Transformation Distances, Strings and Identification of DNA Promoters.

Maciej Marczak[1], Włodzisław Duch[2], Karol Grudziński[2], and Antoine Naud[2]

[1] Systems Research Institute, Polish Academy of Sciences, Warsaw, Poland
[2] Department of Informatics, Nicholas Copernicus University, Grudziądzka 5, 87-100 Toruń, Poland. www.phys.uni.torun.pl/kmk

Abstract. Computational intelligence methods usually work in vector spaces and are not able to deal with objects that have complex structures. Methods based on similarity may be applied in structural domains. Similarity may be defined by minimal cost needed to transform one object into another. The costs of the substitution operations that such transformation is composed from may be treated as adaptive parameters. For strings this leads to a generalization of edit (Levenshtein) distance. This distance is computed using dynamic programming method and applied to the problem of identifying DNA gene promoter sequences.

1 Introduction

Neural networks and other computational intelligence (CI) methods assume very simple vector space paradigm for representation of knowledge. Artificial intelligence (AI) deals with symbolic representation of knowledge that usually cannot be represented in such way. In effect AI and CI became widely separated, with AI community focusing on modeling of complex knowledge and inductive symbolic machine learning approaches, and CI community focusing on pattern recognition problems. Both approaches are insufficient in many situations and a new learning paradigm is needed [1]. Complex systems, such as commercial organizations, chemical molecules or molecular biology objects (DNA sequences, genes, proteins) cannot be easily accommodated in AI or CI knowledge representation frameworks.

A general framework for processing of structural data, based on recurrent neural networks and hidden Markov models, has been introduced [2], but it is rather difficult to implement and use. The two most common knowledge representation schemes in AI are based on the state or the problem description [3]. The goal is also a state or a simple problem that has known solution. A set of operators is defined, transforming the initial object (state, problem), into the final object (goal). Each operator has some costs associated with its use. Solutions are represented by paths in the search graph. The best solution has lowest costs of transforming the initial object into the final object.

This seems to be a very good model for CI methods applied to complex objects. It is used in this paper for DNA sequences, but the method is quite general and may be applied to arbitrary complex objects, as long as transformation operators and associated costs may be defined. Evaluation of similarities between objects should be done efficiently, therefore we have developed a method based on dynamic programming [4]. Members of the same class (cluster) should have high internal similarity,

while those from different classes (clusters) should differ significantly. This may be achieved by adapting costs of operations that are performed on objects. Experimental results for the identification of promoter sites in DNA strings are given in the third section. A short discussion closes this paper.

2 Transformation distances

Comparison of two objects is done using similarity or equivalently distance measures. A set of operators should be given, with associated costs of using each operator. In case of strings such operators are substitutions of single letters or substrings by other substrings, including an empty substring (representing deletion). In case of numerical values shift operators may change one value into another at the cost proportional to the numerical difference between the values. The set of operators should be complete, i.e. for every pair of objects a transformation that changes one of the objects into another should exist.

A transformation τ is defined by a finite sequence of operators $\tau = \prod_i \hat{\Omega}_i$. Transformation $\tau\, O_i$ always creates a single object O_j , but for every pair of objects O_i and O_j many transformations connecting these objects (i.e. changing one object into the other) may exist. The cost of the transformation $W(\tau)$ is a sum of the costs of all the operations $W(\tau) = \sum_i W(\hat{\Omega}_i)$. These costs will be called below the operation (or substitution) weights.

The edit distance of two strings S_1 and S_2 , is defined as the minimum number of single-letter insertions, deletions and substitutions required to transform S_1 into S_2. Each of these operations has its own weight. Efficient sequential and parallel algorithms for computing the edit distance between two strings exist, based on the idea of dynamic programming. A distance $D(O_i, O_j)$ between a pair of objects O_i, O_j is equal to the minimum weight of all transformations connecting these objects. At first sight calculation of such distances may look like an NP-complete problem. One of us has proved [4] that it is equivalent to the generalization of the edit distance (Levenshtein distance). An efficient algorithm based on dynamic programming was found to calculate these distances, with the complexity of the order of $O(|O_i| \times |O_j|)$, where $|O_i|$ is the length of the object O_i, i.e. it distance from an empty object.

A String Transformation System (STS) is a triple (O, S, W), where:
O – a set of objects, which are finite sequences of features
S – a set of elementary substitutions, which are finite pairs of sequences of features
W – a normalized weight function, assigning to all substitutions weights.

Normalization means that the sum of weights for all elementary substitutions is equal to one. This condition may be replaced by fixing the value of one, frequently used substitution. String Transformation Systems may be treated as a special case of the general Transformation Systems described in [1,5]. Sets of substitutions, each with its own weight, create multi-letter substitutions, e.g. $(abdc, cg)$, $(a, atccg)$, $(daac, \emptyset)$, where $\emptyset$ is a null string. In this case the edit distances are computed using the dynamic programming [6].

3 Classification of complex objects

The ability to calculate distances or similarities between complex objects allows to use the powerful similarity-based approaches for calculation of classification probabilities [7]. Consider a two-class problem with a set of $C+$ training cases from the first class and $C-$ training cases from the second class. For all objects belonging to these sets the distance matrix $D_W(O_i, O_j)$ is computed. This matrix depends on the set of weights $\{W_i\}$ assigned to the elementary substitutions Ω_i. Classification probabilities $p(C_i|\mathbf{X};W)$ depend on the distances, therefore they also depend on the choice of substitutes and their weights. In supervised learning these weights are optimized using the error function:

$$E(W) = \sum_{\mathbf{X}} \sum_{i} (p(C_i|\mathbf{X};W) - P_i(\mathbf{X}))^2 \tag{1}$$

where $P_i(\mathbf{X})$ is the true probability of assigning the vector $\mathbf{X}$ to the class C_i, known for the training set vectors. This error function should be minimized in respect to parameters W and all other parameters and procedures used to calculate the $p(C_i|\mathbf{X};W)$ probability [7]. In an ideal case distances between all objects in the $C+$ class should be zero, and the same for the distances inside $C-$ class. A set of weights and substitutions that achieve this explains the class structure, but for real life data this situation will rarely be possible.

The disadvantage of error function minimization is the need to recalculate all lowest-cost distances. Below a computationally less expensive approach is followed. Objects will be represented in vector spaces reflecting their structural relations. Some minimal transformations connect only one pair of objects, while some may connect many. Among all transformations with minimal cost for every pair of objects O_i and O_j those that connect the largest subset are called Primary Transformations (PTs). They represent the most probable hypothesis or the most useful transformations.

For every object O two vectors defining its relations with all objects from its own class and from all other classes are created:

$$V^+(O) = \left[v_1^+, ..., v_n^+\right], V^-(O) = \left[v_1^-, ..., v_n^-\right], \tag{2}$$

where v_i^+ (similarly v_i^-) is the number of S_i substitutes ($i = 1 \ldots n$) in the primary transformations that connect this object with all others in the $C+$ set. These vectors provide structural representation of object relations. This representation depends on the choice of the elementary substitution set S and the choice of weights W. For the string objects a simple set of letter substitutions (identity substitutions $s \leftrightarrow s$ plus symbol removal/addition $s \leftrightarrow \theta$ substitutions) and constant weights lead to quite good results. Thus for n symbols objects are represented by two vectors in the $3n$-dimensional space. In this space the difference of the two vectors $X(O) = V^+(O) - V^-(O)$ may be used with the linear discriminiation to define a separating hyperplane W_R:

$$O_i \in C^+ \Leftrightarrow X(O_i) \cdot W_R > 0; \quad O_i \in C^- \Leftrightarrow X(O_i) \cdot W_R < 0 \tag{3}$$

for all O_i objects in the training set. Once the $X(O)$ vectors are defined any method of classification may be used, in particular linear discrimination or a Support Vector Machine approach. Of course there is no guarantee that this representation will lead to a separable problem.

4 Experiments with promoters

A site on a DNA string to which RNA polymerase will bind and initiate transcription is called a promoter. These sites contain signals that are used to regulate gene expression. *Escherichia coli* is a well-known bacterium with 4.6 million bases in its genome. A small number (106) of short DNA subsequences (57 nucleotides) have been identified, exactly half of them containing the promoter signals. The task is to distinguish between these sequences. The data contains strings of the type ... tttatatttttcgcttgtcaggccg ... The data is available from the UCI repository [8].

Since 4 letters, a, c, t, g are present in the string a naive way to change them into numerical values is to replace them by 1, 2, 3 ,4. In the STS approach structural representation of strings described above is defined in 12-dimensional space (4 symbols, 3 transformations: identity and $s \leftrightarrow \theta$. For example, the value of the first feature for object O is equal to the number of $a \leftrightarrow a$ substitutions in the primary transformation connecting this object to all others from the $C+$ class. Classification procedures applied directly to promoter data work in 57-dimensional space.

STS results were compared with the results of the nearest neighbor methods with various distance functions and optimization of k, the number of neighbors taken into account during classification. Using the leave-one-out method with k=1 and Euclidean distance (the most common k-NN variant) gives only 70.8% correct answers. Optimization of k indicates best solution for $k = 18, 19$, increasing accuracy to 84.0%. Other distance function improve results significantly, with the best results obtained using probabilistic, data dependent Value Difference Metric [7] that was computed separately for the 105 training vectors in each partition of the leave-one-out procedure, and applied to the one test vector. Optimizing k allows to reach 91.5% of accuracy with this method.

Table 1. Classification results for the promoters data with different distance functions.

Method	Leave-one-out accuracy %	No. of errors
ID3 [9]	82.1	19
Euclides, k=19	84,0	17
Manhatan, k=19	85,9	15
Canberra, k=33	87,7	13
STS, k=1	90,6	10
Value Difference Metric, k=7	91,5	9
MLP [9]	92.5	8
STS, k=9	94,3	6

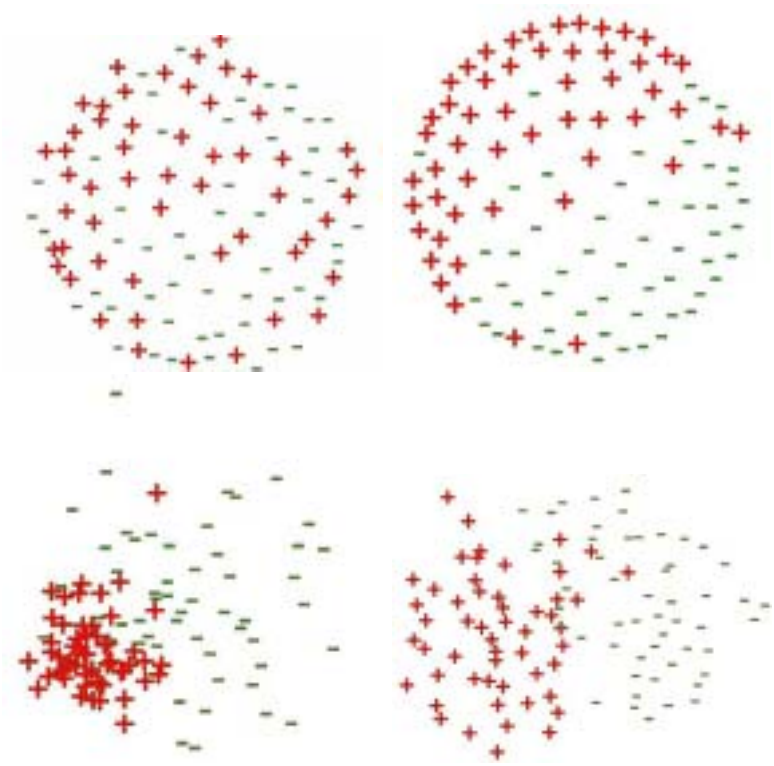

Fig. 1. MDS representation of the original (Euclidean) DNA string distances (top-left) and 3 representations of the same data shown using transformation distances with different costs of substitutions. Promoter instances are marked as $+$, remaining as $-$.

Turning to the STS approach, the simplest set of substitutions with equal costs leads to the distance matrix that has been used in the kNN procedure, giving in the leave-one-out test 90,6% accuracy for k=1. Several methods for adjusting costs of substitutions were tried, resulting with results reaching 94.3% accuracy in the kNN leave-one-out procedure (corresponding to about 6 errors). Similar results are achieved with linear discrimination using $V^+ - V^-$ matrix. Table 1 contains comparison of results obtained on this dataset. Towell *et. al* [9] report that adding domain knowledge and refining it with the neural network they obtained 4 errors only. Knowledge-based support vector machine has been used on this data recently, making 5 errors [10]. We have not used any additional information to improve the results.

Multidimensional scaling technique allows to visualize relations between data objects [11]. In our case original space has 57 dimensions. Ideal transformation distance should lead to two well separated small clusters. Clusterization is positively correlated with the accuracy of classification. Several examples of the effects of transformation distances with different substitution costs are shown in Fig.1. In all

cases configuration obtained from the principal component analysis was taken for the start of the MDS procedure. Transformation distances improve the clusterization of sequences taken from promoter and the non-promoter DNA quite clearly.

5 Conclusions

Although the research reported here is still in the preliminary stage it offers a relatively easy extension of computational intelligence methods to complex domains where problems cannot be easily represented in the vector space as a set of a fixed number of features. Our preliminary results are very encouraging and it is obvious that many methods may be used in connection with the transformation distances. Full optimization of costs of all operations may become difficult, requiring global optimization, therefore methods should be thought in which efficient gradient techniques are applicable. Numerous extensions and applications of the approach described here are planned in the near future.

Acknowledgments: Support by the Polish Committee for Scientific Research, grant 8 T11C 006 19, is gratefully acknowledged.

References

1. Goldfarb, L, Golubitsky O. (2001) What is a structural measurement process? Faculty of Computer Science, U.N.B., Technical Report TR01-147
2. Frasconi P, Gori M, Sperduti A. (1998) A General Framework for Adaptive Processing of Data Structures. IEEE Transactions on Neural Networks **9**, 768-786
3. Rich E, Knight K. (1991) Artificial Intelligence. 2nd ed, McGraw-HillInc, New York.
4. Marczak, M. (2001) Zastosowanie uogólnienia odległości Levenshteina w systemie transformacji symbolicznych. Analiza Systemowa w Finansach i Zarządzaniu, Wybrane problemy, Vol.3, IBS PAN Warszawa
5. Goldfarb, L. Nigam, S. (1994) The unified learning paradigm: A foundation for AI. In: V.Honovar, L.Uhr, Eds. Artificial Intelligence and Neural Networks: Steps Toward Principled Integration. Academic Press, Boston
6. Gusfield D. (1997) Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology. Cambridge Univ Press, Cambridge, UK
7. Duch, W. (2000) Similarity base methods: a general framework for classification, approximation and association. Control and Cybernetics **29**, 937-968
8. Blake, C.L, Merz, C.J. (1998). UCI Repository of machine learning databases http://www.ics.uci.edu/ mlearn/MLRepository.html. Irvine, CA: University of California, Department of Information and Computer Science.
9. Towell, G, Shavlik, J, Noordewier, M. (1990) Refinement of Approximate Domain Theories by Knowledge-Based Artificial Neural Networks. In: Proc. 8th National Conf. on Artificial Intelligence (AAAI-90), Boston, MA, p. 861-866
10. Fung, G, Mangsarian, O.L, Shavlik, J, (2001) Knowledge-BasedSupport VectorMachineClassifiers. Data Mining Institute Technical Report 01-09, Univ. of Wisconsin, 2001
11. Naud A. (2001) Neural and statistical methods for the visualization of multidimensional data. PhD Thesis, Dept. of Informatics, N. Copernicus University http://www.phys.uni.torun.pl/kmk/publications.html