KI - Künstliche Intelligenz manuscript No.
(will be inserted by the editor)

Slow Feature Analysis: Perspectives for Technical Applications of a Versatile Learning Algorithm

Alberto N. Escalante-B., Laurenz Wiskott

the date of receipt and acceptance should be inserted later

Abstract Slow Feature Analysis (SFA) is an unsupervised learning algorithm based on the slowness principle and has originally been developed to learn invariances in a model of the primate visual system. Although developed for computational neuroscience, SFA has turned out to be a versatile algorithm also for technical applications since it can be used for feature extraction, dimensionality reduction, and invariance learning. With minor adaptations SFA can also be applied to supervised learning problems such as classification and regression. In this work, we review several illustrative examples of possible applications including the estimation of driving forces, nonlinear blind source separation, traffic sign recognition, and face processing.

Keywords Slow Feature Analysis · Hierarchical networks · Nonlinear feature extraction · Dimensionality reduction · High-dimensional data · Driving forces · Blind source separation · Object recognition · Face processing

1 Introduction

At a first glance the sensory information perceived by an animal (e.g., the excitation of individual receptors in the retina) may appear as a large number of signals that vary quickly and in a seemingly unordered fashion. When looking at a zebra, for instance, the receptor responses quickly switch between black and white as soon as the eyes move. However, collectively the input represents the zebra faithfully. Thus, the information of interest, namely the presence of the zebra, is somehow hidden in the sensory signal and can potentially be extracted by some complicated input-output function. Furthermore, the information of interest is much more stable than the receptor activities. The *slowness principle* generalizes this idea and assumes that many aspects of the environment essential for the survival of an animal (e.g., the position and identity of its prey and fellows) change at a time scale much slower than the sensory signals encoding these aspects. The assumption then is that input-output functions that extract slowly varying features, automatically extract aspects that are of particular relevance to the animal.

This principle has probably first been formulated by Hinton [13] and online learning rules were developed shortly after [9, 22]. *Slow Feature Analysis* (SFA) [31] is the first closed-form algorithm and one of its advantages is that it is guaranteed to find the optimal solution within the considered function space. The concise formulation of the SFA optimization problem also permits an extended mathematical treatment so that its properties are well understood analytically [10, 25, 30]. SFA was initially developed for learning invariances in a model of the primate visual system [11, 31] and was subsequently also used for learning complex-cell receptive fields [3] and place cells in the hippocampus [10]. However, SFA has turned out to be useful also for technical applications, which is the main focus of this article.

Alberto N. Escalante-B. · Laurenz Wiskott
Institut für Neuroinformatik, Ruhr-Universität Bochum
Universitätsstr. 150, D-44801 Bochum, Germany
E-mail: alberto.escalante@ini.rub.de
E-mail: laurenz.wiskott@ini.rub.de

2 Slow Feature Analysis (SFA)

2.1 SFA Optimization Problem

The optimization problem solved by SFA can be stated as follows [3, 31]. Given an I-dimensional input sig-

nal $\mathbf{x}(t) = (x_1(t), \ldots, x_I(t))^T$, find a vectorial function $\mathbf{g}(\mathbf{x}) = (g_1(\mathbf{x}), \ldots, g_J(\mathbf{x}))^T$ within a function space $\mathcal{F}$ such that for each component $y_j(t)$ of the output signal $\mathbf{y}(t) \stackrel{\text{def}}{=} \mathbf{g}(\mathbf{x}(t))$,

$$\Delta(y_j) \stackrel{\text{def}}{=} \langle \dot{y}_j(t)^2 \rangle \text{ is minimal (objective function)} \quad (1)$$

under the constraints

$$\langle y_j(t) \rangle = 0 \text{ (zero mean)}, \quad (2)$$

$$\langle y_j(t)^2 \rangle = 1 \text{ (unit variance)}, \quad (3)$$

$$\langle y_j(t) y_{j'}(t) \rangle = 0, \forall j' < j \text{ (decorrelation and order)}. \quad (4)$$

The delta value $\Delta(y_j)$ is defined as the temporal average $\langle \cdot \rangle$ of the squared derivative of y_j and is therefore a measure of slowness (or rather fastness). The constraints (2–4) assure that the output signals are normalized, not constant, and uncorrelated, so that different components code for different information of the input signal. Notice that the problem is solved iteratively for each y_j from y_1 to y_J, so that the first component is the slowest possible and the later ones are faster because they have the additional constraint of being uncorrelated to the earlier ones.

One obvious way of achieving slowly varying output would be low-pass filtering or temporal averaging. However, such an approach would have two disadvantages: (i) processing would be delayed by the averaging time, (ii) high-frequency information would be systematically eliminated. Notice that $\mathbf{g}(\mathbf{x})$ is an instantaneous function that does not permit temporal averaging. Consequently, SFA extracts features instantaneously but favors those that vary slowly and are stable over time. This also implies that the temporal structure of the input signal matters only during the training phase. Afterwards single input samples can be processed independently of each other.

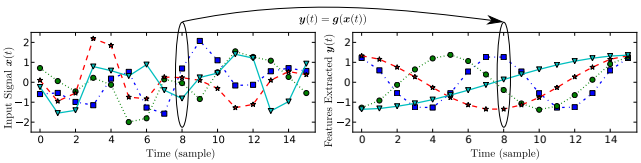

Fig. 1 Extraction of features from a 10-dimensional signal. Four components of the input signal (left) and the four slowest features extracted (right) are shown. This example was designed such that the output signals are the theoretically predicted optimal responses, namely cosines of increasing frequency.

2.2 Linear SFA

SFA is typically a nonlinear algorithm. However, in this section we present the linear version [31], in which $\mathcal{F}$ is the space of linear functions. Therefore, $\mathbf{g}(\mathbf{x}) \in \mathcal{F}$ can be written as $\mathbf{g}(\mathbf{x}) = \mathbf{W}^T\mathbf{x}$ with an $I \times J$ matrix $\mathbf{W}$. The mean of the training data $\mathbf{x}(t)$ has previously been removed for convenience without loss of generality. Usually discrete time is used, thus the input is a sequence of samples and the derivative signal is approximated as $\dot{\mathbf{x}}(t) \approx \mathbf{x}(t+1) - \mathbf{x}(t)$. First, a sphered signal $\mathbf{z} \stackrel{\text{def}}{=} \mathbf{S}^T\mathbf{x}$ is computed with a sphering matrix $\mathbf{S}$ that diagonalizes the covariance matrix $\mathbf{C} \stackrel{\text{def}}{=} \langle \mathbf{x}\mathbf{x}^T \rangle$, i.e. $\mathbf{S}^T\mathbf{C}\mathbf{S} = \mathbf{I}$. Then, the J directions of least variance in the derivative signal $\dot{\mathbf{z}}$ are found by principal component analysis on the derivative covariance matrix $\dot{\mathbf{C}}_z \stackrel{\text{def}}{=} \langle \dot{\mathbf{z}}\dot{\mathbf{z}}^T \rangle$ and represented by an orthogonal matrix $\mathbf{R}$. Finally, the algorithm returns $\mathbf{W} = \mathbf{S}\mathbf{R}$, $\mathbf{y} = \mathbf{W}^T\mathbf{x}$, and $\Delta(\mathbf{y})$. It has been shown that the solution fulfills the optimality criteria in the linear function space [31]. Interestingly, in this linear version SFA is closely related to independent component analysis based on time-delayed covariance matrices [4].

2.3 Nonlinear SFA

In order to solve real-world problems, nonlinear feature extraction is often desirable. This can be achieved by expanding the input data nonlinearly through a so-called *expansion function* followed by linear SFA. This results in slow features that are nonlinear with respect to the original data. The expansion can either be explicit [3, 31] or implicit using kernels [5, 28].

The choice of an appropriate expansion function is crucial [8]. If it is too simple (low-dimensional), it does not solve the problem; if it is too complex (high-dimensional), it might overfit on the training data and not generalize well to test data. In the applications below we use linear and quadratic SFA, as well as some other expansions.

2.4 High-Dimensional Data and Hierarchical SFA

Interesting types of data like images or 3D voxel data are usually high-dimensional. The complexity of the SFA algorithm is $\mathcal{O}(NI^2 + I^3)$ where N is the number of samples and I is the input dimension (possibly after nonlinear expansion), thus for high-dimensional data standard SFA is not feasible. In such cases a divide and conquer strategy to extract slow features is usually a good solution. For instance, one can divide the data into lower-dimensional blocks and extract local slow features separately with different instances of SFA, the so-called SFA nodes. Then, global slow features are extracted from the local slow features with

another SFA node in a next layer. This works because the first SFA nodes can be used to perform a dimensionality reduction, so that the input dimension to the SFA node in the second layer is much less than I. This strategy can be repeated iteratively for each block until the dimensionality is small enough, resulting in a multilayer hierarchical network. While this does not guarantee optimal global slow features anymore, in practice it has shown to be effective, probably because low-level features are spatially localized in most real data.

Interestingly, hierarchical processing is also a means to reduce or even avoid the overfitting problem mentioned above since each node in the hierarchical network gets to process relatively low-dimensional data, leading to good generalization. An additional advantage is that nonlinearity accumulates across layers, so that even when using simple expansions the network as a whole can realize a complex nonlinearity.

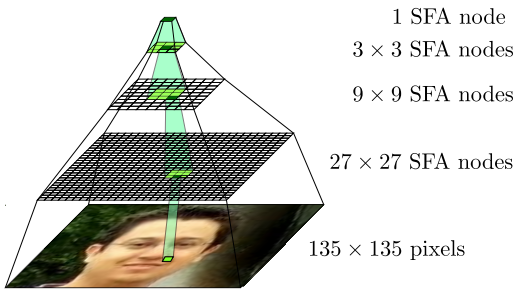

Fig. 2 An example of a hierarchical network with 4 layers. Each node in the hierarchy extracts 30 features and sends them to the next layer. The final layer consists of a single node and also outputs 30 features.

2.5 Extensions of SFA for Supervised Learning

SFA, as described so far, learns features that are implicit in the input signal. It extracts the slowly varying aspects and becomes insensitive or even invariant to quickly changing aspects. However, in many learning problems the input data does not take the form of a temporal sequence and the features to be learned are provided explicitly as labels along with the input samples, resulting in a supervised learning problem. How can SFA be applied in these cases? As an example, consider a number of face images for which the age of the person is known and the task is to automatically extract this feature from the images. A large part of the problem can be solved by SFA if one simply orders the images by increasing age and presents this to SFA. With this approach, age becomes the most slowly varying feature and SFA will naturally extract it – although not age directly but rather a feature that is monotonically related to it. A final step mapping the first SFA outputs to the real age label is therefore still required.

If the labels indicate classes, such as identity, one could order the images by class and proceed as above. However, it is more efficient to give up the linear temporal sequence and use graph structures instead. For classes one would connect all samples of the same identity with each other and make no connection between samples of different identity. In that case the graph separates into fully connected subgraphs, and we refer to it as a *clustered graph*. The SFA objective function would have to be modified to take into account all connections and try to minimize the output differences between connected samples. The ideal SFA output would therefore be constant for any given identity and different for different identities. Again a final supervised step would be required to map the arbitrary SFA output values to identity labels. Interestingly, in this special case linear SFA is equivalent to Fisher discriminant analysis [1, 2, 17].

This approach can be generalized to arbitrary similarity structures if the training samples, i.e. the nodes of the training graph, are weighted by v_t to indicate importance or frequency of a sample and the connections between samples, i.e. the edges of the graph, are weighted by $w_{tt'}$ indicating the similarity between the samples or rather their labels. In that case we have weighted versions of the objective and the constraints:

$$\Delta(y_j) := \frac{\sum_{t,t'} w_{tt'}(y_j(t) - y_j(t'))^2}{\sum_{t,t'} w_{tt'}} \text{ (objective function)} \tag{5}$$

$$\sum_t v_t y_j(t) = 0 \text{ (zero mean)}, \tag{6}$$

$$\frac{\sum_t v_t (y_j(t))^2}{\sum_t v_t} = 1 \text{ (unit variance)}, \tag{7}$$

$$\sum_t v_t y_j(t) y_{j'}(t) = 0, \text{ for } j' < j \text{ (decorrelation)}. \tag{8}$$

By defining the graph, the explicit feature information of the labels is made implicit in the graph structure. This permits the use of unsupervised learning, which has the great advantage that training can be done in a hierarchical network where no useful labels are available at intermediate layers, and the credit assignment problem is avoided.

3 Feature Extraction with SFA

3.1 Estimating Driving Forces

One problem in the analysis of nonlinear dynamical systems is that some parameters of the system might change over time, so that the system is actually not stationary [24]. Such parameters are called *driving forces*. If they vary slowly compared to the dynamics of the system, SFA can be used to estimate them [29]. As an example, consider an iterative tent map $f(w, \gamma)$, see Fig. 3. A time series is created by starting with an arbitrary value $w_0 \in [0, 1]$ and then applying the tent map repeatedly, i.e. $w_1 = f(w_0, \gamma_0)$, $w_2 = f(w_1, \gamma_1)$, ..., $w_{i+1} = f(w_i, \gamma_i)$, ..., with time index i and γ_i a given but slowly varying driving force cyclically shifting the tent map within $[0, 1]$.

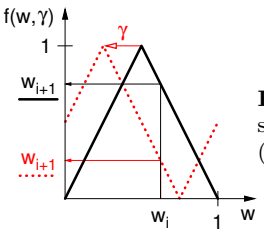

Fig. 3 Iterated tent map in null position (solid line) and shifted by γ (dotted line).

Since this signal is one-dimensional, one cannot apply SFA directly but has to apply time-embedding (see also [19]), i.e. one treats ten successive time points as one input vector, $\mathbf{x}(i) := (w_{i-4}, w_{i-3}, w_{i-2}, ..., w_{i+5})^T$. Time still increases in steps of one, so that two successive input vectors have nine values in common.

Applying SFA with polynomials of order 3 to this input signal yields a slow component that correlates well with the underlying driving force. If means are removed, variances are normalized to one, and the sign is chosen correctly (these three values cannot be recovered in any case), the first SFA output is somewhat noisy but otherwise follows the driving force accurately, see Fig. 4. Notice that this has been achieved without any knowledge of the dynamical system or the driving force itself.

3.2 Blind Source Separation

The term *blind source separation* refers to the problem of extracting unknown sources from an unknown mixture. A didactic example often given is the case of several persons in a room talking simultaneously, and being recorded by the same number of microphones. Each microphone then has a different mixture of all speakers. Thus in principle, it is possible to extract

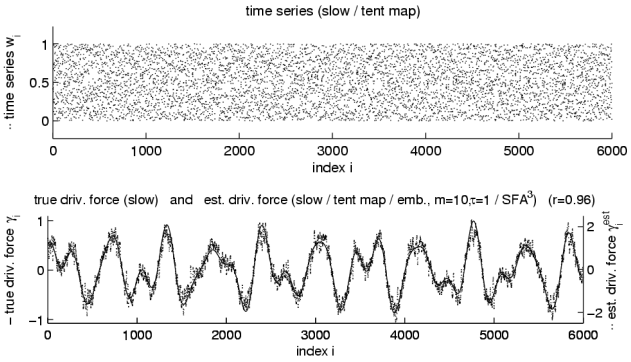

Fig. 4 Time series of the tent map (top), true driving force (bottom, solid line) and estimated driving force (bottom, dots). The correlation between the latter two is $r = 0.96$. Reproduced from [29].

each single speaker's voice from the collective recording, if one knew the details of the mixing (and ignoring echoes and time delays). If the mixture is linear, statistical independence is a sufficient optimization criterion to find the correct unmixing, assuming that the sound waves of the speakers are independent of each other. The corresponding algorithm is referred to as independent component analysis (ICA) [15].

ICA, at least in the more common instantaneous form, is not sufficient if the mixture is nonlinear [16]. However, unmixing can still be achieved if one takes into account the temporal structure of the signals (assuming they have some and are not iid). As shown in [26], SFA offers one way of doing this (see [12] for a related approach). This is possible for two reasons: Firstly, if one distorts a signal nonlinearly, the resulting signal usually varies more quickly. For instance, if you square a sine wave, you get frequency doubling. Secondly, if you mix two signals, the mixture varies more quickly than the slower of the two original signals. These two properties together guarantee that after sufficient nonlinear expansion, and if the original sources are somewhere in the expanded space, SFA will find the slowest of them (or a closely related version of it). Once that source is found, one can project it out of the expanded signal and find the next source, etc. See Fig. 5 for an example. Again this is achieved without any knowledge about the signals (apart from a smoothness assumption) or the mixture. We call this procedure extended SFA (xSFA).

3.3 General Purpose Feature Extraction

The two applications above were specific examples of unsupervised feature extraction, where the features of interest were well defined. However, SFA can be applied

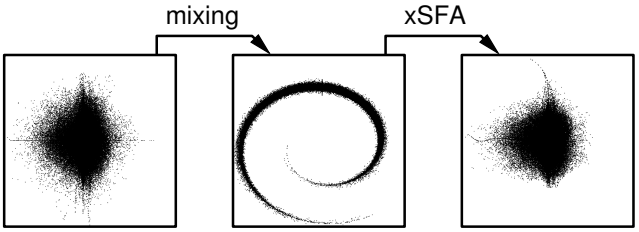

Fig. 5 An example of nonlinear blind source separation with xSFA. Left: A scatter plot of two sound sources. Middle: Nonlinear mixture. Right: Scatter plot of estimated sources. The correlation between the estimated and the true sources is above 0.9 in about 93% of the cases if tested on different sound sources.

also to less well defined situations. In [6] linear SFA was applied to *EEG data* recorded with 63 electrodes during an auditory discrimination task. Fisher discriminant analysis was then applied to the extracted features to do the classification between two auditory stimuli. The system was therefore able to extract the human auditory percept from the EEG recording, which is an example of a brain computer interface. A similarly unspecific feature extraction with SFA was used in [14] for the classification of *humanoid robot postures*.

4 Dimensionality Reduction with SFA

Feature extraction with SFA from high-dimensional input was used in [21], where the input was a sequence of 155×155 pixel images showing one out of two animated fish and two objects. Each fish had a different object as a target, and the other object was a distractor. The swimming of the fish was controlled by a reinforcement learning system, which had to learn to recognize the type of fish present and lead it to its target while avoiding the distractor. Reinforcement learning is generally notoriously slow on high-dimensional input. It is therefore mandatory to reduce the dimensionality by some means. In this example SFA was able to reduce the dimensionality from 24,025 down to 32 while still preserving all information necessary to solve the task.

5 Classification with SFA

In the applications above SFA was used as a purely unsupervised learning algorithm. We now consider the extension for supervised learning, first for classification and later for regression.

The first example of classification with SFA was on *handwritten digits* from the MNIST database [2]. Random pairs of training samples from the same class were connected to form mini-sequences of length two, and SFA with polynomials of degree 3 was trained on the collection of these mini-sequences. A Gaussian classifier was applied to the nine slowest features extracted to do the final classification. Error rate was 1.5%, which is close to the 0.95% achieved by LeNet-5, a hierarchical special-purpose architecture for digit recognition. The same approach was also applied to *human gesture recognition* [18] and a similar approach to *monocular road segmentation* [20].

So far SFA was applied in its original formulation (1–4) to one or a set of input sequences. We now present an application of the more general formulation with graph structures (5–8) to the German *traffic sign recognition* benchmark (GTSRB) [27], in which the goal was to classify photographs of traffic signs taken from a car traveling on German roads, which is of major interest for the development of driver assistance systems. The database consisted of 26,640 training and 12,569 test images of 43 different types of traffic signs. The position of the signs in the images was known and precomputed HOG-features (histograms of oriented gradients) were provided as well. For this classification problem, we created a clustered training graph and used the generalized SFA algorithm, see Section 2.5. Training SFA was done on the HOG features: First, linear SFA was used to reduce the dimensionality, followed by nonlinear SFA to extract the slowest features. Ideally these are step functions with constant values for all signs of the same class and different values for different classes, leading to one cluster per class in output space. A Gaussian classifier was then trained on these clusters to do the classification. This system ranked 8th place out of 24 groups in the GTSRB online competition with a performance of 96.4%, whereas human performance was 98.8% and the best algorithm achieved 99.0% recognition rate.

An elaborated system for *human action recognition* was proposed in [32]. In this case, SFA was applied to cuboids, i.e. subsequences within localized regions, extracted from video sequences of subjects performing various actions, such as walking, jogging, hand clapping, etc. Three supervised learning strategies were proposed, which all exploit the fact that SFA learns features that vary slowly for the action used for training. Changing the type of action during testing typically produces outputs that change much faster than sequences of the same action used for training. It was shown that SFA achieves comparable or even better performance than previous methods.

6 Regression with SFA

6.1 Age and Gender Estimation

Human-Computer Interaction benefits greatly from elementary information about the interacting subjects, such as their age, gender, and mimic expression. However, in particular automatic age estimation is a challenging problem, since aging results in only subtle changes compared to variations due to different identity.

Here we present a four-layer hierarchical SFA network [7] (see Fig. 2) that estimates age and gender from frontal static face images of size 135×135-pixels showing artificial subjects, created with special software for 3D face modeling and rendering. For age, the training and test images varied from 16 to 65 years, while gender, racial background, and identity were chosen randomly. For gender, a similar database was created and the same algorithm was applied. This was possible because gender was given as a continuous variable from -3 (very feminine) to $+3$ (very masculine) rather than a binary one (male vs. female). In both cases the images were ordered to produce a sequence of images across which the relevant parameter – age or gender – changes most slowly. For computational efficiency, images of similar age or gender were grouped. This resulted in a special graph structure in which images of a group were fully connected with each other and with images of neighboring groups (cf. Section 2.5). After training the SFA network, the first three outputs were used for training a Gaussian classifier to estimate the age or gender group. This yields *a posteriori* group probabilities from which an expectation value can be computed as the final estimate.

In both cases, good performance was achieved (on test data), with an RMSE of 3.8 years for age and 0.33 units for gender, compared to a chance level of 13.8 years and 1.73 units, respectively. As expected, age estimation was more difficult than gender. Interestingly, best performance was achieved with a linear SFA network, outperforming various (more complex) nonlinear networks. One reason might be that the number of training images was insufficient (only 4140/10800 images for age/gender) to train a nonlinear network, but it might also be that the rendering software uses a too simple model for age and gender. The recent release of large publicly available databases with age/gender labels make tests on real photographs possible in the near future.

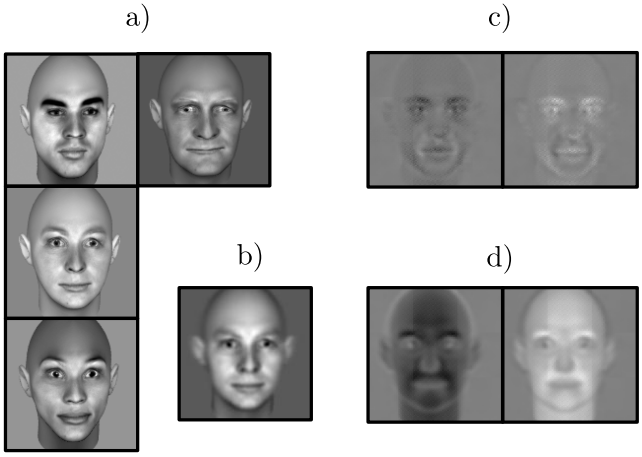

Fig. 6 a) Sample images for age estimation, b) average image c) image variation that specifically activates the slowest feature extracted for age (right, its negative) d) image variation that specifically activates the slowest feature for gender (right, its negative). Notice how the image variation for gender resembles a masculine face, whereas its negative resembles a feminine one.

6.2 Face Detection

Systems for face detection from images are becoming very popular due to advances in computing power and specialized algorithms capable of running even on portable devices. Face detection is also a prerequisite for tasks such as tracking or age and gender estimation, as described above. Although face detection is popular, robustness is still a challenge in *uncontrolled* scenarios, in particular in the presence of facial hair, image artifacts, strong or unusual lighting conditions, and/or partial occlusion.

We have shown above how a hierarchical SFA network can be used to estimate a continuous parameter such as age or gender from a facial image. This approach can also be applied to other parameters such as x-position, y-position, or scale. Using three separate networks trained like this, potential faces can be normalized as follows: Estimate x-position with the first network and center the face horizontally; estimate y-position with the second network and center the face vertically; estimate scale and resize the image. A fourth network can be trained to estimate the quality of the normalization, again as a continuous parameter, and to indicate whether a face is present at all. To improve quality, this process is repeated three times in succession, leading to increasingly accurate and reliable localization. Finally, eye positions are determined in the expected regions within the normalized face with an eye specific SFA network. To detect several faces, an image is first tiled with overlap into many candidate regions of different sizes, and the algorithm above is then ap-

plied to each region separately. Thus, most candidate regions do not contain a face at all, which is the reason why the fourth network is necessary to decide on the presence of a face.

In this example application, 40,000 frontal face photographs from different sources were used for training. The networks applied are improved versions of the ones for age and gender estimation, and were redesigned with a 9-layer structure characterized by a very small fan-in in all layers, which is particularly useful to reduce overfitting. Furthermore, different nonlinear expansions are introduced, offering better performance and still low expanded dimensionality and good generalization.

Performance of the resulting system on various image databases was competitive [23] and yielded a detection rate on grayscale photographs from 71.5% to 99.5% depending on the difficulty of the test images.

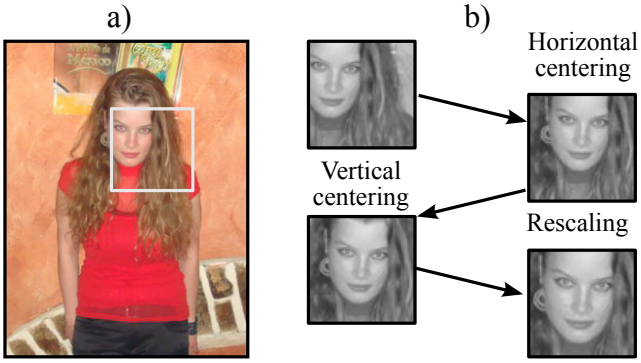

Fig. 7 a) Test image. A single image region containing a face is shown b) Processing of the region: initial region, horizontal centering, vertical centering, and rescaling.

7 Conclusion

Slow feature analysis is an unsupervised learning algorithm based on the slowness principle. It extracts features from a temporal input signal that vary as slowly as possible over time without resorting to averaging or low-pass filtering. It thereby learns stable and robust representations invariant to frequently occurring but usually irrelevant variations. Although SFA has originally been developed within the field of computational neuroscience to model the primate visual system, the algorithm is computationally efficient and suitable for technical applications in machine learning and computer vision. It is easy to implement and straightforward to use, since it is basically parameter free. In standard SFA the only choice is that of the function space used, i.e. the type of nonlinearity. In hierarchical SFA one also has to define a network structure and decide on the number of SFA outputs passed to the next layer. Of great help in using SFA is the fact that it is well understood analytically. Optimal free responses, which are not constrained by input or function space, are known [30] and can be exploited to design the training procedure and the analysis of the output signals, see [10] for a complex example.

The applications reviewed in this paper demonstrate the versatility of SFA and its potential use in a variety of different problem cases. We believe that its robustness and flexibility make SFA a useful general purpose preprocessing tool for feature extraction and dimensionality reduction.

In this review we put some emphasis on supervised learning problems on high-dimensional data. Such problems require hierarchical processing, which is typically difficult to train in a supervised fashion. We propose to convert such problems to unsupervised learning problems by defining graphs that reflect the similarity relationships between the labels of the training samples. SFA can then be applied in an unsupervised fashion and a hierarchical network with a well defined objective function on all levels can be trained. This avoids the credit assignment problem that supervised learning algorithms often have in hierarchical networks. Only a simple final supervised step is required to compute the label values from the final SFA output.

Hierarchical processing is advantageous for high-dimensional input data for several reasons. For instance, by breaking down the computation into smaller parts, the computational effort becomes manageable. Furthermore, each node in the network gets a much lower-dimensional input, so that less training data is required and generalization is improved.

Current research in our group focusses on better understanding the translation of a supervised learning problem into a learning problem suited for SFA, on finding suitable function spaces providing sufficient nonlinearity while keeping dimensionality low, and on improving generalization by using very deep networks with small fan-in.

References

1. Berkes P (2005) Handwritten digit recognition with nonlinear fisher discriminant analysis. In: ICANN, Lecture Notes in Computer Science, vol 3697, Springer Berlin / Heidelberg, pp 285–287
2. Berkes P (2005) Pattern recognition with slow feature analysis. Cognitive Sciences EPrint Archive (CogPrints), URL `http://cogprints.org/4104/`
3. Berkes P, Wiskott L (2005) Slow feature analysis yields a rich repertoire of complex cell properties. J Vis 5(6):579–602
4. Blaschke T, Berkes P, Wiskott L (2006) What is the relationship between slow feature analysis and independent

component analysis? Neural Computation 18(10):2495–2508
5. Bray A, Martinez D (2003) Kernel-based extraction of slow features: Complex cells learn disparity and translation invariance from natural images. In: NIPS, MIT Press, Cambridge, MA, vol 15, pp 253–260
6. Dähne S, Höhne J, Schreuder M, Tangermann M (2011) Slow feature analysis - a tool for extraction of discriminating event-related potentials in brain-computer interfaces. In: ICANN, Lecture Notes in Computer Science, vol 6791, Springer Berlin / Heidelberg, pp 36–43
7. Escalante A, Wiskott L (2010) Gender and age estimation from synthetic face images with hierarchical slow feature analysis. In: Hüllermeier E, Kruse R (eds) International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU'10), pp 240–249
8. Escalante A, Wiskott L (2011) Heuristic evaluation of expansions for Non-Linear Hierarchical Slow Feature Analysis. In: Proc. The 10th Intl. Conf. on Machine Learning and Applications (ICMLA'11), IEEE Computer Society, Los Alamitos, CA, USA, pp 133–138
9. Földiák P (1991) Learning invariance from transformation sequences. Neural Computation 3(2):194–200
10. Franzius M, Sprekeler H, Wiskott L (2007) Slowness and sparseness lead to place, head-direction, and spatial-view cells. PLoS Computational Biology 3(8):e166
11. Franzius M, Wilbert N, Wiskott L (2011) Invariant object recognition and pose estimation with slow feature analysis. Neural Computation 23(9):2289–2323
12. Harmeling S, Ziehe A, Kawanabe M, Müller KR (2003) Kernel-based nonlinear blind source separation. Neural Computation 15:1089–1124
13. Hinton GE (1989) Connectionist learning procedures. Artificial Intelligence 40(1-3):185–234
14. Höfer S, Hild M, Kubisch M (2010) Using slow feature analysis to extract behavioural manifolds related to humanoid robot postures. In: Tenth International Conference on Epigenetic Robotics, pp 43–50
15. Hyvärinen A (1999) Survey on independent component analysis. Neural Computing Surveys 1:94–128
16. Hyvärinen A, Pajunen P (1999) Nonlinear independent component analysis: Existence and uniqueness results. Neural Networks 12:429–439
17. Klampfl S, Maass W (2010) A theoretical basis for emergent pattern discrimination in neural systems through slow feature extraction. Neural Computation 22(12):2979–3035
18. Koch P, Konen W, Hein K (2010) Gesture recognition on few training data using slow feature analysis and parametric bootstrap. In: Neural Networks (IJCNN), The 2010 International Joint Conference on, pp 1 –8
19. Konen W, Koch P (2011) The slowness principle: SFA can detect different slow components in nonstationary time series. International Journal of Innovative Computing and Applications (IJICA) 3(1):3–10
20. Kuhnl T, Kummert F, Fritsch J (2011) Monocular road segmentation using slow feature analysis. In: Intelligent Vehicles Symposium (IV), 2011 IEEE, pp 800 –806
21. Legenstein R, Wilbert N, Wiskott L (2010) Reinforcement learning on slow features of high-dimensional input streams. PLoS Comput Biol 6(8):e1000,894
22. Mitchison G (1991) Removing time variation with the anti-hebbian differential synapse. Neural Computation 3(3):312–320
23. Mohamed NM, Mahdi H (2010) A simple evaluation of face detection algorithms using unpublished static images. In: 10th International Conference on Intelligent Systems Design and Applications, ISDA, pp 1–5
24. Schreiber T (1999) Interdisciplinary application of nonlinear time series methods. Physics Reports 308(1):1–64
25. Sprekeler H, Wiskott L (2011) A theory of slow feature analysis for transformation-based input signals with an application to complex cells. Neural Computation 23(2):303–335
26. Sprekeler H, Zito T, Wiskott L (2010) An extension of slow feature analysis for nonlinear blind source separation. Cognitive Sciences EPrint Archive (CogPrints), URL `http://cogprints.org/7056/`
27. Stallkamp J, Schlipsing M, Salmen J, Igel C (2011) The German Traffic Sign Recognition Benchmark: A multiclass classification competition. In: International Joint Conference on Neural Networks
28. Vollgraf R, Obermayer K (2006) Sparse optimization for second order kernel methods. In: Proc. IJCNN'06, pp 145–152
29. Wiskott L (2003) Estimating driving forces of nonstationary time series with slow feature analysis. arXiv.org e-Print archive, URL `http://arxiv.org/abs/cond-mat/0312317/`
30. Wiskott L (2003) Slow feature analysis: A theoretical analysis of optimal free responses. Neural Computation 15(9):2147–2177
31. Wiskott L, Sejnowski T (2002) Slow feature analysis: Unsupervised learning of invariances. Neural Computation 14(4):715–770
32. Zhang Z, Tao D (2012) Slow feature analysis for human action recognition. Pattern Analysis and Machine Intelligence, IEEE Transactions on 34(3):436 –450

Alberto N. Escalante-B. received his BEng in Computer Engineering from the National Autonomous University of Mexico, in 2003, and an MSc degree in Computer Science from the University of Saarland, Saarbrücken, Germany, in 2008. Currently, he is a research assistant at the Institut für Neuroinformatik at the Ruhr-Universität Bochum. His research interests include face detection and recognition, age and gender estimation from face images, slow feature analysis and hierarchical data processing.

Laurenz Wiskott is full professor at the Ruhr-Universität Bochum, Germany. He holds a Diploma degree in Physics from the Universität Osnabrück and a PhD from the Ruhr-Universität Bochum. The stages of his career include The Salk Institute in San Diego, the Institute for Advanced Studies in Berlin, and the Institute for Theoretical Biology, Humboldt-Universität Berlin. He has been working in the fields of Computer Vision, Neural Networks, Machine Learning and Computational Neuroscience.