Tik-61.181 Special Course on Information Science I – Time in Self-Organizing Maps: Operator maps

Pasi Lehtimäki
plehtima@cis.hut.fi

October 2, 2002

1 Introduction

One of the most widely used unsupervised neural network algorithm is the Self-Organizing Map (SOM) [3]. The SOM consists of neurons or map units, each having a location in a continuous input space as well as in the discrete low-dimensional output lattice. The training of the SOM enables the formation of a topology preserving mapping between the input and output spaces, thus providing a nonlinear projection of multidimensional data into a low-dimensional grid that can be used for data visualisation purposes.

2 Self-Organizing Operator Maps

In the basic SOM algorithm the input patterns as well as the SOM map units are data points in the measurement space, thus representing only static information in the data. In order to import dynamical dependencies between the successive observations into SOM training, a variation of the SOM algorithm called *operator map* has been proposed. In operator maps, the data sequence $X(t) = \{x(t), x(t-1), \dots, x(t-n+1)\}$ is assumed to be generated by unknown *parametric models*. The central problem in the operator map training is the definition of the *similarity matching* between the observations, that is, the true but unknown data generating functions and the operators of the map as well as the *tuning* of the operator free parameters according to the observations. In other words, the training of the operator maps consists of estimation of the true generating function from the observed data sequence $X(t)$, matching the operators of the map to the estimated generating function in order to find the best-matching map unit (BMU) and finally the tuning of the operators towards the estimated generating function using the available observations $X(t)$.

2.1 Types of operators

For different applications, a different form for the data generating function as well as the operators of the SOM is assumed. The two forms considered in this report are dynamical input-output models and probabilistic models, both of which having quite different similarity matching and parameter tuning criteria. Next, the different dynamical input-output model parametrisations for the operators of the map are described. Then, the general framework for considering the operators as probabilistic models is discussed.

2.2 Dynamical input-output models

When the data is assumed to be generated a dynamical system, the operators are often called as *estimators*. The goal of each estimator i is to provide an estimate $\hat{x}_i(t)$ for the current input sample $x(t)$ given the past observation sequence $\{x(t-1), x(t-2), \ldots, x(t-n)\}$. The best estimator in the operator map is selected according to the winner search rule

$$c = \arg\min_i ||x(t) - \hat{x}_i(t)||,$$

where c is the index of the best estimator. Thus, the error measure used in the winner estimator search is the instantaneous squared prediction error produced by the estimators.

The parameters θ_i of the estimator i are usually updated using the gradient descent method

$$\theta_i(t+1) = \theta_i(t) - \frac{1}{2}\alpha(t)\frac{\partial E(t)}{\partial \theta_i(t)},$$

where $E(t)$ is the sample error functional

$$E(t) = \sum_{i \in L} h_{c,i}(t)||x(t) - \hat{x}_i(t)||^2,$$

where $h_{c,i}(t)$ is the value of the neighborhood function for the estimator i and the best-matching estimator c at time step t. Depending on the parametrisation of the estimator i, this update rule takes a variety of different forms.

2.2.1 Linear estimators

The simplest form of an operator map consists of linear estimators. The output of the linear estimator i is

$$\hat{x}_i(t) = \sum_{k=1}^{n} w_{ik}(t)x(t-k),$$

where the free parameters w_{ik} of the estimator i are also called *linear predictive coding (LPC) coefficients*. Each of the estimators provide an estimate of the next input sample as a linear combination of the past input samples, and the map unit with smallest prediction error is declared as the winner. Then, the parameters of the operators are updated using the LMS-algorithm

$$w_{ik}(t+1) = w_{ik}(t) + \alpha(t)h_{c,i}(t)w_{ik}[x(t) - \hat{x}_i(t)].$$

The main advantage of the linear estimators are the simple and effective parameter estimation.

2.2.2 Nonlinear estimators

The estimators of the operator map can also be non-linear. In that case, the output of the estimator i is

$$\hat{x}_i(t) = F[x(t-1), x(t-2), \ldots, x(t-k)],$$

where F is a nonlinear function. In [5], two parametrisations for the function F was described. The first choice for the function F was the *multilayer perceptron (MLP)*, since it is able to approximate any non-linear function with arbitrary accuracy given enough hidden layer neurons.

The second choice for the operator parametrisation was the *adaptive neuro-fuzzy inference system (ANFIS)*, in which the mapping performed by the estimator is completely described by a set of linguistic *fuzzy rules*, making it useful for process analysis. For more details of MLP and ANFIS operator maps, see [5].

2.2.3 Operator structure optimization

In previous sections, linear and non-linear estimators for the operator map were described. For both types of estimators, the estimation of the local models in the operator map are in principle concerned by the same problems occurring in linear and non-linear system identification. One of the most difficult problem in system identification is the estimation of the *model structure* in addition to the model parameters. Several strategies to estimate local operator structure and parameters have been proposed.

In [2, 6], an approach based on *evolutionary computing* is described. Each operator consist of a set of randomly initialized *individuals*. The training of the operator map consists of finding the best-matching operator according to the input data. Then, the operators on the neighborhood of the winning operator are updated by producing *crossovers* and *mutations* between the individuals of operator i and the best-matching operator c. Thus, the structure of the operators go through evolutionary changes during the map training.

In [5], the *regularization* framework for optimal operator structure is described. The basic idea in the regularization framework is to penalize the operators from unnecessary complexity. This is usually done by setting constraints for the model parameters by including an additional penalty functional for the error measure to be minimized by the operator map

$$E(t) = \sum_{i \in L} h_{c,i}(t) ||x(t) - \hat{x}_i(t)||^2 + \lambda \eta [\theta_i(t)],$$

where λ is a regularization parameter and η is a penalty function determining which values of the parameters are favoured in parameter estimation. The goal is to find a model structure and the value of λ that gives the best representation of the input data.

2.3 Probabilistic operators

A completely different approach to model data generating functions is to assume that they are probabilistic models. In such models, the mean squared prediction error used in the estimator maps is not appropriate. Instead, the *Kullback-Leibler* distance between the estimated data generating models and the operators of the SOM is used as the similarity measure

$$KL(p||q) = -\int p(x) \log \frac{q(x;\theta_i)}{p(x)} dx,$$

where $p(x)$ is the estimate of the true probability distribution of the data generating model and $q(x, \theta_i)$ is the corresponding distribution of the operator i with reference vector θ_i. The probabilistic models can be used in temporal data analysis by letting the operators $q(x, \theta_i)$ model the probabilities of the current input sample given the past input samples

$$P[x(t)|x(t-1), x(t-2), \ldots, x(t-k)],$$

where $x(t)$ is either a continuous input variable or a discrete random variable describing a random event.

3 Applications

3.1 Linear and nonlinear estimators

The operator maps with linear estimators have been applied in variety of problems. In [4], such a map was used in image segmentation based on textures. Each map unit was responsible of predicting the gray-level of a pixel as a linear combination of the gray-levels in a small 2D neighborhood, thus modeling a certain type of texture. The parts of the image containing different textures had responses on different parts of the operator map, thus providing a segmentation of the original image.

The linear estimators have also been used in speech recognition [3]. The time-domain speech signal was used to train the operator map. The different operators of the map learned distinct features of the speech signal, resulting in so-called phoneme map.

Recently, the linear estimator map has been used in process analysis [5]. 3G mobile network data was used to train the linear local estimators, each responsible of predicting the quality of the mobile cell operation based on current and past state variables. The different estimators provided a variety of linear dependencies between the mobile cell state variables and performance. The visualisation of the parameters of the local models provided qualitative information about the different linear dependencies between the system input and output variables that can be used in mobile cell parameter optimization.

In the same work, the nonlinear ANFIS estimators were used for the same purpose. In this case, the SOM operators were able to model also nonlinear dependencies between the system input and output. In addition, the local estimators did have a linguistic representation in the form of fuzzy rules. Thus, it was able to describe *how* the different operators produce their predictions for the system output given the system input.

3.2 Probabilistic models

In [1], probabilistic models for the SOM operators were used in order to *detect frauds* in a mobile communication network. The possible events of a mobile phone usage such as an on-going call, an on-going silence, the beginning of a call and the termination of a call were encoded in a set of random events with the corresponding probabilities. Then, the operator map was trained using the mobile phone user profiles. Finally, the different operators learned different types of user profiles. Some of the operators learned user profiles that are likely to be a result of a fraud.

4 Conclusion

In this report, the concept of self-organizing operator maps is shortly described. The operator maps consisted of a set of local parametric models, each representing certain input patterns. Depending on the problem at hand, different parametrisations for the operators can be used. The design of an operator maps requires a definition of the similarity matching between the observations and the map operators that can be used in the winner search and the operator tuning phases of the SOM algorithm.

It turns out that the concept of self-organizing parametric functionals is useful in many classes of data-driven engineering problems.

References

[1] J. Hollmen, V. Tresp, and O. Simula. A self-organizing map algorithm for clustering probabilistic models. In *Ninth International Conference on Artificial Neural Networks*, volume 2, pages 946–951, September 1999.

[2] H. Hyötyniemi, A. S. Nissinen, and H. N. Koivo. Evolution based self-organization of structures in linear time-series modeling. In *3rd Nordic Workshop on Genetic Algorithms*, pages 135–152, Helsinki, Finland, August 1997.

[3] T. Kohonen. *Self-Organizing Maps, 3rd edition.* Springer, 2001.

[4] J. Lampinen and E. Oja. Self-organizing maps for spatial and temporal ar models. In *Scand. Conf. on Image Analysis (SCIA)*, pages 120–127, Helsinki, Finland, 1989.

[5] P. Lehtimäki. *Self-Organizing Operator Maps in Complex System Analysis, Master's Thesis.* 2002.

[6] A. Nissinen. *Neural and evolutionary computing in modeling of complex systems.* PhD thesis, Control Engineering Laboratory, Helsinki University of Technology, 1999.