Adjusting Fuzzy Automata for String Similarity Meassuring

J.J. Astrain
Dpt. Matemática e Informática
Universidad Pública de Navarra
Pamplona, Spain
josej.astrain@unavarra.es

J. Villadangos
Dpt. Automática y Computación
Universidad Pública de Navarra
Pamplona, Spain
jesusv@unavarra.es

J.R. González de Mendívil
Dpt. Matemática e Informática
Universidad Pública de Navarra
Pamplona, Spain
mendivil@unavarra.es

J.R. Garitagoitia
Dpt. Matemática e Informática
Universidad Pública de Navarra
Pamplona, Spain
joserra@unavarra.es

F. Farina
Dpt. Matemática e Informática
Universidad Pública de Navarra
Pamplona, Spain
fitxi@unavarra.es

Abstract

In this paper[1], we introduce a fuzzy automaton for computing the similarity between pairs of strings and a genetic method for adjusting its parameters. The fuzzy automaton models the edit operations needed to transform any string into another one. The selection of appropriate fuzzy operations and fuzzy membership values for the transitions leads to improve the system performance for a particular application.

Keywords: fuzzy automata, genetic algorithms, string similarity.

1 Introduction

The applications of pattern recognition based on structural and syntactic methods have to cope with the problem of recognizing strings of symbols that do not fit with any one of the defined patterns. The problem is usually resolved by defining a function that allows measuring the similarity (distance) between pairs of strings [1].

Similarity (distance) measures are defined by assigning a cost (edit distance) to each possible edit operation that allows transforming an input string into a pattern string. The elementary edit operations usually considered are the deletion, insertion and substitution of symbols.

Several variants of the string edit distance have been proposed in the literature [2, 3, 4, 5]. However, in order to use these measures, one has to choose previously an adequate cost matrix, which specifies the costs of individual elementary operations for all combination of symbols. This selection may have a considerable impact on the recognition performance. Usually, the cost matrix is determined heuristically during the system development by trial and error.

In [6], we have introduced a new fuzzy method for computing the similarity between two strings. In our method the similarity between two strings, α and ω, is the membership value of α in a fuzzy language associated to ω. Such a language is generated by a fuzzy automaton whose transitions model every possible edit operations for matching an arbitrary input string to the target string ω [7, 8]. The elementary edit operations have associated fuzzy values which allow determining the effect of such edit operations. The fuzzy transitions of the automaton can be calculated using different t-conorms and t-norms [9, 10].

We have shown in [6] that our method is able to compute different string edit distances by selecting appropriate t-conorms and t-norms. Moreover, when the fuzzy automaton uses a parametric t-conorm/t-norm, it is possible to modify the final classification result by simply varying the t-conorm/t-norm parameter. Therefore, our fuzzy method presents more capability of being adapted to particular problems that previous ones.

Given a particular string classification problem we have to choose adequately the fuzzy values of the elementary edit operations as well as the values of

[1]This work has been partially supported by the government of Navarra and the spanish government under the research grant TIC99-0280-CO2

the t-conorm/t-norm parameter. The problem of selecting such parameters is an optimization problem. Common optimizing techniques (like gradient descendent) cannot be used here because the function that we want to optimize, which depends on the fuzzy automaton, is not differentiable. In this extended abstract we propose to tuning the parameters of the fuzzy automaton following a genetic algorithm [11]. To illustrate the advantages of our proposal, we provide an experiment for a lexicon driven word recognition system.

2 Fuzzy String Similarity

In this section, we introduce a fuzzy automaton that allows to compute the similarity between two strings. Let Σ be a finite set of symbols (alphabet) and Σ^* be the set of all strings over Σ. Let $\alpha \in \Sigma^*$ and $\omega \in \Sigma^*$, $\alpha = x_1 x_2 \dots x_m$ and $\omega = a_1 a_2 \dots a_n$, be two arbitrary strings. In the following, α will be called the *observed string*, and ω the *pattern*. The proposed method begins by defining, for $\omega \in \Sigma^*$, a finite deterministic automaton [12] which accepts (recognizes) it. The automaton $M(\omega) = (Q, \Sigma, \delta, q_0, \{q_n\})$ is defined as follows: $Q = \{q_0, q_1, \dots, q_n\}$ is the set of states; Σ is the alphabet; q_0 and q_n are the start and final states respectively; the transition function $\delta : Q \times Q \times \Sigma \longrightarrow \{0,1\}$ is defined as $\delta(q_{k-1}, q_k, a_k) = 1$ $(1 \leq k \leq n)$ where a_k is the k-th symbol in the string ω.

In order to measure the similarity between α and ω, we need to modify $M(\omega)$ in the sense that it accepts every observed string providing a value for such a similarity. For that purpose a fuzzy automaton [13], based on the automaton $M(\omega)$, denoted by $MF(\omega)$ is introduced. This fuzzy automaton models every possible insertion, deletion or substitution operations when it carries out the matching between the observed string α and the pattern string ω. The fuzzy automaton $MF(\omega) = (Q, \Sigma, \mu, \sigma, \eta)$ is defined as follows: Q and Σ are the same sets that were introduced in $M(\omega)$ definition; $\sigma : Q \longrightarrow [0,1]$ is the start state, where $\sigma(q_0) = 1$ and $\sigma(q_k) = 0$ $(0 < k \leq n)$; $\eta : Q \longrightarrow [0,1]$ is the final state, where $\eta(q_n) = 1$ and $\eta(q_k) = 0$ $(0 \leq k < n)$; the fuzzy transition function $\mu : Q \times Q \times (\Sigma \cup \{\varepsilon\}) \longrightarrow [0,1]$ is defined by the following procedure

(i) If $\delta(q,p,a) = 1$, $q,p \in Q$, $a \in \Sigma$, then
   (i.a) $\mu(q,p,\varepsilon) \in [0,1]$ (insertion of a)
   (i.b) $\forall x \in \Sigma$: $\mu(q,p,x) \in [0,1]$ (substitution of x by a)

(ii) $\forall q \in Q$, $\forall x \in \Sigma$: $\mu(q,q,x) \in [0,1]$ (deletion of x)

(iii) The rest of fuzzy transitions values are 0.

The fuzzy automaton deals with string of symbols and the fuzzy states they induce. We now introduce a function that represents the transition from fuzzy state $\tilde{P}$ to a new state $\hat{\mu}(\tilde{P}, x)$ induced by the symbol x contained in the observed string α. Let $\mathcal{F}(Q)$ be the possible fuzzy sets in Q. The function $\hat{\mu} : \mathcal{F}(Q) \times \Sigma \longrightarrow \mathcal{F}(Q)$ is defined as:
$\hat{\mu}(\tilde{P}, x) = \{(p,\mu) \mid \mu = \oplus_{q \in Q}(\mu(q,p,x) \otimes \mu_{\tilde{P}}(q)), p \in Q\}$, with $\tilde{P}$ in $\mathcal{F}(Q), x \in \Sigma$ (1)
where the operators $\oplus$ and $\otimes$ denote a t-conorm and a t-norm respectively [9, 10].

In addition, the fuzzy automaton has to make transitions by empty strings. We also introduce a function $\mu^\varepsilon : \mathcal{F}(Q) \longrightarrow \mathcal{F}(Q)$ which represents the transition from fuzzy state $\tilde{P}$ to a new state $\mu^\varepsilon(\tilde{P})$ induced by the empty string ε. The function $\mu^\varepsilon(\tilde{P})$, with $\tilde{P}$ in $\mathcal{F}(Q)$, is defined as:

$\mu^\varepsilon(\tilde{P}) = \{(p,\mu) \mid \mu = \mu_{\tilde{P}}(p) \oplus (\oplus_{q \in Q}(\mu_{\tilde{E}(q)}(p) \otimes \mu_{\tilde{P}}(q))), p \in Q\}$, with $\tilde{P}$ in $\mathcal{F}(Q)$ (2)

where $\tilde{E}(q)$ is the fuzzy set in Q representing the reachable set of states from q by repeatedly using transitions by empty string.

Finally, by combining the equation 1 and 2, we can provide a function that determines the behavior of the fuzzy automaton for any observed string. The function $\mu^* : \mathcal{F}(Q) \times \Sigma^* \longrightarrow \mathcal{F}(Q)$ is defined as follows:

(i) $\mu^*(\tilde{P}, \varepsilon) = \mu^\varepsilon(\tilde{P})$, $\tilde{P}$ in $\mathcal{F}(Q)$, and (ii) $\mu^*(\tilde{P}, \alpha x) = \mu^\varepsilon(\hat{\mu}(\mu^*(\tilde{P}, \alpha), x))$, $\tilde{P}$ in $\mathcal{F}(Q)$, $x \in \Sigma$, $\alpha \in \Sigma^*$.

In the automaton $MF(\omega)$ the values $\mu_{i_a^{qp}}$, $\mu_{c_{xa}^{qp}}$, $\mu_{d_x^q} \in [0,1]$, $\forall x, a \in \Sigma$, $\forall q, p \in Q$, and the selected t-conorm and a t-norm, determine the final similarity value that $MF(\omega)$ can reach for an observed string α. As $M(\omega)$ has a unique final state q_n, for an observed string α the fuzzy automaton $MF(\omega)$ calculates the membership value $\mu_{\mu^*(\tilde{\sigma},\alpha)}(q_n)$ of α

in the fuzzy language $\tilde{L}(MF(\omega))$. Therefore, the fuzzy automaton may be interpreted as a *similarity operator* such that for each observed input string α in Σ^* it assigns the value $MF(\omega, \alpha) = \mu_{\mu^*(\bar{\sigma},\alpha)}(q_n) \in [0,1], \omega, \alpha \in \Sigma^*$. In [6], we provide an algorithm for computing $MF(\omega, \alpha)$. The complexity of the algorithm is $\mathcal{O}(m \times n)$ where m is the length of observed string α, and n is the length of the pattern string ω.

3 Tuning the fuzzy automaton

In this section, we describe the learning process applied to select the parameters of the Fuzzy automaton. The scenario considers patterns, $\omega \in \Sigma^*$, and observed strings, $\alpha \in \Sigma^*$, such that there exist a mapping $f : \Sigma^* \rightarrow \Sigma^*$ which associates to each observed string α a unique pattern ω, $f(\alpha) = \omega$. We represent the similarity between an observed string, α, and the whole set of patterns with a vector $out(\alpha) = [MF(\omega_1, \alpha), \ldots, MF(\omega_n, \alpha)]$. Each component represents the similarity of α with one of the patterns. As α is associated to a unique pattern, the desired output is a vector $target(\alpha) = [a_1, \ldots, a_i, \ldots, a_n]$, such that $target(\alpha)[i] = 1$ if and only if $f(\alpha) = \omega_i$; in other case, $target(\alpha)[i] = 0$.

In order to evaluate the error of the fuzzy automaton when classifying a given observed string α, we are going to consider as metric the euclidean distance: $||target(\alpha) - out(\alpha)||$. The metric allows evaluating the goodness of the fuzzy automaton depending on the selected parameters. Once we have the function to be optimized and the set of parameters to be tuned, we use a genetic algorithm [11] to explore in the space of the parameters. The metric allows selecting those parameters that maximizes the similarity between the observed string and its associated pattern and also maximizes the difference between the observed string and the other patterns.

The genetic algorithm can be formalized as a function, $GA(O, P, MF, \{\mu\}, \{\gamma_\omega\})$, where O is the set of observed strings; P the set of patterns; MF the fuzzy automaton; $\{\mu\}$ the membership values of the elementary edit operations and $\{\gamma_\omega\}$ the parameters of the parametric t-conorm/t-norm. The goal of the genetic algorithm is to determinate the parameter values that minimize the mean quadratic error of the system: $J(O) = (1/2) \sum_{\alpha \in O} ||target(\alpha) - out(\alpha)||$.

We have used a genetic algorithm with the following characteristics. There are 10 populations, which evolve during 300 evolution steps. At each step, the best four populations remain in the system for the next step of evolution. The other six are generated by selecting pairs of populations. The crossing and mutation probabilities are 0.25 and 0.1 respectively.

Each population is composed by a binary chromosome that contains two kinds of genes. One represents the values of the edit operations while the other represents the parameter of the t-norm and t-conorm used to evaluate the transition between consecutive states in the automata. The edit operation values are 27x27 values in order to consider the substitute, delete and insert operations among all characters and the empty string. Each value uses 7 bits to represent a value between 0.1 and 1. In order to evaluate the similarity the automaton uses the same expression of the t-norm and t-conorm, but with a different parameter for each pattern. So, in order to adjust a parameter for each pattern we use a gen with 4 bits representing a value between 0.1 and 10.

The experiment considers a set of 500 patterns with an average length of 7 characters. There are 1000 observed strings that derive from the patterns. The observed strings are generated by including one edit error in each subsequence of five characters. The probability of each edit error is uniformly distributed. The scheme of error generation provides a set of strings with an average of 1.5 errors per observed string.

Finally, we consider two possibilities for the automaton depending on the t-norm and t-conorm used by it. In the first case, the fuzzy automaton uses the max-min operator. These operators are not parametric so, the genetic algorithm selects only the edit operations membership values. In the second case, the fuzzy automaton applies the Hamacher operator [10]. We want to show how the inclusion of parametric norms leads to better performance.

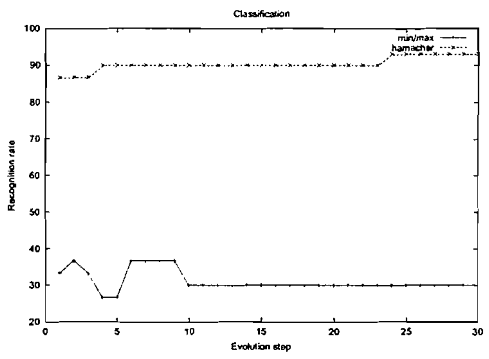

Figure 1: Recognition rates obtained in the experiment

The results are summarized in Figure 1. It shows that the use of a parameterized operator allows to improve the recognition rate and that the genetic algorithm provides an useful method to select automatically the parameters of the system. In this case, the system classifies correctly the 93% of the observed strings. Moreover, it is showed that genetic algorithms exploit the flexibility provided by the parametric t-norm/t-conorm in order to increase the classification rate.

The results obtained show that it is possible to use an automatic method to select the parameters of the similarity measurement system and that the flexibility introduced by the parameters of the transition operators are useful to better tune the fuzzy automaton in order to adequate its performance for a particular problem.

4 Conclusions

In this paper, we propose a method to select the parameters of a similarity measurement system based on a fuzzy automaton which models the typical edit operations used to transform an observed string into a pattern string. The similarity value is the membership value of the observed string into the fuzzy language associated to the pattern string via the fuzzy automaton.

The proposed fuzzy automaton is able to compute several similarity measurements by modifying the parameters that characterize it. We have shown that the selection of parametric t-norms/t-conorms leads to improve the performance of the system as well as its adaptability.

The system not only offers good performance without knowing a priori the kind of errors that the input string could contain. Its parameters can also can be trained in order to better tune the fuzzy automaton performance for a particular problem.

References

[1] D. Sankoff, J.B. Kruskal; *Time Warps, String Edits and Macromolecules: The Theory and Practice of Sequence Comparison*; Addison-Wesley, Reading, Mass. 1983.

[2] V.I. Levenshtein; 'Binary codes capable of correcting deletions, insertions, and reversals'; *Soviet Physics Docklady*, vol. 10, n. 8; pp. 707-710; 1966.

[3] B. Oommen; 'Constrained String Editing'. *Information Sciences*, vol. 40, pp. 267-284, 1986.

[4] A. Marzal, E. Vidal; 'Computation of Normalized Edit Distance and Applications'. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 15, n. 9, pp. 926-932, Sep. 1993.

[5] H. Bunke, J. Csirik; 'Parametric String Edit Distance and its Application to Pattern Recognition'; *IEEE Transactions on Systems, Man and Cybernetics*, vol. 25, n. 1, pp. 202-206, Jan. 1995.

[6] González de Mendívil;*Fuzzy automata for imperfect string matching*; Proccedings of ESTYLF 2000, pp. 527-532, Sevilla; 2000.

[7] Novak, V.: Fuzzy Sets and Their Applications. Ed. Adam Hilger, Bristol 1989.

[8] Schek, H. J. Tolerating Fuzziness in Keywords by Similarity Searchers. Kybernetes 6, 1077, 175-184.

[9] H.J. Zimmermann; *Fuzzy Set Theory and its Applications*; Kluwer Academic Pub., Boston, MA; 1990.

[10] G.J. Klir, B. Yuan; *Fuzzy sets and fuzzy logic: Theory and applications*; Prentice Hall, N.J., 1995.

[11] D. E: Goldberg; *Genetic algorithms in search, optimization and machine learning*, Addison-Wesley, Reading MA; 1989.

[12] J. Hopcroft, J. Ullman; *Introduction to Automata Theory, Languages and Computation*; Addison-Wesley Publishing Company, Reading Massachusetts; 1979.

[13] M. Mizumoto, J. Toyoda, K. Tanaka; 'Fuzzy Languages'; *Systems Computers Control*, vol. 1, n. 3; pp. 36-43; 1970.