Heuristic Evaluation of Expansions for Non-Linear Hierarchical Slow Feature Analysis

Alberto N. Escalante-B.
Institut für Neuroinformatik
Ruhr-University of Bochum, Germany
e-mail: alberto.escalante@ini.rub.de

Laurenz Wiskott
Institut für Neuroinformatik
Ruhr-University of Bochum, Germany
e-mail: laurenz.wiskott@ini.rub.de

Abstract—**Slow Feature Analysis (SFA) is a feature extraction algorithm based on the slowness principle with applications to both supervised and unsupervised learning. When implemented hierarchically, it allows for efficient processing of high-dimensional data, such as images. Expansion plays a crucial role in the implementation of non-linear SFA. In this paper, a fast heuristic method for the evaluation of expansions is proposed, consisting of tests on seven problems and two metrics. Several expansions with different complexities are evaluated. It is shown that the method allows predictions of the performance of SFA on a concrete data set, and the use of normalized expansions is justified. The proposed method is useful for the design of powerful expansions that allow the extraction of complex high-level features and provide better generalization.**

I. Introduction

One important problem in machine learning is the development of algorithms for dimensionality reduction (cf. [1]). Noticeable examples of algorithms for *linear* dimensionality reduction (or linear transformations) are Principal Component Analysis (PCA), Independent Component Analysis (ICA), and Linear Discriminant Analysis (LDA) [2]. These algorithms can be applied to several other tasks, such as data compression, signal unmixing, prediction, classification, and regression. Their popularity and effectiveness is remarkable given that they all rely on a simple linear model, and all of them can be regarded as feature extraction algorithms that *optimally* solve a particular feature extraction problem when restricted to a *linear* feature space.

In many realistic scenarios, feature extraction problems (e.g., dimensionality reduction) cannot be satisfactorily solved by linear models. Thus, several non-linear variations of the algorithms above have been proposed [1]. For instance, in the field of image processing, feature extraction from an object is generally a non-linear problem if non-homogeneous backgrounds are used and if invariance to the position of the object is desired. Pose-normalization and other types of preprocessing are helpful to reduce the complexity of the underlying problem, but do not guarantee linearization.

Slow Feature Analysis (SFA) [3] is a promising learning algorithm for feature extraction inspired by biology (i.e., by the visual system) based on the slowness principle (cf. [4]). Given a multi-dimensional signal as a function of time, the objective of SFA is to extract those features that change as slowly as possible over time, while feature extraction is done instantaneously. These features are called *slow features* and additionally satisfy a few normalization and decorrelation restrictions (see Section III-A).

Hierarchical SFA is one form of implementing SFA that allows processing high-dimensional data such as images. In this paper, we study the design of various types of non-linear expansions that are particularly suitable for hierarchical SFA. We concentrate on the heuristic evaluation of non-linear expansions that allow complex and effective feature extraction (dimensionality reduction) while at the same time offering resistance against the frequent problems of overfitting and outlier amplification (see Section III-E). For this purpose, we make use of a theoretical analysis on SFA [5] and of empirical observations gained after long lasting experimentation with the algorithm at solving several feature extraction problems. The proposed method might be also applicable to non-linear hierarchical implementations of PCA, ICA, or LDA.

II. Previous Work

SFA is typically a non-linear algorithm that can be implemented in two ways. The most sophisticated one relies on the introduction of kernels [6], [7] and is possible due to the kernel trick. This allows the consideration of large or even infinite feature spaces. In practice, however, kernel SFA has been used less. This might be explained by the difficulty of choosing a good kernel or by high computational costs for large data sets. The second way, which is the one followed in this paper, consists in the application of linear SFA after the data has been transformed by an explicit non-linear expansion [3]. A widely used expansion is the Quadratic Expansion ($QExp$), as follows: Given some data sample $\mathbf{x} = (x_1, x_2, \dots, x_I)$, $QExp(\mathbf{x}) \stackrel{\text{def}}{=} (I(\mathbf{x}), QM(\mathbf{x}))$, that is, $QExp$ is the vectorial concatenation of $I(\mathbf{x}) \stackrel{\text{def}}{=} (x_1, x_2, \dots, x_I)$, the identity function, and $QM(\mathbf{x}) \stackrel{\text{def}}{=} (x_1^2, x_1 x_2, \dots, x_1 x_I, x_2^2, \dots, x_2 x_I, x_3^2, \dots, x_I^2)$, all the quadratic monomials, (i.e., all the products of pairs of elements of $\mathbf{x}$). Higher degree expansions can be defined analogously. For example, $CExp(\mathbf{x}) \stackrel{\text{def}}{=} (I(\mathbf{x}), QM(\mathbf{x}), CM(\mathbf{x}))$ is a cubic expansion, where CM are all cubic monomials.

We now review previous architectures based on SFA, the expansions that were used, and the measures employed to reduce overfitting. Franzius et al. [8] developed a hierarchical

SFA network capable of learning pose and object identity from synthetic fish images. The expansion $QExp$ was used, and a clipping step was introduced to keep signal amplitudes within the range $(-4, 4)$. Moreover, Gaussian noise was added to improve generalization.

Berkes et al. [4] analyzed sequences of pairs of patches taken from natural images with (quadratic) SFA. It was demonstrated that the features extracted by SFA show the same properties as complex cells found in the visual cortex (e.g., direction selectivity). The structure of their system was: dimensionality reduction with PCA, followed by $QExp$ and SFA. Overfitting was avoided through the preprocessing step with PCA and through the use of a large amount of training data (about 400,000 samples).

Escalante et al. [9] developed a linear and a non-linear hierarchical network for SFA. Both networks were capable of accurately estimating the age and gender of artificial subjects generated from 3-D face models under constant pose and expression. A new expansion for the non-linear network was proposed: $\mathit{0.8Exp}(\mathbf{x}) \stackrel{\text{def}}{=} (I(\mathbf{x}), |\mathbf{x}|^{0.8}) = (x_1, x_2, \dots, x_I, |x_1|^{0.8}, |x_2|^{0.8}, \dots, |x_I|^{0.8})$. This simple expansion overcame the problem of uncontrolled outlier amplification without resorting to clipping. However, the linear network still outperformed the non-linear one on test data due to insufficient training data.

Later, Escalante et al. (cf. [10, Appendix A]) developed SFA networks with expansions similar to *0.8Exp* and applied them to various feature extraction problems on frontal face photographs, such as the estimation of position and size of a face in an image patch. This was consolidated in a system for face detection. For the problems addressed, linear SFA gave poor performance. Non-linear SFA, however, performed much better, confirming that these problems are highly non-linear and showing that even a modest expansion repeated in several layers is capable of untangling the underlying problem.

III. Linear and Non-Linear SFA

A. SFA Optimization Problem

The SFA problem [3], [4] can be stated as follows. Given an I-dimensional signal $\mathbf{x}(t) = (x_1(t), \dots, x_I(t))^T$, find a set of real-valued instantaneous functions $\mathbf{g}(\mathbf{x}) = (g_1(\mathbf{x}), \dots, g_J(\mathbf{x}))$ lying in a function space $\mathcal{F}$ so that for each component $y_j(t)$ of the output signal $\mathbf{y}(t) \stackrel{\text{def}}{=} \mathbf{g}(\mathbf{x}(t))$, for $1 \leq j \leq J$, the following holds:

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

under the constraints

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

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

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

The constraints (2–4) assure that the output signals are not constant and code different information of the input signal. Notice that the delta value $\Delta(y_j)$ is defined as the average energy of the temporal derivative of y_j and is therefore a measure of the temporal variation of such signal. The problem is solved iteratively beginning with y_1 and ending with y_J.

In some theoretical analysis (e.g., [5]), it is common to let $\mathbf{g}$ belong to an unrestricted function space. In practice, $\mathbf{g}$ is often limited to the function space of all quadratic functions.

B. Linear SFA Algorithm

The SFA algorithm [3] takes advantage of the fact that the optimal *linear* solutions to the optimization problem only depend on the covariance matrix $\mathbf{C} \stackrel{\text{def}}{=} \langle \mathbf{x}\mathbf{x}^T \rangle$ of the training signal $\mathbf{x}(t)$, where the mean of $\mathbf{x}(t)$ was removed, and on the second-moment matrix $\dot{\mathbf{C}} \stackrel{\text{def}}{=} \langle \dot{\mathbf{x}}\dot{\mathbf{x}}^T \rangle$ of the time derivative $\dot{\mathbf{x}}(t)$ of the training signal. In practice, discrete time is used. Therefore, the training signal becomes a sequence of samples, and $\dot{\mathbf{x}}$ is approximated by the difference of consecutive samples: $\dot{\mathbf{x}}(t) \stackrel{\text{def}}{=} \mathbf{x}(t+1) - \mathbf{x}(t)$. The matrix $\mathbf{C}$ is consequently approximated by its corresponding sample covariance matrix: $\mathbf{C} = \frac{1}{N-1} \sum_{t=1}^{N} (\mathbf{x}(t) - \bar{\mathbf{x}})(\mathbf{x}(t) - \bar{\mathbf{x}})^T$, and $\dot{\mathbf{C}}$ is computed as $\dot{\mathbf{C}} = \frac{1}{N-1} \sum_{t=1}^{N-1} (\mathbf{x}(t+1) - \mathbf{x}(t))(\mathbf{x}(t+1) - \mathbf{x}(t))^T$.

In the linear function space, the outputs can be written as $y_j(t) = g_j(\mathbf{x}(t)) = \mathbf{w}_j^T \mathbf{x}$, and the SFA problem can be reduced to a generalized eigenvalue problem (cf. [4]). That is, weights $\mathbf{W} = \{\mathbf{w}_j\}$ are computed such that $\mathbf{W}^T \mathbf{C} \mathbf{W} = \mathbf{I}$ and $\dot{\mathbf{C}} \mathbf{W} = \mathbf{W} \mathbf{\Lambda}$, where $\mathbf{\Lambda}$ is a diagonal matrix with diagonal $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_J$. Hence, efficient algorithms for solving the latter problem can be used, and the SFA algorithm has a complexity similar to PCA.

C. Hierarchical Slow Feature Analysis

For high-dimensional data, the direct use of linear SFA might be too expensive since it has a computational complexity of $\mathcal{O}(NI^2 + I^3)$ where N is the number of samples and I is the input dimensionality. An expansion would make this problem even more severe. Hierarchical SFA is a greedy implementation of SFA that allows to cope with high-dimensional data by dividing it in lower-dimensional chunks (e.g., 2 chunks of N samples of dimensionality $I/2$) that are separately processed by SFA instances, called *nodes*, in a layer, cf. [11]. Afterwards, the resulting slow signals computed by each node are grouped together and further processed by an SFA node in the next layer. This process can be repeated in a cascade and the nodes can be organized in a multi-layer network, where each layer is trained separately from the bottom to the top, and the top layer contains a single node.

D. Non-Linear Hierarchical SFA Networks

In a non-linear SFA network each (linear) SFA node is preceded by an expansion node. The increase in dimensionality of the data chunk depends on the expansion that was applied, which might be different for each SFA node. The number of slow signals preserved by the SFA nodes is kept similar as in the linear case to avoid an explosion in their number.

In principle, the larger the expansion is, the more complex the features learned by SFA might be and, thus, a large expansion might allow the extraction of slower features. Ideally,

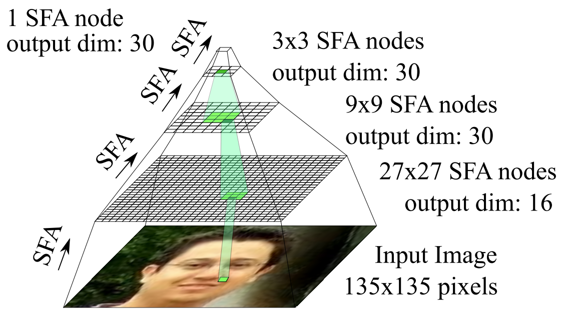

Fig. 1. Example of a hierarchical network, where the inputs are 135 by 135-pixel images and the outputs are 30 slow features.

it would be desirable to allow full polynomial expansion of the data, and let linear SFA find an appropriate linear combination of the expanded data. The feature space available to SFA would contain all polynomial functions, and could, thus, extract extremely complex non-linear features. On the other hand, large expansions drastically increase the number of dimensions of the data, and consequently, increase the number of free parameters in the model to be learned, which under normal conditions gives rise to overfitting and prohibitive computational costs.

E. Overfitting and Outlier Amplification

Overfitting is the over-adaptation of a model to the peculiarities of the training data as opposed to the true distribution of the data, resulting in poor performance on new test data. This problem is more evident when a large expansion is used or when the number of training samples is small. In the context of SFA, the performance can be measured in terms of the delta values $\Delta(y_j)$, for $1 \leq j \leq J$, of the extracted features.

Another recurrent problem found when using some expansions is *outlier amplification*, in which test samples less similar to the samples seen during training are mapped into even more atypical features, for example, by the quadratic terms of $QExp$. In hierarchical implementations, this problem accumulates across layers, resulting in an uncontrolled explosion of the magnitude of the signals for particular samples. For example, on test data we have frequently seen features with amplitudes of 10^{60}, whereas on training data the features have zero mean and unit variance and are rarely larger than 20 in amplitude. We remark that this problem only occurs in a fraction of the samples of the test data, and is directly related but different from the problem of overfitting. Clipping (e.g. [8]) can reduce this problem, but is a suboptimal solution because feature extraction is impaired due to information loss.

Overfitting and outlier amplification in the context of expansions can be measured with heuristics proposed in Section V.

IV. Expansions and Basis Functions

Let us assume that an SFA node is trained with data containing an underlying (hidden) slow parameter that monotonically increases as a function of time, and that the parameter can be accurately computed from each data sample (with an unknown arbitrarily complex function). Then, the theory of optimal free SFA responses [5] predicts that the slowest feature learned by SFA with an unrestricted function space is a half-cosine function (with arbitrary sign) $\pm\sqrt{2}\cos(\pi t/T)$, for $0 \leq t \leq T$. In principle, since such a signal is also monotonically increasing (or decreasing) in this interval, a mapping from the slowest signal to the slow parameter can be found. When restricted function spaces are used, such as in quadratic SFA, we cannot expect the extraction of the half-cosine function exactly, but we can expect a distorted and noisy version of this function.

When SFA is implemented as an expansion followed by linear SFA, linear SFA finds an optimal linear combination of the expanded signal, such that the slowness is minimized. Basically, the expansion is playing the role of a basis that defines a feature space, from which SFA linearly chooses the slowest possible function. If the expansion is rich enough, the slowest output of SFA will closely resemble the optimally slow function ideally found by unrestricted SFA.

Not all expansions are equally useful. This is a direct consequence of the richness of the different feature spaces defined by different expansions, and of the fact that in hierarchical networks the input to each node has a particular distribution. Recall that the input to an SFA node is the concatenation of the outputs of previous SFA nodes, which must satisfy normalization and decorrelation constraints (2–4). Thus, the definition of an expansion is mostly relevant where the probability density of the input is significant, and not on the whole possible input domain $\mathbb{R}^I$.

The polynomial expansion (polynomial basis), $PExp(\mathbf{x}) \overset{\text{def}}{=} (1, I(\mathbf{x}), QM(\mathbf{x}), CM(\mathbf{x}), \cdots)$, describes a great function space that allows fitting to any function at an arbitrary number of points. In practice, infinite expansions cannot be used, and they should be truncated. In this paper, we truncate by keeping functions up to a given degree. Clearly, $QExp$ and $CExp$ are just truncated versions of $PExp$. The main disadvantage of $PExp$ is that its functions (e.g. $x_1^2 x_2^3$) typically grow at a fast pace resulting in bad function fitting for test samples that are too different from training samples.

A. Generation of Expansions

(Truncated) polynomial expansions are very useful because they allow the derivation of various families of expansions with normalization properties that improve generalization. Here we propose two of them given an expansion Exp:

- Let $q(\mathbf{x})$ a positive scalar function. Then, $Exp(\mathbf{x})/q(\mathbf{x})$ is a (component-wise) normalized expansion. In this paper, we use $Exp \in \{QExp, CExp\}$ and $q_2(\mathbf{x}) = 1 + ||\mathbf{x}||^2$. Notice that $QExp(\mathbf{x})/q_2(\mathbf{x})$ includes components (e.g., $x_1^2/(1+||\mathbf{x}||^2)$) that at most saturate to 1 for input vectors $\mathbf{x}$ with a large norm and behave as $QExp$ for smaller $\mathbf{x}$.
- Let $m(\mathbf{x})$ be a bijective mapping of vectors. Then, $Exp(m(\mathbf{x}))$ is a useful expansion. In this paper we use $QExp(\mathbf{x}^{*0.4})$ and $CExp(\mathbf{x}^{*0.3})$, where $\mathbf{x}^{*k} \overset{\text{def}}{=} x_1^{*k}, \ldots, x_I^{*k}$, and $x_i^{*k} \overset{\text{def}}{=} \text{sgn}(x_i)|x_i|^k$ is an invertible sign-preserving exponential function of x_i, for $k > 0$.

Of course, more expansions can be defined by starting from expansions other than *QExp* or *CExp* and by using other functions for $q(\mathbf{x})$ or $m(\mathbf{x})$. The expansions evaluated in this paper are listed in Table I.

V. Heuristic Evaluation of Expansions

For concrete data, a good choice of an expansion for SFA (one achieving high slowness) depends on the peculiarities of the data (number of samples, dimensionality, type of data, etc.). Different expansions could be tested for each SFA node of a network to locally achieve top slowness. However, such an optimization procedure would be computationally expensive because it requires training each node for every particular expansion. Therefore, here we propose a heuristic method to evaluate expansions without requiring concrete training of SFA. This method is intended to aid in the design of good expansions for a great variety of natural problems.

In a concrete execution of SFA, when the data originates from a single monotonically increasing parameter, the outputs ordered in decreasing slowness are ideally equal to the optimal free responses (ignoring polarity w.l.o.g.): $c_1(t) = \sqrt{2}\cos(t)$, $c_2(t) = \sqrt{2}\cos(2t)$, $c_3(t) = \sqrt{2}\cos(3t)$, etc., for $0 \leq t \leq \pi$. That is, in an ideal case SFA computes cosine functions $\mathbf{c}(t) = (c_1(t), \dots, c_J(t))$ of increasing frequencies from the samples $\mathbf{x}(t)$. When appropriate expansions are employed, the computation of noisy versions of these signals has been empirically verified, particularly at the top nodes [8], [9].

In a simple model, we approximate the input signals of a node (e.g., extracted by a previous node) as $\tilde{x}_i(t) = a_i \cdot (c_i(t) + n_i(t))$, where a_i is a normalization constant assuring unit variance and $n_i(t)$ is Gaussian noise with variance σ_i^2 and zero mean. The input to a node in a network is typically the concatenation of the outputs of many nodes, but for simplicity we view the input as if it only originated from a single one. In practice, noise variance is smaller for the slower changing signals than for the faster changing ones. For simplicity, we consider here all variances σ_i^2 to be constant and equal to σ^2.

Roughly speaking, it can be conceived that an SFA network attempts to compute $\mathbf{c}(t)$ from the training data. This computation is performed on each layer yielding very crude estimations at the first layers and more accurate estimations at the top node. The signal c_1 (or a noisy version of it) is the most important one because it allows the computation of the slow parameter, and it is coded distributedly and redundantly in the signals $\tilde{x}_1, \dots, \tilde{x}_I$. Of course, $\tilde{x}_1$ provides more information about c_1 than any other component of the input, however, due to noise, the other components also contain useful information that might be exploited after a non-linear transformation to better approximate c_1. Thus, in hierarchical processing, the non-linearities should be powerful enough to allow the computation of versions of c_j, for $1 \leq j \leq J$, and particularly of c_1, as faithfully as possible from the input signals $\tilde{x}_1, \dots, \tilde{x}_I$.

Based on previous empirical observations, we postulate that a good expansion *Exp* is characterized by its capability to linearize a particular set of problems **P1–P7**, see below. By **P**: $(x_1, \dots, x_I) \mapsto y$ we denote the problem of approximating a given function $y(t)$ from the input data $\mathbf{x}(t) \overset{\text{def}}{=} (x_1(t), \dots, x_I(t))$. In the context of a particular expansion *Exp*, one approximates $y(t)$ as $\ell(t) \overset{\text{def}}{=} (1, Exp(\mathbf{x}(t))) \cdot \mathbf{w}$, where $\mathbf{w}$ is a weight column vector. The constant 1 compensates for feature means different from zero, and $\mathbf{w}$ is computed with linear regression to best fit y. The goal is to approximate $y(t)$ linearly as closely as possible from $Exp(\mathbf{x}(t))$. The quality of an approximation $\ell(t)$ is measured in terms of the Root Mean Squared Error (RMSE) between $y(t)$ and $\ell(t)$. We propose the following function approximation problems to test the computational richness of an expansion (from the point of view of SFA).

- **P1**: $\tilde{x}_1 \mapsto \tilde{x}_1$; (Identity function) That is, linearly approximate/compute $\tilde{x}_1$ from (the expansion of) $\tilde{x}_1$.
- **P2**: $\tilde{x}_1 \mapsto c_2$; (Frequency doubler) Compute c_2 from $\tilde{x}_1$.
- **P3**: $\tilde{x}_1, \tilde{x}_2 \mapsto c_3$; Approximate c_3 from $\tilde{x}_1$ and $\tilde{x}_2$.
- **P4**: $\tilde{x}_1 \mapsto \sin(t)$; Approximate $\sin(t)$ from a noisy version of $\cos(t)$, where $0 \leq t \leq \pi$.
- **P5**: $\tilde{x}_2, \tilde{x}_3 \mapsto c_1$; Approximate c_1 from higher frequency noisy harmonics $\tilde{x}_2$ and $\tilde{x}_3$.
- **P6**: $\tilde{x}_2, \tilde{x}_3, \tilde{x}_4 \mapsto c_1$. Similar to **P5**.
- **P7**: $\tilde{x}_3, \tilde{x}_4, \tilde{x}_5 \mapsto c_1$. Also similar to **P5**.

Each one of these problems is motivated by specific reasons that might facilitate the extraction of slow features. Suppose that a particular node extracts a very slow signal, then a good performance on **P1** ensures that a subsequent SFA node in the network is at least able to preserve such a signal after expansion. A good performance on **P2** and **P3** indicates that a node is capable of generating (or reducing the noise of) higher frequency harmonics given access to lower frequency ones. Thus, improving the approximation of later components of the ideal output $\mathbf{c}$. **P4** ensures that the output signals could be cleared from sine components, if needed. ($\mathbf{c}$ is free from sine components.) A good performance on **P5**, **P6**, and **P7** indicates that lower frequency (slower) harmonics might be generated (or its noise reduced) from higher frequency ones.

The problems above are representative and subsume a larger number of problems, for instance $\tilde{x}_1 \mapsto \tilde{x}_1$ is equivalent to $\tilde{x}_i \mapsto \tilde{x}_i$, whereas $\tilde{x}_2, \tilde{x}_3 \mapsto c_1$ is equivalent to $\tilde{x}_{2\cdot i}, \tilde{x}_{3\cdot i} \mapsto c_i$, and $\tilde{x}_1 \mapsto \sin(t)$ is equivalent to $\tilde{x}_i \mapsto \sin(it)$, where $i \in \mathbb{N}$.

In the context of SFA, *overfitting* might be defined as the difference (or ratio) between the delta values of the slowest signals extracted from training and test data, but computing this requires training of SFA. We propose a simple and practical way of measuring overfitting for non-linear expansions. The basic idea is to use the expanded data to attempt to linearly learn a *random* zero-mean signal. Since the labels are random, the best fit is to always output the constant 0 for any data sample. A zero complexity algorithm achieves this by always yielding 0. Thus, any deviation of the learned output from 0 for training and test data is purely due to overfitting and is easy to measure.

More concretely, let $\tilde{\mathbf{x}}(t)$ be the simulated input to SFA (before expansion), and let $s(t) = \Pi(c_1(t))$ be a randomly time-

Name	Definition	Example of functions contained	Expanded dimensionality
Identity	$I(\mathbf{x})$	x_1, x_2, x_I	I
QExp	$(I(\mathbf{x}), QM(\mathbf{x}))$	$x_1, x_2, x_I, x_1^2, x_1x_2, x_2^2, x_I^2$	$I(I+3)/2$
CExp	$(I(\mathbf{x}), QM(\mathbf{x}), CM(\mathbf{x}))$	$x_1, x_I, x_1^2, x_1x_2, x_I^2, x_1^3, x_1x_2x_3, x_3^2x_4, x_I^3$	$I(I^2+6I+11)/6$
$QExp/q_2$	$q_2(\mathbf{x}) = (1+\|\|\mathbf{x}\|\|^2)$	$x_1/q_2(\mathbf{x}), x_1x_2/q_2(\mathbf{x}), x_I^2/q_2(\mathbf{x})$	$I(I+3)/2$
$CExp/q_3$	$q_3(\mathbf{x}) = (1+\|\|\mathbf{x}\|\|^2)$	$x_1/q_3(\mathbf{x}), x_1x_2/q_3(\mathbf{x}), x_I^2/q_3(\mathbf{x}), x_I^3/q_3(\mathbf{x})$	$I(I^2+6I+11)/6$
$QExp^{*0.4}$	$QExp(\mathbf{x}^{*0.4})$	$x_1^{*0.4}, x_2^{*0.4}, \|x_1\|^{0.8}, (x_1x_2)^{*0.4}, \|x_I\|^{0.8}$	$I(I+3)/2$
$CExp^{*0.3}$	$CExp(\mathbf{x}^{*0.3})$	$x_1^{*.3}, x_2^{*0.3}, \|x_1\|^{0.6}, (x_1x_2)^{*0.3}, x_1^{*0.9}, (x_1x_2x_3)^{*0.3}, x_I^{*0.9}$	$I(I^2+6I+11)/6$
0.8Exp	$(I(\mathbf{x}), \|\mathbf{x}\|^{0.8})$	$x_1, x_2, x_I, \|x_1\|^{0.8}, \|x_2\|^{0.8}, \|x_I\|^{0.8}$	$2I$
SExp	$(I(\mathbf{x}), \mathbf{x}^2)$	$x_1, x_2, x_I, x_1^2, x_2^2, x_I^2$	$2I$

TABLE I
LIST OF EXPANSIONS EVALUATED IN THIS PAPER.

shuffled version of $c_1(t)$ (shuffling is possible because time is discrete when using samples). A function f approximating $s(t)$ from $\tilde{\mathbf{x}}(t)$ is found as $f(\tilde{\mathbf{x}}) \stackrel{\text{def}}{=} Exp(1, \tilde{\mathbf{x}}(t)) \cdot \mathbf{w} \approx s(t)$, where the column vector $\mathbf{w}$ is computed with linear regression. Let $\tilde{\mathbf{x}}'(t)$ be the test data (with the same distribution as $\tilde{\mathbf{x}}(t)$), and $\tilde{y}'(t) \stackrel{\text{def}}{=} f(\tilde{\mathbf{x}}'(t))$ be the estimated labels for the test data. Then, overfitting is defined as:

$$Ov(Exp, \tilde{\mathbf{x}}(t), \tilde{\mathbf{x}}'(t)) \stackrel{\text{def}}{=} \sqrt{\langle(\tilde{y}')^2\rangle}, \qquad (5)$$

where $\langle\cdot\rangle$ indicates temporal averaging.

We also present a metric to quantify *outlier amplification*, where extreme/atypical outputs computed at different layers of the network are magnified, which is particularly notorious for test data (cf. Section III-E). Roughly speaking, the metric compares the scaled amplitude of extreme samples in the data before and after expansion. Let $\mathbf{X} \stackrel{\text{def}}{=} (\mathbf{x}(1), ..., \mathbf{x}(N))$ be the training samples, and *Exp* the expansion we want to evaluate. Let $\mathbf{X}' \stackrel{\text{def}}{=} \mathcal{W}(X)$ and $\mathbf{Y}' \stackrel{\text{def}}{=} \mathcal{W}(Exp((X))$ be whitened versions of $\mathbf{X}$ and the expanded samples, respectively. Let $\mathbf{S}$ be a set of vectors (samples or expanded samples), and $A(\mathbf{v}) \stackrel{\text{def}}{=} \|\mathbf{v}\|/\sqrt{D}$ be a scaled norm (amplitude) of a D-dimensional vector $\mathbf{v}$. A vector $\mathbf{v} \in \mathbf{S}$ is called an *outlier* if $A(\mathbf{v})$ is in the top 5% among all vectors $\mathbf{v}' \in \mathbf{S}$, and $\mathbf{O_S} \subset \mathbf{S}$ denotes the set of all outliers of $\mathbf{S}$. Outlier amplification is defined as

$$O_{amp}(Exp, \mathbf{X}) \stackrel{\text{def}}{=} \frac{\sum_{\mathbf{y}\in\mathbf{O_{Y'}}} A(\mathbf{y})}{\sum_{\mathbf{x}\in\mathbf{O_{X'}}} A(\mathbf{x})} \qquad (6)$$

and can be interpreted as the ratio between the scaled norm of the outliers after and before expansion, in both cases including a whitening step.

Due to whitening, this metric is invariant to linear transformations, and due to the scaling of the norm by $\sqrt{D}$, it provides some robustness to the expanded dimensionality. Various definitions of outliers exist, such as distance- or density-based [12], which might lead to alternative definitions of $A(\mathbf{v})$. For instance, $A(\mathbf{v})$ could be the distance to the nearest neighbour of $\mathbf{v}$. However this might be less efficient for a high number of samples and dimensions.

VI. Experimental Results

It has been observed that the inputs to SFA nodes in a network are typically much more noisy and distorted at the lower layers than at the top layers. Based on this observation, we considered three scenarios to evaluate the expansions. In each scenario, the input signals before expansion $\tilde{x}_i(t)$ are modelled as cosine functions of increasing frequency including additive Gaussian noise with variance ranging from a large variance $\sigma^2 = 0.75^2$ for the bottom layers, to $\sigma^2 = 0.5^2$ for the middle layers, and a small variance $\sigma^2 = 0.25^2$ for the top layers.

The performance for problems **P2** to **P7** and *Ov* was measured on test data generated with the same distribution as the training data. For **P1** and O_{amp}, only training data was used because the corresponding problem does not demand test data. In order to obtain realistic results, parameters were set similar to those used in previous experiments with real data and SFA networks; 10,000 simulated samples were used and the data dimensionality before expansion was set to 60.

Table II presents the performances of each expansion on each problem, metric, and on real data. Results were averaged over $\sigma^2 \in \{0.25^2, 0.5^2, 0.75^2\}$, except for O_{amp}, where only $\sigma^2 = 0.75^2$ was relevant (large noise is useful to introduce and measure outliers). The expansions $QExp/q_2$ and $QExp^{*0.4}$ gave poor performance on **P1**. This could be corrected by including the identity function in them. For problems **P2–P7**, quadratic expansions provided good performance in general, whereas cubic expansions performed excellent, however the latter are inappropriate in this setup because the expanded dimensionality is too high compared to the number of samples (consequently, they would be strongly penalized by *Ov*). In contrast, expansions *0.8Exp* and *SExp* are only good at **P2** and **P4**. For the metrics *Ov* and O_{amp}, the expanded dimension for the cubic expansions (39,711) was larger than the number of samples and would cause extreme overfitting and computational complications. Thus, *Ov* and O_{amp} were not evaluated for cubic expansions. *Ov* strongly depended on the expanded dimensionality and the particular expansion played a less important role. In the case of outlier amplification, the opposite was observed: The concrete shapes of the expansion functions are determinant, while the expanded dimensionality is less important.

To validate the heuristics on real data, the expansions were tested in a concrete hierarchical network to learn and estimate the horizontal position of a face in face photographs. Similar to [10, Appendix A], the resulting estimation was robust

	Identity	$QExp$	$CExp$	$QExp/q_2$	$CExp/q_3$	$QExp^{*0.4}$	$CExp^{*0.3}$	$0.8Exp$	$SExp$
P1	0.0000	0.0000	0.0000	0.3777	0.1122	0.2648	0.0122	0.0000	0.0000
P2	1.0000	0.7810	0.7810	0.7348	0.7815	0.7431	0.7498	0.7431	0.7810
P3	1.0000	0.7071	0.7032	0.6706	0.6421	0.7225	0.6659	1.0001	1.0001
P4	1.0000	0.7933	0.7934	0.7725	0.8259	0.7766	0.7853	0.7766	0.7933
P5	1.0000	0.8209	0.7755	0.8084	0.7118	0.8202	0.7251	1.0000	1.0000
P6	1.0000	0.7137	0.6962	0.6603	0.6134	0.6673	0.6099	1.0000	1.0000
P7	1.0000	0.7163	0.6599	0.6608	0.5518	0.6607	0.5574	1.0001	1.0000
Ov	0.9969	1.0959	$\gg 1$	1.0946	$\gg 1$	1.0938	$\gg 1$	0.9970	0.9977
O_{amp}	1.0000	1.0520	—	0.9631	—	0.9793	—	0.9854	1.0872
$\Delta(y_1)$ training (real data)	1.0460	0.1844	—	0.0588	—	0.0283	—	0.2829	0.5176
$\Delta(y_1)$ test (real data)	1.0701	275.3278	—	0.1555	—	0.1154	—	0.2855	12.7479
Estimation error [pixels]	17.96	27.54	—	5.85	—	6.18	—	6.16	16.44

TABLE II
PERFORMANCE OF EACH EXPANSION ON EACH PROBLEM, METRIC, AND ON REAL DATA. THE PERFORMANCE FOR PROBLEMS **P1**–**P7** AND THE ESTIMATION ERROR ARE REPORTED IN TERMS OF THE RMSE. FOR ALL TESTS, SMALLER VALUES ARE BETTER.

to the face's size, vertical position, expression, illumination, background, etc. It was verified that a good performance on problems **P1**–**P7** is related to slower extracted features for training data (see "$\Delta(y_1)$ training" in Table II). It was also verified that the metric O_{amp} is a good predictor of outlier amplification for test data (which occurred for $QExp$ and $SExp$). Whenever $O_{amp} \leq 1$, Ov showed correlation with overfitting when expressed as the ratio between delta values for test and training data.

VII. CONCLUSION

In this paper, a method for the heuristic evaluation of expansions used in non-linear hierarchical SFA was proposed. It consists of tests on seven different problems and two metrics. Expansions are crucial for the quality of the features extracted in terms of noise, generalization to new data, and accuracy when used in supervised learning setups.

It was verified experimentally with a concrete subproblem on face detection that a good performance on problems **P1**–**P7** is related to the richness of the spanned feature space and, thus, to the computational capabilities of SFA. The metric O_{amp} provides a simple way to predict (and avoid) the occurrence of the outlier amplification phenomenon. The metric Ov in conjunction with O_{amp} provides a rough way to predict high overfitting.

The main advantage of the method is that it allows for very fast evaluation of expansions because it does not require training of SFA or the use of real data. The method can be easily applied to efficiently tune parameterized expansions, such as $QExp/(a + ||\mathbf{x}||^b)$ for good choices of a and b.

The method has the disadvantage that it is less realistic for the lower layers of the hierarchical network, in which the extracted signals typically do not resemble noisy cosine functions, but the results are still consistent with real data.

One open problem is the computation of a single numerical score from all individual tests, which would ease the automated refinement of expansions. In future work, we are interested in using this method for the guided design of powerful expansions with good scores on all considered problems and metrics, but with a small expanded dimensionality. For instance, we would like to select a small subset of the functions of a normalized cubic expansion (e.g., of size of about 120) that provides a complex feature space and allows for efficient training.

The results on outlier amplification suggest the use of normalized expansions with functions that saturate or grow at most linearly for *large* inputs (e.g., in contrast to $QExp$) in order to achieve better performances.

REFERENCES

[1] I. K. Fodor, "A survey of dimension reduction techniques," *Communications of The ACM*, 2002.

[2] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals Eugen.*, vol. 7, pp. 179–188, 1936.

[3] L. Wiskott and T. Sejnowski, "Slow feature analysis: Unsupervised learning of invariances." *Neural Computation*, vol. 14, no. 4, pp. 715–770, 2002.

[4] P. Berkes and L. Wiskott, "Slow feature analysis yields a rich repertoire of complex cell properties," *J. Vis.*, vol. 5, no. 6, pp. 579–602, 2005.

[5] L. Wiskott, "Slow feature analysis: A theoretical analysis of optimal free responses," *Neural Computation*, vol. 15, no. 9, pp. 2147–2177, Sep. 2003.

[6] A. Bray and D. Martinez, "Kernel-based extraction of slow features: Complex cells learn disparity and translation invariance from natural images." in *Proc. Advances in neural information processing systems (NIPS'03)*, vol. 15. Cambridge, MA: MIT Press, 2003, pp. 253–260.

[7] R. Vollgraf and K. Obermayer, "Sparse optimization for second order kernel methods," in *Proc. IJCNN'06*, 2006, pp. 145 – 152.

[8] M. Franzius, N. Wilbert, and L. Wiskott, "Invariant object recognition with slow feature analysis," in *Proc. 18th Intl. Conf. on Artificial Neural Networks*, ser. LNCS, vol. 5163. Springer, Sep. 2008, pp. 961–970.

[9] A. Escalante and L. Wiskott, "Gender and age estimation from synthetic face images with hierarchical slow feature analysis," in *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, 2010, pp. 240–249.

[10] N. M. Mohamed and H. Mahdi, "A simple evaluation of face detection algorithms using unpublished static images," in *10th International Conference on Intelligent Systems Design and Applications, ISDA*, 2010, pp. 1–5.

[11] M. Franzius, H. Sprekeler, and L. Wiskott, "Slowness and sparseness lead to place, head-direction, and spatial-view cells," *PLoS Computational Biology*, vol. 3, no. 8, p. e166, Aug. 2007.

[12] S. Subramaniam, T. Palpanas, D. Papadopoulos, V. Kalogeraki, and D. Gunopulos, "Online outlier detection in sensor data using non-parametric models," in *Proc. 32nd international conference on Very large data bases*, ser. VLDB '06. VLDB Endowment, 2006, pp. 187–198.