., Chang C. and Lin C., 2010. *A Practical Guide to Support Vector Classification*.

Joachims T., 1998. *Text Categorization with Support Vector Machines: Learning with Many Relevant Features*. European Conference on Machine Learning

Kao A. & Poteet S., 2007. *Natural language processing and text mining*. New York: Springer

Kurgan L. & Cios K., 2004. *CAIM Discretization Algorithm*. IEEE Transactions on knowledge and data engineering, vol. 16, no. 2

Prado H. & Ferneda E., 2007. *Emerging technologies of text mining*. New York: Information Science Reference.

Salton G. & McGill M.J., 1983. *The SMART and SIRE experimental retrieval systems.* New York: Mc-Graw Hill.

Sanver M. & Karahoca A., 2009. *Fraud Detection Using an Adaptive Neuro-Fuzzy Inference System in Mobile Telecommunication Networks*. Journal of Multiple-Valued Logic and Soft Computing 15 (2-3), pp. 155-179

Sever H. & Bitirim Y., 2003. FindStem: *Analysis and Evaluation of a Turkish Stemming Algorithm*. Lecture Notes in Computer Science

Shing J. & Janj R., 1993. *ANFIS: Adaptive Network-Based Fuzzy Inference System*. IEEE transactions on systems, man, and cybernetics, Vol. 23

Shing J. & Janj R., 1996. *Input Selection for ANFIS Learning*. IEEE International conference on fuzzy systems

Tsai C.J., Lee C.I. and Yang W.P., 2008. *A discretization algorithm based on Class-Attribute Contingency Coefficient*. Information Sciences vol:178 pp: 714–731

Vapnik V., 2000. *The Nature of Statistical Learning Theory.* Second Edition. New York: Springer

Zadeh L. & Kacprztk J., 1999. *Computing with words in information intelligent systems*. New York: Springer

Zemberek Turkish stemming project. 2001 http://code.google.com/p/zemberek/

CURRICULUM VITAE

Name-Surname: Erdem Alparslan

Address: Karanfil Sokak 6 / 1 Evren Apt. Göztepe Kadıköy İstanbul

Birth Place / Date : Aydın / 20.04.1983

Foreign Language : English - French

Primary Education : Adnan Menderes Anadolu Lisesi - 1995

Secondary Education : İzmir Fen Lisesi - 2001

Bachelor's Education : Galatasaray Üniversitesi - 2006

Institute : Institute of Science

Programme: Computer Engineering

Publications:

E. Alparslan, H. Bahşi, "Security Level Classification of Turkish Documents", Mining User-Centric Data for Security, 2009

E. Alparslan, H. Bahşi, A. Karahoca, "Classification of Turkish News Documents Using Support Vector Machines", International Symposium on Innovations in Intelligent Systems and Applications, 2009

E. Beydağlı, M. Kara, H. Bahşi, E. Alparslan, "Güvenli Yazılım Geliştirme Modelleri ve Ortak Kriterler Standardı", UYMS 2009

E. Alparslan, A. Karahoca, D. Karahoca, H. Uzunboylu, Z. Özçınar, "Gender Differences Between Theoretical and Practical Achievements of CS Students", Procedia - Social and Behavioral Sciences, Volume 2 Issue 2, pp. 5788-5792

Work Experince :

May. 2008 – Present	Researcher, National Research Institute of Electronics and Cryptology (UEKAE)
July. 2006 – May. 2008	Software Specialist, TURKCELL Communication Services