Learning in Networks of Similarity Processing Neurons

Lluís A. Belanche

Computer Science School - Dept. of Software
Technical University of Catalonia
Jordi Girona, 1-3 08034, Barcelona, Catalonia, SPAIN
belanche@lsi.upc.edu

Abstract. Similarity functions are a very flexible container under which to express knowledge about a problem as well as to capture the meaningful relations in input space. In this paper we describe ongoing research using similarity functions to find more convenient representations for a problem –a crucial factor for successful learning– such that subsequent processing can be delivered to linear or non-linear modeling methods. The idea is tested in a set of challenging problems, characterized by a mixture of data types and different amounts of missing values. We report a series of experiments testing the idea against two more traditional approaches, one ignoring the knowledge about the dataset and another using this knowledge to pre-process it. The preliminary results demonstrate competitive or better generalization performance than that found in the literature. In addition, there is a considerable enhancement in the interpretability of the obtained models.

Key words: Similarity representations; Classification; Neural Networks

1 Introduction

The intuitive notion of *similarity* is very useful to group objects under specific criteria and has been used with great success in several fields like Case Based Reasoning [1] or Information Retrieval [2]. Interest around purely similarity-based techniques has never faded way; on the contrary, it has grown considerably since the appearance of kernel-based methods [3]. In learning systems, a non-written principle states that similar inputs should have similar outputs for the model to be successful. While this is no guarantee of good performance –specially near class boundaries, where the principle is violated– it certainly is a *sine qua non* condition. If the principle is not true, generalization becomes almost impossible. For a learning system, the trick is then to capture (that is, to *learn*) meaningful similarity relations in relation to the prescribed target variable.

Specific similarity functions from the point of view of data analysis have been used with success since the early days of pattern recognition. Modern modelling problems are difficult for a number of reasons, including dealing with mixtures of data types and a significant amount of missing information [4]. For example,

in the well-known UCI repository [5] over half of the problems contain explicitly declared nominal variables, let alone other data types (*e.g.*, ordinal), usually unreported. In many cases this *heterogeneous* information has to be encoded in the form of real-valued quantities, although there is often enough domain knowledge to characterize the nature of the variables.

The aim of this paper is to demonstrate the learning abilities of simple layered architectures, where the first layer computes a user-defined *similarity function* between inputs and weights. The basic idea is that a combination of partial similarity functions, comparing variables independently, is more capable at capturing the specific properties of an heterogeneous dataset than other methods, which require *a priori* data transformations. An appealing advantage is found in the enhanced interpretability of the models, so often neglected in the neural network community. In order to develop the idea, we propose to compute the similarities among the elements in the learning dataset, and then use a *reduction* method to select a small subset thereof. These selected observations are the *centers* of the first hidden layer. In other words, the first hidden layer is a change of the representation space from the original feature space to a similarity space [6].

2 Methodology

2.1 Preliminaries

We depart from a training data matrix $D_{N\times d}$ composed of N observations $\mathbf{x}$ described by d variables, plus a target matrix $D_{N\times t}$ containing the known targets of the N observations. For simplicity, in this paper we concentrate in classification problems only (two-class or multiclass) and set $t = 1$.

Given a similarity function s, we first compute the associated symmetric similarity matrix $S_{N\times N}$, where $S_{ij} = s(\mathbf{x}_i, \mathbf{x}_j)$. An algorithm is then needed to select a number d' of prototypes, that best represent the learning data in the following sense:

1. the prototypes must be known elements of the input space (observations);
2. all observations that are not prototypes must show a high similarity to (only) one of the prototypes in relation to their similarity to the other prototypes;
3. d' should be set much smaller than N.

This is an ideal task for a clustering algorithm, although not all clustering methods are adequate, and certainly alternative techniques could be possible. As an example, artificial immune systems have been used for prototype selection tasks using nearest-neighbor classifiers [7].

The result of this process is a layer of d' units, which we call S-neurons. Any learning method can now operate in the representation space spanned by the layer of d' S-neurons. However, it pays to start using linear methods, both for computational (they are fast) and analytical (they have a single optimum) reasons. It also turns out that the resulting model can be much more interpretable.

2.2 Detailed description

Let us represent the observations as belonging to a space $X \neq \emptyset$ as a vector $\mathbf{x}$ of d components, where each component x_k represents the value of a particular feature k. A *similarity measure* is a unique number expressing how "like" two observations are, given these features. It can be defined as an upper bounded, exhaustive and total function $s : X \times X \to I_s \subset \mathbb{R}$ such that I_s has at least two different elements (therefore I_s is upper bounded and $s_{max} \equiv \sup_{\mathbb{R}} I_s$ exists).

A basic but very useful S-neuron can be devised using a Gower-like similarity index, well-known in the literature on multivariate data analysis [8]. For any two vector objects $\mathbf{x}_i, \mathbf{x}_j$ to be compared on the basis of feature k, a score s_{ijk} is defined, described below. First set $\delta_{ijk} = 0$ when the comparison of $\mathbf{x}_i, \mathbf{x}_j$ cannot be performed on the basis of feature k for some reason; for example, by the presence of missing values, by the feature semantics, etc; $\delta_{ijk} = 1$ when such comparison is meaningful. If $\delta_{ijk} = 0$ for all the features, then $s(\mathbf{x}_i, \mathbf{x}_j)$ is undefined. The partial scores s_{ijk} are defined as follows:

Binary (dichotomous) variables indicate the presence/absence of a trait, marked by the symbols + and −. Their similarities are computed according to Table 1, leading to a partial coefficient introduced by Jaccard and well known in numerical taxonomy as the Jaccard Coefficient [9].

Table 1: Similarities for dichotomous (binary) variables.

	Values of feature k			
Object $\mathbf{x}_i$	+	+	−	−
Object $\mathbf{x}_j$	+	−	+	−
s_{ijk}	1	0	0	0
δ_{ijk}	1	1	1	0

Categorical variables can take a number of discrete values, which are commonly known as *modalities*. For these variables no order relation can be assumed. Their *overlap* similarity is $s_{ijk} = 1$ if $x_{ik} = x_{jk}$ and $s_{ijk} = 0$ if $x_{ik} \neq x_{jk}$.

Real-valued variables are compared with the standard metric in $\mathbb{R}$: $s_{ijk} = 1 - |x_{ik} - x_{jk}|/R_k$, where R_k is the *range* of feature k (the difference between the maximum and minimum values). The overall coefficient of similarity is defined as the average score over all partial comparisons:

$$S_{ij} = s(\mathbf{x}_i, \mathbf{x}_j) = \frac{\sum_{k=1}^{n} s_{ijk}\delta_{ijk}}{\sum_{k=1}^{n} \delta_{ijk}}$$

Ignorance of the absent elements and normalization by the number of the present ones has been found superior to other treatments in standard data anal-

ysis experiments [10][1]. This coefficient has been extended to deal with other data types, like ordinal and circular variables [11]. Notice that we now have $s_{max} = 1$.

As for the clustering, we choose the PAM algorithm, which partitions data into k clusters, very much like k-means. However, PAM offers two advantages: first, the cluster centers (called *medoids*) are chosen *among* the data points; second, the algorithm first looks for a good initial set of medoids and then finds a suboptimal solution such that there is no single switch of an observation with a medoid that will decrease the reconstruction error (the sum of distances of the observations to their closest medoid). The algorithm is fully described in [12].

3 Experiments

In this section we report on experimental work in which the previous ideas are applied both to linear learners –logistic and multinomial regression (LogReg and Multinom)– and linear discriminant analysis (LDA) and also to a non-linear learner (a standard SVM using the RBF kernel). All methods are deterministic and use the same data partitions. The smoothing parameter in the RBF kernel is estimated using the *sigest* method, based upon the 10% and 90% quantiles of the sample distribution of $\|\mathbf{x}_i - \mathbf{x}_j\|^2$ [13]; the cost parameter C is set to 1.

All datasets are split into learning and test parts (respecting original partitions, if available). For missing value imputation, we use the Multivariate Imputation by Chained Equations (MICE) method [14], which generates multiple imputations for incomplete multivariate data by Gibbs sampling. This method is attractive because, if the data contains categorical variables, these are also used in the regressions on the other variables.

We study three approaches:

raw There is no effort in identifying variable types (all information is considered numerical, and scaled); missing values are either not identified or left as they come (for example, treated as zeros).

std All variable types are properly identified; non-numerical information is binarized with a standard dummy code [15]. Missing values are identified and imputed with MICE.

sim Same as before with a first layer of S-neurons, as described in Section (2.2); then PAM selects $d' = \lfloor 0.05 \cdot N \rfloor$ prototypes in the learning part. Notice that, in this case, the model has the architecture of a neural network.

3.1 Datasets

Some challenging problems have been selected as characteristic of modern modeling datasets because of the diversity in data heterogeneity and the presence of

[1] It is not difficult to realize that this is equivalent to the replacement of the missing similarities by the average of the non-missing ones. Therefore, the conjecture is that the missing values, if known, would not change the overall similarity significantly.

missing values. The problem descriptions and the datasets are taken from the UCI repository [5]. The available documentation has been analyzed for an assessment on the more appropriate treatment. Missing information is also properly identified – see Table 2. The Horse Colic dataset has been investigated with two different targets (variables #23 and #24, resp.).

Table 2: Basic characteristics of the datasets: #Obs (learning, test). Def. (default accuracy), Missing (percentage of missing values). In→Out (no. of inputs and outputs). The last column shows variable types: (R)eal, (N)ominal, or(D)inal.

Name	#Obs	Def.	Missing	In→Out	Data
Pima Diabetes	768 (500,268)	65.1%	10.6%	8 → 2	8R, 0N, 0D
Horse Colic-23	363 (295,68)	61.4%	25.6%	22 → 3	7R, 7N,8D
Horse Colic-24	364 (296,68)	63.5%	25.6%	22 → 2	7R, 7N,8D
Audiology	226 (200,26)	66.3%	2.1%	31 → 4	0R, 24N, 7D

Pima Diabetes. This is a much studied dataset, in which a population of Pima Indian women living near Phoenix, Arizona, was tested for diabetes according to World Health Organization criteria. In this dataset, most of the variables show impossible zero values (e.g, the diastolic blood pressure), which are actually missing values [15]. Upon careful analysis, it turns out that only 392 out of the 768 observations are unaffected by missing values.

Horse Colic. This dataset makes an excellent case study, because of the diversity in data heterogeneity and a significant amount of missing values; it has been used as a paradigmatic example in some textbooks [16]. Each observation is the clinical record of a horse and the variables are specially well documented[2].

Audiology. This problem is interesting for many reasons: it is multiclass, has a low number of observations and all variables are categorical (with different numbers of modalities, and some of them ordered)[3]. We have reduced the original 24 classes to 4 by grouping and eliminated non-informative variables.

3.2 Results

The results are displayed in Tables 3, 4, and 5. At a first look, it is surprising how the learning methods are able to grasp the task using the **raw** method. In this sense, the **std** method is markedly better for LogReg and the SVM, but not for Multinom or LDA. However, the difference for Multinom is very small; for LDA, the **std** method increases input dimension quite a lot (specially if there are many categorical variables or these have many modalities). The explosion in

[2] This dataset is made available thanks to M. McLeish and M. Cecile (Computer Science Dept., Univ. of Guelph, Ontario, Canada).

[3] Original owner: Professor Jergen at Baylor College of Medicine.

the number of binary variables (due to the dummy coding) changes the data distribution to something extremely non-gaussian, and this causes trouble to LDA (this is specially acute in Audiology). The **sim** method presents similar results for LogReg and the SVM, and much better for both LDA and Multinom.

We would like to point out the good results delivered by linear models, like LogReg –when applicable– and LDA, specially for the **sim** method. On the other hand, few efforts have been devoted to a fine tuning of the SVM models beyond educated guesses, but this issue affects all approaches. Finally, no effort has been put in selecting the optimal number of centers for the **sim** approach.

Table 3: Generalization errors for the **raw** method.

	LogReg	Multinom	SVM	LDA
Pima	0.201	0.187	0.194	0.187
HorseColic-23	–	0.309	0.279	0.279
HorseColic-24	0.176	0.162	0.162	0.162
Audiology	–	0.231	0.154	0.269
AVERAGE	0.189	0.222	0.197	0.224

Table 4: Generalization errors for the **std** method.

	LogReg	Multinom	SVM	LDA
Pima	0.190	0.198	0.205	0.201
HorseColic-23	–	0.265	0.279	0.353
HorseColic-24	0.147	0.191	0.147	0.147
Audiology	–	0.269	0.038	0.731
AVERAGE	0.169	0.231	0.168	0.358

Table 5: Generalization errors for the **sim** method.

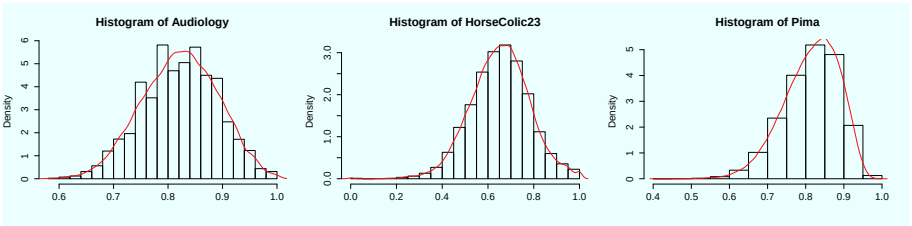
	LogReg	Multinom	SVM	LDA
Pima	0.183	0.190	0.194	0.175
HorseColic-23	–	0.324	0.294	0.309
HorseColic-24	0.162	0.176	0.176	0.191
Audiology	–	0.115	0.000	0.038
AVERAGE	0.172	0.201	0.166	0.178

3.3 Discussion

When comparing to related previous work, the obtained results are very competitive in relation to those typically reported for these problems, sometimes achieved using very sophisticated techniques. For example, for the Pima dataset, the best reported results are around 20%, topping at 19.8% [15], while typical results are in the range 21%-25% [17]. Our best result is 17.5% using LDA and the **sim** method. Among other causes, this is due to the bad identification or treatment of missing values, something that has been advocated elsewhere [15]. What is more, many (if not most) of these reported results are cross-validation ones, which means that there is no independent assessment of true generalization ability. In our case, the Pima results are reported in a test set that is large, compared to the learning set size (35%). For HorseColic-23, the best reported result seems to be 13.6% [17], while typical results are in the range 14%-23% [18]. Our best result is 14.7%, achieved using three different learners and the **std** method. There seems to be no comparable previous work for HorseColic-24. Finally, for Audiology it is difficult to compare because in this paper we have cleaned the dataset prior to learning; in any event, both the SVM and LDA achieve very good results with the **sim** method.

Another important issue is the distribution of the similarities across the observations and the classes. Fig. 1 shows this distribution for the different datasets. It can be seen that in all cases similarities are rather high and well-behaved (fairly symmetrical, unimodal). Given the assumed relation between similarity computations and learning ability, we computed the intra-class similarities. These were: Cochlear (0.847), Mixed (0.822), Normal (0.914) and Other (0.794) for Audiology, No (0.811) and Yes (0.788) for Pima, Died (0.646), Euthanized (0.634) and Lived (0.673) for HorseColic-23 and No (0.688) and Yes (0.673) for HorseColic-24. These numbers not only reflect how relatively compact the different classes are, they also indicate the hardness for a learning method based on distances or similarities (however they are computed). For example, HorseColic-23 and HorseColic-24 show markedly less compact classes. The relation to overall performance and to class-by-class performance is left for a further dedicated study.

Fig. 1: Similarity distributions for the different datasets.



4 Conclusions and Future Work

A shortcoming of many existent learning methods –and, in particular, neural networks– is the difficulty of adding prior knowledge to the model in a principled way. Current practice assumes that input vectors may be faithfully represented as a point in $\mathbb{R}^d$, and the geometry of this space is meant to capture the meaningful relations in input space. There is no particular reason why this should be the case, at least not with small numbers of hidden neurons. This paper has described ongoing research on more flexible learning frameworks, offering means for the injection of prior knowledge, and permitting a natural extension to operate in problems with non-numerical data types and showing missing values.

When talking about real problems, however, accuracy may not tell the whole picture about a model. Other performance criteria include development cost, interpretability and usability.

- The *cost* here refers to how much pre-processing effort we need in order to build the model. Undoubtedly, all but the **raw** method require more analysis time compared to doing (almost) nothing. Prior to learning the variable types must be identified and coded properly; for the S-neurons, suitable similarity measures must be chosen, using available background knowledge;
- The *interpretability* refers to the complexity of the obtained model in human terms. It is generally believed that accuracy and interpretability are in conflict [19]. In the present case, the methods based on similarity have a clear advantage in this case when combined with a linear learner: the prediction is a weigthed combination of the similarity of the input to a selected (and small) subset of prototypes. This framework resembles that of an SVM; however, in SVMs the kernel is providing an implicit transformation of the input space rather than a purely similarity-based representation. Moreover, the chosen similarity should be a valid kernel function;
- Finally, models must be *useful* in practice: in a real deployment of the model, new and unseen observations emerge which we need to classify, which display the same variable types and may contain missing values (that certainly could not be imputed at learning time). The similarity approaches are able to face this situation without further effort.

Current lines of research include the extension to new data types (a suitable similarity measure is needed in each case) and the design of formal measures to compute the relation between overall and, particularly, intra-class similarities with class distribution itself. A measure of similarity that is maximized for observations of the same class and minimized for observations of different classes is envisaged via the introduction of weights into the comparisons. This approach would permit the optimization of the similarity measure to some extent. Another important issue is the selection of the best centers, which could be performed in a supervised way. For example, performing a separate clustering per class and merging the results, or by using GLVQ methods [20]. This latter family of algorithms makes good use about the classes to which the input vector and the

winning codevector belong at prototype selection time. It is conjectured that a supervised reduction method will deliver better modelling results when coupled with subsequent stages of the method.

Acknowledgments. Partially supported by the MICINN project BASMATI (TIN2011-27479-C04-03) and by SGR2009-1428 (LARCA).

References

1. Osborne, H., Bridge, D. Models of similarity for case-based reasoning. Interdisciplinary Workshop on Similarity and Categorisation, pp. 173–179 (1997)
2. Baeza-Yates, R., Ribeiro, B. *Modern information Retrieval*. ACM Press, New York (1999)
3. Shawe-Taylor, J., Cristianini, N. *Kernel Methods for Pattern Analysis*. Cambridge Univ. Press (2004)
4. Little, R., Rubin, D. *Statistical Analysis with Missing Data*. Wiley, (2009)
5. Bache, K. and Lichman, M. UCI Machine Learning Repository. `http://archive.ics.uci.edu/ml`. Irvine, CA: University of California (2013)
6. Pekalska, E. *The Dissimilarity representations in pattern recognition. Concepts, theory and applications.* ASCI Dissertation Series no. 109. Delft University of Technology, Delft, (2005)
7. Garain, U. Prototype reduction using an artificial immune model. *Pattern Analysis and Applications* 11:3-4, 353-363 (2008)
8. Gower, J.C. A General Coefficient of Similarity and Some of Its Properties. *Biometrika*, 27(4), pp. 857–871 (1971)
9. Sokal, R. R. and Michener, C. D *Principles of Numerical Taxonomy*. San Francisco: W.H. Freeman (1963)
10. Dixon, J.K. Pattern recognition with partly missing data. *IEEE Trans. on Systems, Man and Cybernetics*, 9: 617-621, (1979)
11. Pavoine, S., Vallet, J., Dufour, A.B., Gachet, S., Daniel, H. On the challenge of treating various types of variables: application for improving the measurement of functional diversity. *Oikos*, 118(3), pp. 391–402, (2009)
12. Kaufman, L. and Rousseeuw, P.J. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley, New York, (1990)
13. Caputo, B., Sim, K., Furesjo, F., Smola, A. Appearance-based object recognition using SVMs: which kernel should I use? NIPS Workshop on Statistical methods for computational experiments in visual processing and computer vision, (2002)
14. van Buuren, S., Groothuis-Oudshoorn, K. mice: Multivariate imputation by chained equations in R. *Journal of Statistical Software*, 45(3):1–67, (2011)
15. Ripley, B. *Pattern Recognition and Neural Networks*. Cambridge Univ. Press, (1996)
16. Xu, R., Wunsch, D. *Clustering*. Wiley-IEEE Press, (2008)
17. Torre, F. Boosting Correct Least General Generalizations. Technical Report GRAppA-0104, (2004)
18. Yang, J., Parekh, R., Honavar, V. DistAI: An Inter-pattern Distance-based Constructive Learning Algorithm. *Intelligent Data Analysis* 3, pp. 55-73 (1999)
19. Breiman, L. Statistical Modeling: The Two Cultures. *Statistical Science* 16 (3), 199-231, (2001)
20. Pal, N.R., Bezdek, J.C., Tsao, E. Generalized clustering networks and Kohonen's self-organizing scheme. *IEEE Trans. Neural Networks*, 4(4) pp.549-557, (1993)