Machine Learning (2020) 109:999–1037
https://doi.org/10.1007/s10994-019-05860-9

Improved graph-based SFA: information preservation complements the slowness principle

Alberto N. Escalante-B.[1] · **Laurenz Wiskott**[1]

Received: 1 February 2016 / Revised: 4 November 2019 / Accepted: 7 November 2019 /
Published online: 26 December 2019

Abstract
Slow feature analysis (SFA) is an unsupervised learning algorithm that extracts slowly varying features from a multi-dimensional time series. SFA has been extended to supervised learning (classification and regression) by an algorithm called graph-based SFA (GSFA). GSFA relies on a particular graph structure to extract features that preserve label similarities. Processing of high dimensional input data (e.g., images) is feasible via hierarchical GSFA (HGSFA), resulting in a multi-layer neural network. Although HGSFA has useful properties, in this work we identify a shortcoming, namely, that HGSFA networks prematurely discard quickly varying but useful features before they reach higher layers, resulting in suboptimal global slowness and an under-exploited feature space. To counteract this shortcoming, which we call unnecessary information loss, we propose an extension called hierarchical information-preserving GSFA (HiGSFA), where some features fulfill a slowness objective and other features fulfill an information preservation objective. The efficacy of the extension is verified in three experiments: (1) an unsupervised setup where the input data is the visual stimuli of a simulated rat, (2) the localization of faces in image patches, and (3) the estimation of human age from facial photographs of the MORPH-II database. Both HiGSFA and HGSFA can learn multiple labels and offer a rich feature space, feed-forward training, and linear complexity in the number of samples and dimensions. However, the proposed algorithm, HiGSFA, outperforms HGSFA in terms of feature slowness, estimation accuracy, and input reconstruction, giving rise to a promising hierarchical supervised-learning approach. Moreover, for age estimation, HiGSFA achieves a mean absolute error of 3.41 years, which is a competitive performance for this challenging problem.

Keywords Supervised dimensionality reduction · Similarity-based learning · Information preservation · Deep neural networks · Age estimation

Editor: Tijl De Bie.

✉ Alberto N. Escalante-B.
alberto.escalante@ini.rub.de

Laurenz Wiskott
laurenz.wiskott@ini.rub.de

[1] Institut für Neuroinformatik, Ruhr-Universität Bochum, Universitätsstraße 150, 44801 Bochum, Germany

1 Introduction

Dimensionality reduction (DR) has many applications in computer vision and machine learning, where it is frequently used, for example, as a pre-processing step for solving classification and regression problems on high-dimensional data. DR can be done in either a supervised or an unsupervised fashion.

An approach for *unsupervised* DR relies on the slowness principle, which requires the extraction of the slowest varying features (Hinton 1989). This principle forms the basis of the slow feature analysis (SFA) algorithm (Wiskott 1998; Wiskott and Sejnowski 2002), which extracts temporally stable features. Moreover, it has been shown that this principle might explain in part how the neurons in the brain self-organize to compute invariant representations.

When the final goal is supervised learning, *supervised* DR is more appropriate. Its objective is to extract a low-dimensional representation of the input samples that contains the predictive information about the labels (e.g., Rish et al. 2008). An advantage is that dimensions irrelevant to the supervised problem can be discarded, resulting in a more compact representation and more accurate estimations. Although SFA is unsupervised it can be used for supervised DR, because the order of the samples can be regarded as a weak form of supervised information (consecutive samples are likely to have similar labels). However, an extension to SFA called graph-based SFA (GSFA, Escalante-B. and Wiskott 2012, 2013) is preferable for supervised DR, because it uses the labels explicitly, yielding more accurate label estimations. The input data of GSFA is a graph structure where the vertices are the samples and label similarities can be encoded by the weights of the edges connecting pairs of samples. Technical applications of GSFA include the estimation of age and gender from synthetic face images (Escalante-B. and Wiskott 2010), traffic sign classification (Escalante-B. and Wiskott 2016), and face detection (Mohamed and Mahdi 2010).

A frequent problem of DR algorithms is their high computational cost when the input data is high-dimensional. For example, when GSFA is applied to the data directly, what we call direct GSFA, it has *cubic* complexity w.r.t. the number of dimensions. However, great efficiency can be achieved by resorting to hierarchical GSFA (HGSFA), where the extraction of slow features is done progressively: in a first layer, slow features are extracted from lower dimensional splits of the input data by independent instances of GSFA, called GSFA nodes, reducing the dimensionality of each split. This procedure can be repeated in the next layers until the number of dimensions is small enough. Additional advantages of HGSFA over direct GSFA include lower memory complexity, a more complex global nonlinear feature space, and less overfitting.

Although SFA is based on the promising slowness principle, HSFA has not achieved state-of-the-art performance on practical applications. The question of why this is the case has puzzled researchers (Goodfellow et al. 2016). This work contributes to the solution of this question. Through an assessment of HGSFA and HSFA networks, we show that HGSFA suffers from an important shortcoming: The GSFA nodes of the network may prematurely discard features that are not slow at a local scale but that would have been useful to improve global slowness (i.e., the slowness of the final output features) if combined in subsequent nodes with features computed by other nodes. This drawback, which we call unnecessary information loss, leads to an under-exploited feature space, i.e., the global feature space contains slower features than those actually extracted by HGSFA.

As a countermeasure to unnecessary information loss, we propose to complement slowness with information preservation (i.e., maximization of mutual information between the input data and the output features). For simplicity and efficiency, we implement this idea

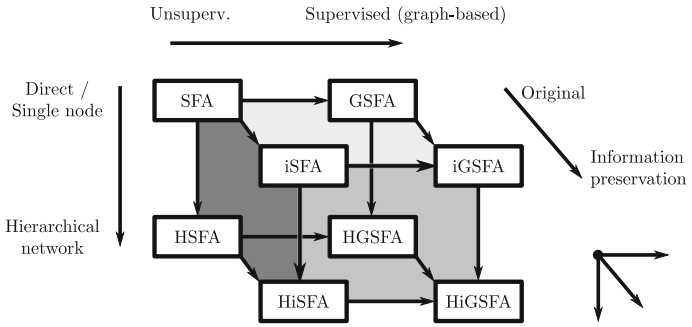

Fig. 1 Illustration of three basic extensions to SFA (graph-based, hierarchical, information-preserving), each of them is represented by a different direction. The combined use of these extensions results in 8 different variants of SFA. This article proposes information preservation, which is used in iSFA, iGSFA, HiSFA, and HiGSFA, the last one being the most promising variant of SFA

as a minimization of the mean squared reconstruction error with PCA. The resulting network that considers both optimization goals is called *hierarchical information-preserving GSFA* (HiGSFA), and the algorithm used in each node of the network is called *information-preserving GSFA* (iGSFA). The feature vector computed by iGSFA comprises two parts: (1) a slow part, which is a linear mixture of the (nonlinear) features computed using GSFA, and (2) an input-reconstructive part computed using PCA.

The iGSFA algorithm, which is the main building block of HiGSFA, reduces the redundancy between the slow and reconstructive parts by making both parts decorrelated: The reconstructive part does not directly approximate the input data but only a version of it where the slow part has been projected out, called residual data. Moreover, iGSFA also ensures that the scale of the slow components is compatible with the scale of the reconstructive components. This enables meaningful processing by PCA in subsequent layers (PCA is sensitive to input feature magnitudes). Different versions of SFA with and without information preservation are shown in Fig. 1.

A fundamental motivation for investigating H(iG)SFA is that, in contrast to gradient-based training of neural networks, it has a stronger biological feasibility because each iGSFA node adapts only to its local input. Thus, the method allows massive parallelization and scalability. On the other hand, it does not minimize a global loss function explicitly, as gradient-based method do, but our experimental results prove that one can achieve remarkable accuracy with the resulting networks.

The experiments show the versatility and generality of the proposed extensions in three setups: (1) Unsupervised learning, where the input data is the view of a simulated rat moving inside a box. This is solved with HiSFA, a special case of HiGSFA. (2) Supervised learning, where four pose parameters of faces depicted in image patches must be estimated. This problem is closely connected to face detection and tracking. (3) A supervised problem requiring the estimation of age from facial photographs. All three experiments show the advantages of using information preservation: (a) slower features, (b) better generalization to unseen data, (c) much better input reconstruction (see Fig. 2), and (d) improved classification/regression accuracy (experiments 2 and 3). Furthermore, the computational and memory requirements of HiGSFA are moderate, having the same asymptotic order as those of HGSFA.

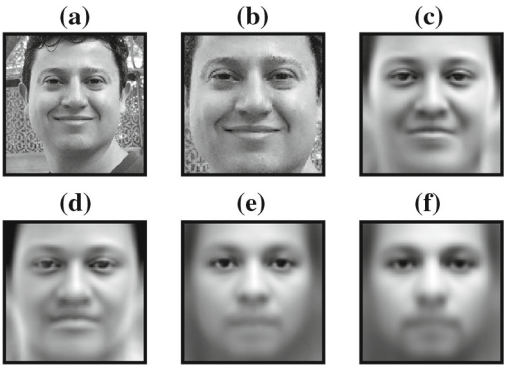

Fig. 2 **a** An image from a private database after pose normalization. **b** The same image fully pre-processed (i.e., after pose normalization and face sampling, 96×96 pixels). Linear reconstructions on 75 features extracted with either **c** PCA, **d** HiGSFA or **e** HGSFA. **f** Average over all pre-processed images of the MORPH-II database. Notice that the reconstruction using HiGSFA features is most similar to that of PCA, whereas the reconstruction using HGSFA features is most similar to the average image.

2 Previous work

HiGSFA is the main extension to SFA proposed in this work and belongs to supervised DR. Other algorithms for supervised DR include Fisher discriminant analysis (FDA, Fisher 1936), local FDA (LFDA, Sugiyama 2006), pairwise constraints-guided feature projection (PCGFP, Tang and Zhong 2007), semi-supervised dimensionality reduction (SSDR, Zhang et al. 2007), and semi-supervised LFDA (SELF, Sugiyama et al. 2010).

Previous extensions to SFA include extended SFA (xSFA, Sprekeler et al. 2014), generalized SFA (genSFA, Sprekeler 2011) and graph-based SFA (GSFA, Escalante-B. and Wiskott 2010, 2012, 2013). With some adaptations, SFA has been shown to be successful for classification (e.g., Berkes 2005b; Koch et al. 2010; Franzius et al. 2011; Kuhnl et al. 2011; Zhang and Tao 2012), and regression (e.g., Franzius et al. 2011). HiGSFA extends hierarchical GSFA (HGSFA) by adding information preservation.

2.1 Slow Feature Analysis (SFA)

Slow features can be computed using a few methods, such as online learning rules (e.g., Földiák 1991; Mitchison 1991), slow feature analysis (SFA, Wiskott 1998; Wiskott and Sejnowski 2002), which is a closed-form algorithm specific for this task and has biologically feasible variants (Sprekeler et al. 2007), and an incremental-learning version (inc-SFA, Kompella et al. 2012).

The SFA optimization problem is defined as follows (Wiskott 1998; Wiskott and Sejnowski 2002; Berkes and Wiskott 2005). Given an I-dimensional input signal $\mathbf{x}(t) = (x_1(t), \ldots, x_I(t))^T$, where $t \in \mathbb{R}$, find J instantaneous scalar functions $g_j : \mathbb{R}^I \to \mathbb{R}$, for $1 \leq j \leq J$, within a function space $\mathcal{F}$ such that the output signal components $y_j(t) \stackrel{\text{def}}{=} g_j(\mathbf{x}(t))$ minimize the objective function $\Delta(y_j) \stackrel{\text{def}}{=} \langle \dot{y}_j(t)^2 \rangle_t$ (delta value), under the following constraints: (1) $\langle y_j(t) \rangle_t = 0$ (zero mean), (2) $\langle y_j(t)^2 \rangle_t = 1$ (unit variance), and (3) $\langle y_j(t) y_{j'}(t) \rangle_t = 0, \forall j' < j$ (decorrelation and order).

The delta value $\Delta(y_j)$ is defined as the time average ($\langle \cdot \rangle_t$) of the squared derivative of y_j and is therefore a measure of the slowness (or rather fastness) of the signal, whereas the

constraints require that the output signals are normalized, not constant, and represent different aspects of the input signal. SFA computes an optimal solution to the problem above within a linear feature space (possibly after the data has been expanded nonlinearly). Typically, discrete time is used, and the derivative is approximated as $\dot{y}_j(t) \stackrel{\text{def}}{=} y_j(t+1) - y_j(t)$.

2.2 Hierarchical SFA (HSFA) and terminology

Hierarchical SFA (HSFA) was already introduced in the paper that first introduces the SFA algorithm (Wiskott 1998), where it is employed as a model of the visual system for learning invariant representations.

Various publications have followed this biological interpretation. Franzius et al. (2007) have used HSFA to learn invariant features from the simulated view of a rat walking inside a box. In conjunction with a sparseness post-processing step, the extracted features are similar to the responses of place cells in the hippocampus.

Other works have exploited the computational efficiency of HSFA compared to direct SFA. Franzius et al. (2011) have used HSFA for object recognition from images and to estimate pose parameters of single objects moving and rotating over a homogeneous background. Escalante-B. and Wiskott (2013) have used an 11-layer HGSFA network to accurately find the horizontal position of faces in photographs.

In general, HSFA networks (composed of several SFA nodes) are directed and acyclic. They are usually structured in multiple layers of nodes, following a grid structure. Most HSFA networks in the literature have a similar architecture. Typical differences are how the data is split into smaller blocks and the particular pre-processing done by before the SFA nodes themselves.

For simplicity, we refer to the input data as layer 0. Important parameters that define the structure of the network include: (a) The *output dimensionality* of the nodes. (b) The *fan-in* of the nodes, which is the number of nodes (or data elements) in a previous layer that feed into them. (c) The *receptive field* of the nodes, which refers to all the elements of the input data that (directly or indirectly) provide input to a particular node. (d) The *stride* of a layer, which tells how far apart the inputs to adjacent nodes in a layer are. If the stride is smaller than the fan-in along the same direction, then at least one node in the previous layer will feed two or more nodes in the current layer. This is called *receptive field overlap*.

2.3 Graph-based SFA (GSFA)

Graph-based SFA (Escalante-B. and Wiskott 2013) is an extension to SFA designed for supervised learning. GSFA extracts a compact set of features that is usually post-processed by a typical supervised algorithm to generate the final label or class estimate.

In GSFA the training data takes the form of a *training graph* $G = (\mathbf{V}, \mathbf{E})$ (illustrated in Fig. 3a), which is a structure composed of a set $\mathbf{V}$ of vertices $\mathbf{x}(n)$, each vertex being a sample (i.e. an I-dimensional vector), and a set $\mathbf{E}$ of edges $(\mathbf{x}(n), \mathbf{x}(n'))$, which are pairs of samples, with $1 \leq n, n' \leq N$. The index n (or n') replaces the time variable t of SFA. The edges are undirected and have symmetric weights $\gamma_{n,n'} = \gamma_{n',n}$, which indicate the label similarity between the connected vertices; also, each vertex $\mathbf{x}(n)$ has an associated weight $v_n > 0$, which can be used to reflect its frequency. This representation includes the standard time series of SFA as a special case (Fig. 3b).

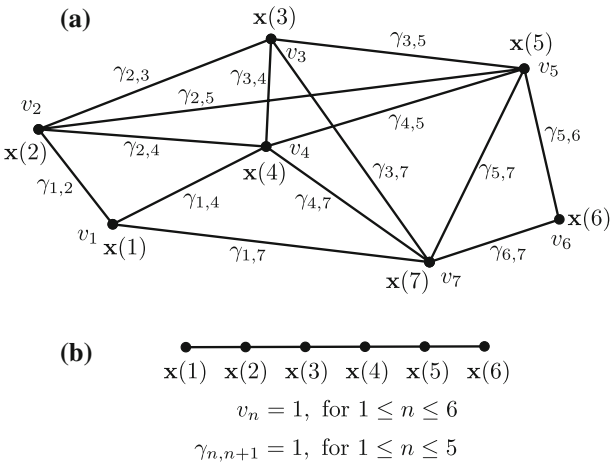

Fig. 3 Two examples of training graphs: **a** A training graph with $N = 7$ vertices. **b** A linear graph suitable for GSFA that learns the same features as SFA on the time series $\mathbf{x}(1), \ldots, \mathbf{x}(6)$. (Figure from Escalante-B. and Wiskott 2013)

The GSFA optimization problem (Escalante-B. and Wiskott 2013) can be stated as follows. For $1 \le j \le J$, find output features $y_j(n) = g_j(\mathbf{x}(n))$ with $g_j \in \mathcal{F}$, where $1 \le n \le N$ and $\mathcal{F}$ is the function space, such that the objective function (**weighted delta value**)

$$\Delta_j \stackrel{\text{def}}{=} \frac{1}{R} \sum_{n,n'} \gamma_{n,n'} (y_j(n') - y_j(n))^2 \text{ is minimal} \tag{1}$$

under the constraints

$$\frac{1}{Q} \sum_n v_n y_j(n) = 0 \text{ (weighted zero mean)}, \tag{2}$$

$$\frac{1}{Q} \sum_n v_n (y_j(n))^2 = 1 \text{ (weighted unit variance)}, \text{ and} \tag{3}$$

$$\frac{1}{Q} \sum_n v_n y_j(n) y_{j'}(n) = 0 \text{ for } j' < j \text{ (weighted decorrelation)} \tag{4}$$

with $R \stackrel{\text{def}}{=} \sum_{n,n'} \gamma_{n,n'}$ and $Q \stackrel{\text{def}}{=} \sum_n v_n$.

The factors $1/R$ and $1/Q$ provide invariance to the scale of the edge and vertex weights. Typically, a linear function space is used, but the input samples are preprocessed by a nonlinear expansion function.

One should choose the edge weights of the training graph properly, because they control what kind of features are extracted by GSFA. In general, to obtain features useful for classification, one should favor connections between samples from the same class (stronger edge weighs for same-class samples). To obtain features useful for regression, one should favor connections between samples with similar labels. In Sects. 5.2.3 and 5.3.2, we combine various training graphs to learn feature representations that allow the solution of various classification and regression problems simultaneously (multi-label learning).

The rest of the article is organized as follows. In the next section we analyze the advantages and limitations of hierarchical networks for slow feature extraction. The shortcomings we

expose motivate the introduction of the iSFA (and iGSFA) algorithms in Sect. 4. In Sect. 5 these algorithms, or more precisely, hierarchical neural networks built with them, are evaluated experimentally using three different problems of supervised and unsupervised DR. We conclude with a discussion section.

3 Assessment of hierarchical processing by HSFA and HGSFA networks

In this section we analyze HSFA networks in terms of their advantages and limitations. This analysis is crucial, since it justifies the extensions with information preservation (HiSFA and HiGSFA) proposed in Sect. 4.

3.1 Advantages of HSFA and HGSFA networks

The central advantages of hierarchical processing in H(G)SFA—compared to direct (G)SFA—have been mentioned in the introduction and Sect. 2.2: (1) It reduces overfitting and can be seen as a regularization method, (2) the nonlinearity of the layers of the neural network accumulates in a compositional manner, so that even when using simple expansions the network as a whole may describe a highly nonlinear feature space, and (3) better computational efficiency than (G)SFA.

Some remarks about these advantages are pertinent: Advantage (1) is explained by the fact that the input dimensionality to the nodes of the hierarchical network is much smaller than the original input dimensionality, whereas the number of samples remains unchanged. Thus, individual nodes are less susceptible to overfitting. Consequently, the gap in generalization performance between HSFA and direct SFA is larger when polynomial expansions are introduced (compared to their linear versions), because the dimensionality of the expanded data is much larger in direct SFA and this translates into stronger overfitting. On the other hand HSFA is not guaranteed to find the optimal (possibly overfitting) solution within the available function space whereas SFA is.

Advantage (2) is valuable in practice, because most real-life problems are nonlinear. A complex feature space may be necessary to extract the slowest hidden parameters and solve the supervised problem with good accuracy.

Advantage (3) is addressed more precisely in the following paragraphs by first recalling the computational complexity of SFA and GSFA. This complexity is then compared with the complexity of a particular HSFA network ("Appendix A"). We focus on the training complexity rather than on the complexity of feature extraction during testing, because the former can be considerable in practice, whereas the latter is relatively lightweight in both HSFA and direct SFA.

Following standard notation of algorithm complexity, computational (time) complexity is denoted by T (e.g., T_{SFA}). Training linear SFA has a time (computational) complexity

$$T_{\mathrm{SFA}}(N, I) = \mathcal{O}(NI^2 + I^3)\,, \tag{5}$$

where N is the number of samples and I is the input dimensionality (possibly after a nonlinear expansion). The same complexity holds for GSFA if one uses an efficient training graph (e.g., the serial and clustered graphs, see Sects. 5.2.3 and 5.3.2), otherwise (for arbitrary graphs) it can be as large as

$$T_{\mathrm{GSFA}}(N, I) = \mathcal{O}(N^2 I^2 + I^3)\,. \tag{6}$$

For large I (i.e., high-dimensional data) direct SFA and direct GSFA are therefore inefficient.[1] In contrast, HSFA and HGSFA can be much more efficient. The exact complexity of HSFA and HGSFA depends on the structure and parameters of the hierarchical network. "Appendix A" proves that the training complexity is linear in I and N for certain networks.

3.2 Limitations of HSFA networks

Although HSFA and HGSFA networks show remarkable advantages, they also have some shortcomings in their current form. The analysis in this section focuses on HSFA, but it also applies to other networks in which the nodes have only one criterion for DR, namely slowness maximization, such as HGSFA. Besides the slowness maximization objective, no other restriction is imposed on the nodes; they might be linear or nonlinear, include additive noise, clipping, various passes of SFA, etc.

It is shown here that relying only on slowness to determine which aspects of the data are preserved results in two shortcomings: unnecessary information loss and poor input reconstruction, explained below.

(1) *Unnecessary information loss* This shortcoming occurs when the nodes of the network discard dimensions of the data that are not significantly slow locally (i.e., at the node level), but which would have been useful for slowness optimization by other nodes in subsequent layers if they had been preserved (and combined with other dimensions).

The following minimal theoretical experiment shows that dimensions crucial to extract global slow features are not necessarily slow locally. Consider four zero-mean, unit-variance signals: $s_1(t)$, $s_2(t)$, $s_3(t)$ and $n(t)$ that can only take binary values, either -1 or $+1$ (n stands for noise here, and t is time, or more precisely, sample number). Assume these signals are ordered by slowness ($\Delta_{s_1} < \Delta_{s_2} < \Delta_{s_3} < \Delta_n = 2.0$) and they are statistically independent. The value 2.0 is precisely the expected Δ value of a random unit-variance i.i.d. noise feature. The same holds for GSFA if the graph is consistent and has no self-loops (Escalante-B. and Wiskott 2016).

Let the 4-dimensional input to the network be $(x_1, x_2, x_3, x_4) \stackrel{\text{def}}{=} (s_2, s_1 n, s_3, n)$ and assume the number of samples is large enough. The direct application of quadratic SFA (QSFA, i.e., a quadratic expansion followed by linear SFA) to this data would allow us to extract the slowest possible feature, namely, $x_2 x_4 = (s_1 n) n = s_1$ (or equivalently $-x_2 x_4$). However, let us assume that a 2-layer quadratic HSFA (QHSFA) network with 3 nodes is used, where the output of the network is: $\text{QSFA}\big(\text{QSFA}(s_2, s_1 n), \text{QSFA}(s_3, n)\big)$. Each quadratic QSFA node reduces the number of dimensions from 2 to 1. Since $\Delta_{s_2} < \Delta_{s_1 n} = 2.0$, the first bottom node computes $\text{QSFA}(s_2, s_1 n) = s_2$, and since $\Delta_{s_3} < \Delta_n = 2.0$, the second bottom node computes $\text{QSFA}(s_3, n) = s_3$. The top node would then extract $\text{QSFA}(s_2, s_3) = s_2$. Therefore, the network would miss the slowest feature, s_1, even though it actually belongs to the feature space spanned by the network.

The problem can be expressed in information theoretic terms:

$$I(s_1 n, s_1) = 0\,, \text{ and} \tag{7}$$

$$I(n, s_1) = 0\,, \text{ but} \tag{8}$$

$$I((s_1 n, n), s_1) = H(s_1) > 0\,, \tag{9}$$

[1] The problem is still feasible when N is small by applying singular value decomposition methods. However, a small number of samples $N < I$ usually results in pronounced overfitting.

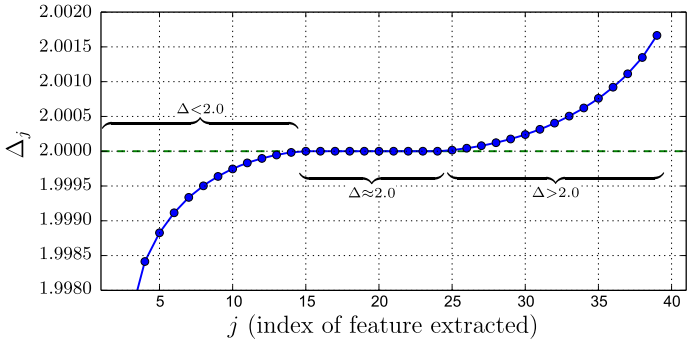

Fig. 4 Δ values of the first 40 slow features of an HGSFA network trained for age estimation and averaged over all the nodes of the first layer ($\Delta_1 = 1.859$, $\Delta_2 = 1.981$, and $\Delta_3 = 1.995$, not shown). The training graph employed is a serial graph with 32 groups (see Sect. 5.3.2). Most Δ values are close to 2.0, indicating that at this early stage, where the nodes have small 6×6-pixel receptive fields, the slowest extracted features are not substantially slow. Some features have $\Delta > 2.0$ and are thus slightly faster than noise.

where H is entropy, and I denotes mutual information.[2] Equations (7)–(9) show that it is impossible to *locally* rule out that a feature contains information that might yield a slow feature (in this case s_1), unless one *globally* observes the whole data available to the network. The problem above could be solved if the network could preserve n and $s_1 n$ by resorting to another criteria besides slowness. For example, if the signals n and $s_1 n$ were $10n$ and $10s_1 n$ instead, one could distinguish and preserve them based on their larger variance.

Unnecessary information loss is not only a theoretical concern, it can affect applications in practice. Figure 4 shows the Δ values of the slowest features extracted by the first layer of an HGSFA network trained for age estimation from human face images. Most Δ values are approximately 2.0, and only a few of them are significanlty smaller than 2.0. The value 2.0 is crucial; a feature with $\Delta = 2.0$ can be a transformation of relevant inputs, a transformation of irrelevant noise inherent to the inputs, or something between these cases. In fact, if two or more feasible features have the same Δ value, GSFA outputs an arbitrary rotation of them, even though one might prefer features that are transformations of the relevant input rather than noise components. Due to DR only a fraction of the features may be preserved and the rest is discarded even though some of them might still contain useful information.

One might try to preserve many features to reduce information loss. However, this might be impractical, because it would increase the computational cost and contradict the goal of dimensionality reduction.

(2) *Poor input reconstruction* Input reconstruction is the task of generating an (artificial) input having a given feature representation (or an approximation of it). Wilbert (2012) has studied this task for HSFA.

In the image domain, reconstruction can be useful to extrapolate images by computing distortions to an input image that reflect modifications introduced to the output features. For example, assume SFA was trained to extract body mass index (BMI) from facial images. Reconstruction would allow us to visualize how a particular person would look after losing or gaining a few kilograms.

[2] $H(X)$ is the average amount of information given by instances of a random variable X. $I(X, Y)$ is the average amount of information that a random variable X gives about another random variable Y (or vice-versa). In other words, it denotes how much information is duplicated in X and Y on average. If $I(X, Y) = 0$, the variables are independent.

Experiments have shown that input reconstruction from top-level features extracted by HSFA is a challenging task (Wilbert 2012). We have confirmed this observation through previous experiments using various nonlinear methods for input reconstruction, including local and global methods.

We show here that poor input reconstruction may not be caused by the weakness of the reconstruction algorithms employed but rather by insufficient reconstructive information in the slow features: The extracted features ideally depend only on the hidden slow parameters and are invariant to any other factor. In the BMI estimation example, the extracted features would be strongly related to the BMI (and nonlinear transformations of it). Thus, the features would be mostly invariant to other factors, such as the identity of the person, his or her facial expression, the background, etc. Therefore, in theory only BMI information would be available for reconstruction.

In practice, residual information about the input data can still be found in the extracted features. However, one cannot rely on this information because it is partial (making reconstructions not unique) and highly nonlinear (making it difficult to untangle). Even the features extracted by linear SFA typically result in inaccurate reconstructions. Since HSFA has many layers of SFA nodes, the problem is potentially aggravated.

The connection between the problems of unnecessary information loss and poor input reconstruction is evident if one distinguishes between two types of information: (a) information about the full input data and (b) information about the global slow parameters. Losing (a) results in poor input reconstruction, whereas losing (b) results in unnecessary information loss. Of course, (a) contains (b). Therefore, both problems originate from losing different but related types of information.

4 Hierarchical information-preserving GSFA (HiGSFA)

In this section, we formally propose HiGSFA, an extension to HGSFA that counteracts the problems of unnecessary information loss and poor input reconstruction described above by extracting reconstructive features in addition to slow features. HiGSFA is a hierarchical implementation of information-preserving GSFA (iGSFA). To simplify the presentation we first focus on information-preserving SFA (iSFA) and explain later how to easily extend iSFA to iGSFA and HiGSFA. We write iSFA with lowercase ‘i’ to distinguish it from independent SFA (ISFA, Blaschke et al. 2007).

iSFA combines two learning principles: the slowness principle and information preservation. Information preservation requires the maximization of the mutual information between the output features and the input data. However, for finite, discrete and typically unique data samples, it is difficult to measure and maximize mutual information unless one assumes a specific probability model. Therefore, we implement information preservation more practically as the minimization of a reconstruction error. A closely related concept is the explained variance, but this term is avoided here because it is typically restricted to linear transformations.

4.1 Algorithm overview (iSFA)

HiSFA improves feature extraction of HSFA networks at the node and global level. The SFA nodes of an HSFA network are replaced by iSFA nodes, leaving the network structure

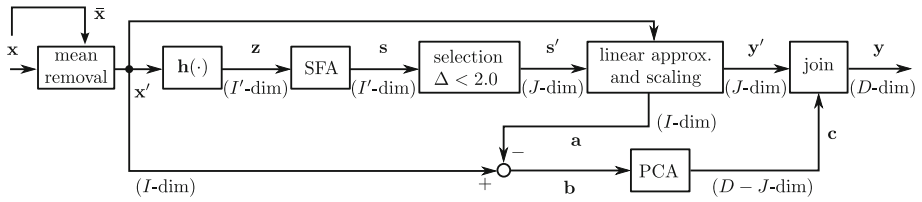

Fig. 5 Block diagram of the iSFA node showing the components used for training and feature extraction. Signal dimensionalities are given in parenthesis. The components and signals are explained in the text.

unchanged (although the network structure could be tuned to achieve better accuracy, if desired).

The feature vectors computed by iSFA consist of two parts: (1) a slow part derived from SFA features, and (2) a reconstructive part derived from principal components (PCs). Generally speaking, the slow part captures the slow aspects of the data and is basically composed of standard SFA features, except for an additional linear mixing step to be explained in Sects. 4.2 and 4.4. The reconstructive part ignores the slowness criterion and describes the input linearly as closely as possible (disregarding the part already described by the slow part). In Sect. 5 we show that, although the reconstructive features are not particularly slow, they indeed contribute to global slowness maximization.

4.2 Algorithm description (training phase of iSFA)

Figure 5 shows the components of iSFA, and Algorithm 1 summarizes the algorithm compactly. The iSFA algorithm (training phase) is defined as follows. Let $\mathbf{X} \overset{\text{def}}{=} (\mathbf{x}_1, \ldots, \mathbf{x}_N)$ be the I-dimensional training data, D the output dimensionality, $\mathbf{h}(\cdot)$ the nonlinear expansion function, and $\Delta_T \approx 2.0$ (a Δ-threshold, in practice slightly smaller than 2.0). First, the average sample $\bar{\mathbf{x}} \overset{\text{def}}{=} \frac{1}{N}\sum_n \mathbf{x}_n$ is removed from the N training samples resulting in the centered data $\mathbf{X}' \overset{\text{def}}{=} \{\mathbf{x}'_n\}$, with $\mathbf{x}'_n \overset{\text{def}}{=} \mathbf{x}_n - \bar{\mathbf{x}}$, for $1 \le n \le N$. Then, $\mathbf{X}'$ is expanded via $\mathbf{h}(\cdot)$, resulting in vectors $\mathbf{z}_n = \mathbf{h}(\mathbf{x}'_n)$ of dimensionality I'. Afterwards, linear SFA is trained with the expanded data $\mathbf{Z} \overset{\text{def}}{=} \{\mathbf{z}_n\}$, resulting in a weight matrix $\mathbf{W}_{\text{SFA}}$ and an average expanded vector $\bar{\mathbf{z}}$. The slow features extracted from $\mathbf{Z}$ are $\mathbf{s}_n = \mathbf{W}_{\text{SFA}}^T(\mathbf{z}_n - \bar{\mathbf{z}})$. The first J components of $\mathbf{s}_n$ with $\Delta < \Delta_T$ and $J \le \min(I', D)$ are denoted $\mathbf{s}'_n$. The remaining components of $\mathbf{s}_n$ are discarded.

$\mathbf{S}' \overset{\text{def}}{=} \{\mathbf{s}'_n\}$ has J features, each of them having zero mean and unit variance. The next steps modify the amplitude of $\mathbf{S}'$: The centered data $\mathbf{X}'$ is approximated from $\mathbf{S}'$ linearly by using ordinary least squares regression to compute a matrix $\mathbf{M}$ and a vector $\mathbf{d}$, such that

$$\mathbf{A} \overset{\text{def}}{=} \mathbf{M}\mathbf{S}' + \mathbf{d}\mathbf{1}^T \approx \mathbf{X}', \tag{10}$$

where $\mathbf{A}$ is the approximation of the centered data given by the slow part (i.e., $\mathbf{x}'_n \approx \mathbf{a}_n \overset{\text{def}}{=} \mathbf{M}\mathbf{s}'_n + \mathbf{d}$) and $\mathbf{1}$ is a vector of 1s of length N. Since $\mathbf{X}'$ and $\mathbf{S}'$ are centered, $\mathbf{d}$ could be discarded because $\mathbf{d} = \mathbf{0}$. However, when GSFA is used the slow features have only *weighted* zero mean, and $\mathbf{d}$ might improve the approximation of $\mathbf{X}'$. Afterwards, the QR decomposition of $\mathbf{M}$

$$\mathbf{M} = \mathbf{Q}\mathbf{R} \tag{11}$$

Algorithm 1 Training phase of iSFA

Require: $D > 0$: output dimensionality
1: **procedure** TRAIN($\mathbf{X}$) % $\mathbf{X} = (\mathbf{x_1}, \ldots, \mathbf{x_N})$: training samples
2: $\forall n : \mathbf{x'_n} \leftarrow \mathbf{x_n} - \bar{\mathbf{x}}$ % $\bar{\mathbf{x}}$: average sample
3: $\forall n : \mathbf{z_n} \leftarrow \mathbf{h}(\mathbf{x'_n})$ % $\mathbf{Z} = (\mathbf{z_1}, \ldots, \mathbf{z_N})$: expanded samples
4: $\mathbf{W}_{\text{SFA}}, \bar{\mathbf{z}} \leftarrow$ SFA.TRAIN($\mathbf{Z}$, OUTPUT_DIM $= \min(I', D)$)
5: $\forall n : \mathbf{s_n} \leftarrow \mathbf{W}_{\text{SFA}}{}^T(\mathbf{z_n} - \bar{\mathbf{z}})$
6: $\forall n : \mathbf{s'_n} \leftarrow (s_{n1}, \ldots, s_{nJ})^T$ % Preserve the first J features with $\Delta < \Delta_T$
7: $\forall n : \mathbf{a_n} \leftarrow \mathbf{M}\mathbf{s'_n} + \mathbf{d}$ % For $\mathbf{M}$ and $\mathbf{d}$, such that $\mathbf{M}\mathbf{S}' + \mathbf{d} \approx \mathbf{X}'$
8: $\forall n : \mathbf{y'_n} \leftarrow \mathbf{R}\mathbf{s'_n}$ % For $\mathbf{QR} = \mathbf{M}$, the QR decomposition of $\mathbf{M}$
9: $\forall n : \mathbf{b_n} \leftarrow \mathbf{x'_n} - \mathbf{a_n}$ % $\mathbf{B} = (\mathbf{b_1}, \ldots, \mathbf{b_N})$
10: $\mathbf{W}_{\text{PCA}} \leftarrow$ PCA.TRAIN($\mathbf{B}$, OUTPUT_DIM $= D - J$)
11: $\forall n : \mathbf{c_n} \leftarrow \mathbf{W}_{\text{PCA}}^T\mathbf{b_n}$ % Only $D - J$ PCs are preserved
12: $\forall n : \mathbf{y_n} = \mathbf{y'_n}|\mathbf{c_n}$ % Concatenation of slow and reconstructive parts
13: **return** $\mathbf{Y} = (\mathbf{y_1}, \ldots, \mathbf{y_N})$, $\bar{\mathbf{x}}$, $\mathbf{W}_{\text{SFA}}$, $\bar{\mathbf{z}}$, $\mathbf{W}_{\text{PCA}}$, J, $\mathbf{Q}$, $\mathbf{R}$, $\mathbf{d}$
14: **end procedure**

is computed, where $\mathbf{Q}$ is orthonormal and $\mathbf{R}$ is upper triangular. Then, the (amplitude-corrected) slow feature part is computed as

$$\mathbf{y}'_n = \mathbf{R}\mathbf{s}'_n \,. \tag{12}$$

Section 4.4 justifies the mixing and scaling (12) of the slow features $\mathbf{s}'_n$.

To obtain the reconstructive part, residual data $\mathbf{b}_n \stackrel{\text{def}}{=} \mathbf{x}'_n - \mathbf{a}_n$ is computed, i.e., the data that remains after the data linearly reconstructed from $\mathbf{y}'_n$ (or $\mathbf{s}'_n$) is removed from the centered data. Afterwards, PCA is trained with $\{\mathbf{b}_n\}$, resulting in a weight matrix $\mathbf{W}_{\text{PCA}}$. There is no bias term, because $\mathbf{b}_n$ is centered. The reconstructive part $\mathbf{c}_n$ is then defined as the first $D - J$ principal components of $\mathbf{b}_n$ and computed accordingly.

Thereafter, the slow part $\mathbf{y}'_\mathbf{n}$ (J features) and the reconstructive part $\mathbf{c_n}$ ($D - J$ features) are concatenated, resulting in the D-dimensional output features $\mathbf{y_n} \stackrel{\text{def}}{=} \mathbf{y}'_\mathbf{n}|\mathbf{c_n}$, where | is vector concatenation.

Finally, the algorithm returns $\mathbf{Y} = (\mathbf{y_1}, \ldots, \mathbf{y_N})$, $\bar{\mathbf{x}}$, $\mathbf{W}_{\text{SFA}}$, $\bar{\mathbf{z}}$, $\mathbf{W}_{\text{PCA}}$, J, $\mathbf{Q}$, $\mathbf{R}$, and $\mathbf{d}$. The output features $\mathbf{Y}$ are usually computed only during feature extraction. Still, we keep them here to simplify the understanding of the signals involved.

We would like to remark that the algorithm proposed above takes care of the following important considerations: (a) From the total output dimensionality D, it decides how many features the slow and reconstructive part should contain. (b) It minimizes the redundancy between the slow and the reconstructive part, allowing the output features to be more compact and have higher information content. (c) It corrects the amplitudes of the slow features (SFA features have unit variance) to make them compatible with the PCs and allow their processing by PCA in subsequent nodes (PCA is a rotation and projection. Thus, it preserves the amplitude of the original data in the retained directions).

4.3 Feature extraction by iSFA

The feature extraction algorithm (described in Algorithm 2) is similar to the training algorithm, except that the parameters $\bar{\mathbf{x}}$, $\mathbf{W}_{\text{SFA}}$, $\bar{\mathbf{z}}$, $\mathbf{W}_{\text{PCA}}$, J, $\mathbf{Q}$, $\mathbf{R}$, and $\mathbf{d}$ have already been learned from the training data. Algorithm 2 processes a single input sample, but it can be easily and efficiently adapted to process multiple input samples by taking advantage of matrix operations.

Algorithm 2 Feature extraction with iSFA

Require: $D, \bar{\mathbf{x}}, \mathbf{W}_{\text{SFA}}, \bar{\mathbf{z}}, \mathbf{W}_{\text{PCA}}, J, \mathbf{Q}, \mathbf{R}, \mathbf{d}$
procedure EXTRACT($\mathbf{x}$) % $\mathbf{x}$: a new sample
  $\mathbf{x}' \leftarrow \mathbf{x} - \bar{\mathbf{x}}$ % $\bar{\mathbf{x}}$: mean of the training data
  $\mathbf{z} \leftarrow \mathbf{h}(\mathbf{x}')$
  $\mathbf{s} \leftarrow \mathbf{W}_{\text{SFA}}^T(\mathbf{z_n} - \bar{\mathbf{z}})$
  $\mathbf{s}' = (s_1, s_2, \ldots, s_J)$ % Extract the first J slow features
  $\mathbf{y}' \leftarrow \mathbf{R}\mathbf{s}'$
  $\mathbf{a} \leftarrow \mathbf{Q}\mathbf{y}' + \mathbf{d}$
  $\mathbf{b} \leftarrow \mathbf{x}' - \mathbf{a}$
  $\mathbf{c} \leftarrow \mathbf{W}_{\text{PCA}}^T\mathbf{b}$
  $\mathbf{y} = \mathbf{y}'|\mathbf{c}$
  return $\mathbf{y}$
end procedure

4.4 Mixing and scaling of slow features: QR scaling

In the iSFA algorithm, the J-dimensional slow features $\mathbf{s}'_n$ are transformed into the scaled $\mathbf{y}'$ features. Such a transformation is necessary to make the amplitude of the slow features compatible with the amplitude of the PCA features, so that PCA processing of the two sets of features together is possible and meaningful in the next layers.

In general, a feature scaling method should ideally offer two key properties of PCA. (1) If one adds unit-variance noise to one of the output features (e.g., principal components), the reconstruction error also increases by one unit on average. (2) If one adds independent noise to two or more output features simultaneously, the reconstruction error increases additively.

The feature scaling method (10)–(12) used by Algorithm 1 is called *QR scaling*, and it can be shown that it fulfills the two properties above. QR scaling ensures that the amplitude of the slow features is approximately equal to the reduction in the reconstruction error that each feature allows. In practice, a lower bound on the scales (not shown in the pseudo-code) ensures that all features have amplitudes > 0 even if they do not contribute to reconstruction.

In the iSFA algorithm, we approximate the input linearly as $\tilde{\mathbf{x}} = \tilde{\mathbf{a}} + \tilde{\mathbf{b}} + \bar{\mathbf{x}}$, where $\tilde{\mathbf{a}} = \mathbf{Q}\mathbf{y}' + \mathbf{d}$ and $\tilde{\mathbf{b}} = \mathbf{W}_{\text{PCA}}\mathbf{c}$ (see Sect. 4.5 and recall that $\mathbf{W}_{\text{PCA}}$ is the Moore-Penrose pseudoinverse of $\mathbf{W}_{\text{PCA}}^T$). Approximations are denoted here using tilded variables. Therefore, vector $\mathbf{y} = \mathbf{y}'|\mathbf{c}$ fulfills the two key properties of reconstruction of PCA above because matrix $\mathbf{Q}$ and matrix $\mathbf{W}_{\text{PCA}}$ are orthogonal, and because the columns of the two matrices are mutually orthogonal.

One small drawback is that (12) mixes the slow features, which is undesirable when one uses certain expansion functions. Therefore, as an alternative we propose a *sensitivity-based scaling method* in "Appendix B", which does not mix the slow features.

4.5 Input reconstruction from iSFA features

One interesting property of iSFA is that both the slow and the reconstructive features are nonlinear w.r.t. the input data. The slow features are nonlinear due to the expansion function. The residual data is nonlinear because it is computed using the (nonlinear) slow part and the centered data. The reconstructive features are computed using the residual data and are linear w.r.t. the residual data but nonlinear w.r.t. the input data.

Even though the computed features are all nonlinear, iSFA allows a linear approximation of the input (linear input reconstruction). In contrast, standard SFA does not have a standard input

reconstruction method, although various gradient-descent and vector-quantization methods have been tried (e.g., Wilbert 2012) with limited success.

The reconstruction algorithm simply computes the input approximation described in Sect. 4.4, and further detailed in Algorithm 3.

Algorithm 3 Linear input reconstruction for iSFA

Require: $D, \bar{\mathbf{x}}, \mathbf{W}_{\text{PCA}}, J, \mathbf{Q}, \mathbf{b}$
1: **procedure** LINEAR- RECONSTRUCTION($\mathbf{y}$)
2: $\quad \mathbf{y}' \leftarrow (y_1, \ldots, y_J)$ % Slow part
3: $\quad \mathbf{c} \leftarrow (y_{J+1}, \ldots, y_D)$ % Reconstructive part
4: $\quad \tilde{\mathbf{x}} \leftarrow (\mathbf{Q}\mathbf{y}' + \mathbf{d}) + \mathbf{W}_{\text{PCA}}\mathbf{c} + \bar{\mathbf{x}}$ % i.e., $\tilde{\mathbf{a}} + \tilde{\mathbf{b}} + \bar{\mathbf{x}}$
5: $\quad$ **return** $\tilde{\mathbf{x}}$
6: **end procedure**

Moreover, the linear reconstruction algorithm has practical properties: It is simpler than the feature extraction algorithm, since the nonlinear expansion $\mathbf{h}$ and $\mathbf{W}_{\text{SFA}}$ are not used, and it has lower computational complexity, because it consists of only two matrix-vector multiplications and three vector additions, none of them using expanded I'-dimensional data.

Alternatively, nonlinear reconstruction from iSFA features is also possible, as described in the algorithm below. Assume $\mathbf{y}$ is the iSFA feature representation of a sample $\mathbf{x}$, which we denote as $\mathbf{y} = \text{iSFA}(\mathbf{x})$. Since $\mathbf{x}$ is unknown, the reconstruction error cannot be computed directly. However, one can indirectly measure the accuracy of a particular reconstruction $\tilde{\mathbf{x}}$ by means of the feature error, which is defined here as $e_{\text{feat}} \stackrel{\text{def}}{=} ||\mathbf{y} - \text{iSFA}(\tilde{\mathbf{x}})||$. The feature error can be minimized for $\tilde{\mathbf{x}}$ using the function iSFA$(\cdot)$ as a black box and gradient descent or other generic nonlinear minimization algorithms. Frequently, such algorithms require a first approximation, which can be very conveniently provided by the linear reconstruction algorithm.

Although nonlinear reconstruction methods might result in higher reconstruction accuracy than linear methods in theory, they are typically more expensive computationally. Moreover, in the discussion we explain why minimizing e_{feat} typically increases the reconstruction error in practice.

4.6 Some remarks on iSFA and derivation of iGSFA and HiGSFA

Clearly, the computational complexity of iSFA is at least that of SFA, because iSFA consists of SFA and a few additional computations. However, none of the additional computations is done on the expanded I'-dimensional data but at most on I or D-dimensional data (e.g., PCA is applied to I-dimensional data, and the QR decomposition is applied to an $I \times I$-matrix, which have an $\mathcal{O}(NI^2 + I^3)$ and $\mathcal{O}(I^3)$ complexity, respectively). Therefore, iSFA is slightly slower than SFA but has the same complexity order. Practical experiments (Sect. 5) confirm this observation.

The presentation above focuses on iSFA. To obtain information-preserving GSFA (iGSFA) one only needs to substitute SFA by GSFA inside the iSFA algorithm and provide GSFA with the corresponding training graph during the training phase. Notice that GSFA features have weighted zero mean instead of the simple (unweighted) zero mean enforced by SFA. This difference has already been compensated by vector $\mathbf{d}$. HiGSFA is constructed simply by

connecting iGSFA nodes in a hierarchy, just as HSFA is constructed by connecting SFA nodes.

The iGSFA node has been implemented in Python (including iSFA) and is publicly available in Cuicuilco[3] as well as in the MDP toolkit (Zito et al. 2009).

5 Experimental evaluation of HiGSFA

This section evaluates the proposed extensions, HiSFA and HiGSFA, on three problems: (1) The unsupervised extraction of temporally stable features from images simulating the view of a rat walking inside a box, which is addressed with HiSFA. (2) The extraction of four pose parameters of faces in image patches, which is a multi-label learning problem suitable for HiGSFA. (3) The estimation of age, race, and gender from human face photographs, which is also solved using HiGSFA. The chosen problems exemplify the general applicability of the proposed algorithms.

5.1 Experiment 1: Extraction of slow features with HiSFA from the view of a rat

The input data in this first experiment consists of the (visual) sensory input of a simulated rat. The images have been created using the RatLab toolkit (Schönfeld and Wiskott 2013). RatLab simulates the random path of a rat as it explores its environment, rendering its view according to its current position and head-view angle. For simplicity the rat is confined to a square area bounded by four walls having different textures. Franzius et al. (2007) have shown theoretically and in simulations that the slowest features one extracted from this data are trigonometric functions of the position of the rat and its head direction (i.e., the slow configuration/generative parameters).

5.1.1 Training and test images

For this experiment, first a large fixed sequence of 200,000 images was generated. Then, the training data of a single experiment run is selected randomly as a contiguous subsequence of size 40,000 from the full sequence. The test sequence is selected similarly but enforcing no overlap with the training sequence. All images are in color (RGB) and have a resolution of 320×40 pixels, see Fig. 6.

5.1.2 Description of HiSFA and HSFA networks

To evaluate HiSFA we reproduced an HSFA network that has already been used (Schönfeld and Wiskott 2013). This network has three layers of quadratic SFA. Each layer consists of three components: (1) linear SFA, which reduces the data dimensionality to 32 features, (2) a quadratic expansion, and (3) linear SFA, which again reduces the expanded dimensionality to 32 features. The first two layers are convolutional (i.e., the nodes of these layers share the same weight matrices).

For a fair comparison, we built an HiSFA network with exactly the *same structure* as the HSFA network described above. The HiSFA network has additional hyperparameters Δ_T (one for each node), but in order to control the size of the slow part more precisely, we

[3] https://github.com/AlbertoEsc/cuicuilco.

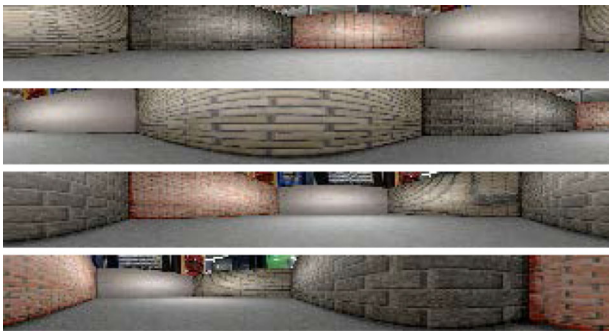

Fig. 6 Example of four images generated using RatLab. Notice that the images are quite elongated. This default image size approximates the panoramic field of view of a rat

removed this parameter and fixed instead the size of the slow part to 6 features in all nodes of the HiSFA network (i.e., $J = 6$).

For computational convenience we simplified the iSFA algorithm to enable the use of convolutional layers as in the HSFA network. Convolutional layers can be seen as a single iSFA node cloned at different spatial locations. Thus, the total input to a node consists not of a single input sequence (as assumed by the iSFA algorithm) but of several sequences, one at each location of the node. The iSFA node was modified as follows: (1) the decorrelation step (between the slow and reconstructive parts) is removed and (2) all slow features are given the same variance as the median variance of the features in the corresponding reconstructive part (instead of QR scaling).

5.1.3 Results

To evaluate HSFA and HiSFA we compute Δ values of the 3 slowest features extracted from test data, which are shown in Table 1. The experiments were repeated 5 times, each run using training and test sequences randomly computed using the procedure of Sect. 5.1.1.

Table 1 shows that HiSFA extracts clearly slower features than HSFA for both training and test data in the main setup (40 k training images). For instance, for test data Δ_1, Δ_2, and Δ_3 are 28–52% smaller in HiSFA than in HSFA. This is remarkable given that HSFA and HiSFA span the same global feature space (same model capacity).

In order to compare the robustness of HSFA and HiSFA w.r.t. the number of images in the training sequence, we also evaluate the algorithms using shorter training sequences of 5 k, 10 k, and 20 k images. As usual, the test sequences have 40k images. Table 1 shows that HiSFA computes slower features than HSFA given the same number of training images. This holds for both training and test data. In fact, when HiSFA is trained using only 5 k images the slowest extracted features are already slower than those extracted by HSFA trained on 40 k images (both for training and test data). In contrast, the performance of HSFA decreases significantly when trained on 10 k and 5 k images. Therefore, HiSFA is much more robust than HSFA w.r.t. the number of training images (higher sample efficiency).

5.2 Experiment 2: Estimation of face pose from image patches

The second experiment evaluates the accuracy of HiGSFA compared to HGSFA in a supervised learning scenario. We consider the problem of finding the pose of a face contained in

Table 1 Delta values of the first three features extracted by HSFA and HiSFA from training data (above) and test data (below)

Hierarchical network	Average Δ-values of the 3 slowest features (training data)		
	Δ_1	Δ_2	Δ_3
HSFA (40k)	0.00436 ± 0.00006	0.00472 ± 0.00004	0.00565 ± 0.00008
HiSFA (40k)	**0.00221** ± 0.00004	**0.00268** ± 0.00008	**0.00415** ± 0.00007
HSFA (20k)	0.00394 ± 0.00009	0.00484 ± 0.00009	0.00599 ± 0.00018
HiSFA (20k)	0.00203 ± 0.00002	0.00275 ± 0.00008	0.00429 ± 0.00009
HSFA (10k)	0.00368 ± 0.00015	0.00481 ± 0.00010	0.00638 ± 0.00024
HiSFA (10k)	0.00191 ± 0.00011	0.00273 ± 0.00010	0.00437 ± 0.00014
HSFA (5k)	0.00337 ± 0.00012	0.00434 ± 0.00013	0.00579 ± 0.00036
HiSFA (5k)	0.00166 ± 0.00003	0.00280 ± 0.00012	0.00446 ± 0.00015
Hierarchical network	Average Δ-values of the 3 slowest features (test data)		
	Δ_1	Δ_2	Δ_3
HSFA (40k)	0.00481 ± 0.00010	0.00508 ± 0.00004	0.00587 ± 0.00005
HiSFA (40k)	**0.00230** ± 0.00002	**0.00276** ± 0.00003	**0.00425** ± 0.00007
HSFA (20k)	0.00548 ± 0.00014	0.00528 ± 0.00016	0.00633 ± 0.00011
HiSFA (20k)	0.00251 ± 0.00004	0.00270 ± 0.00009	0.00430 ± 0.00008
HSFA (10k)	0.00653 ± 0.00023	0.00591 ± 0.00017	0.00789 ± 0.00025
HiSFA (10k)	0.00281 ± 0.00010	0.00285 ± 0.00008	0.00441 ± 0.00009
HSFA (5k)	0.01831 ± 0.00686	0.02166 ± 0.01307	0.01390 ± 0.00116
HiSFA (5k)	0.00353 ± 0.00016	0.00383 ± 0.00032	0.00488 ± 0.00015

Smallest delta values of each column have been highlighted in bold
Smaller values are better. In the proposed setup the training data consists of 40 k images, but we also show the results when using only 20 k, 10 k, and 5 k training images to illustrate the resistance of HiSFA and HSFA to overfitting. All results have been averaged over 5 runs and include the standard error of the mean. For the HiGSFA networks, the size of the slow part has been fixed to 6 features in all nodes, except in the experiment that uses 5 k training images, where the size of the slow part is 3 features. This hyperparameter was tuned using a different run and random seed

an image patch. Face pose is described by four parameters: (1) the horizontal and (2) vertical position of the face center (denoted by x-pos and y-pos, respectively), (3) the size of the face relative to the image patch, and (4) the in-plane rotation angle of the eyes-line. Therefore, we solve a regression problem on four real-valued labels. The resulting system can be easily applied to face tracking and to face detection with an additional face discrimination step.

5.2.1 Generation of the image patches

The complete dataset consists of 64,470 images that have been extracted from a few datasets to increase image variability: Caltech (Fink et al. 2003), CAS-PEAL (Gao et al. 2008), FaceTracer (Kumar et al. 2008), FRGC (Phillips et al. 2005), and LFW (Huang et al. 2007). In each run of the system two disjoint image subsets are randomly selected, one of 55,000 images, used for training, and another of 9000 images, used for testing. The images are processed in two steps. First they are normalized (i.e., centered, scaled, and rotated). Then, they are rescaled to 64×64 pixels and are systematically randomized: In the resulting image patches the center of the face deviates horizontally from the image center by at most ± 20

Fig. 7 Examples of image patches after pose randomization illustrating the range of variations in pose. The eyes of the subjects have been pixelated for privacy and copyright reasons

pixels, vertically by at most ± 10 pixels, the angle of the eye-line deviates from the horizontal by at most ± 22.5 deg, and the size of the largest and smallest faces differs by a factor of $\sqrt{2}$ (a factor of 2 in their area). The concrete pose parameters are sampled from uniform distributions in the above-mentioned ranges. To increase sample efficiency, each image of the training set is used twice with different random distortions, thus the effective training set has size 110,000. Examples of the final image patches are shown in Fig. 7.

5.2.2 HiGSFA and HGSFA networks

For comparison purposes, we adopt an HGSFA network that has been previously designed (and partially tuned) to estimate facial pose parameters by Escalante-B. and Wiskott (2013) without modification. Most SFA nodes of this network consist of expansion function

$$0.8\text{Exp}(x_1, x_2, \ldots, x_I) \stackrel{\text{def}}{=} (x_1, x_2, \ldots, x_I, |x_1|^{0.8}, |x_2|^{0.8}, \ldots, |x_I|^{0.8}), \tag{13}$$

followed by linear SFA, except for the SFA nodes of the first layer, which have an additional preprocessing step that uses PCA to reduce the number of dimensions from 16 to 13. In contrast to the RatLab networks, this network does not use weight sharing, increasing feature specificity at each node location.

For a fair comparison, we construct an HiGSFA network having the same structure as the HGSFA network (e.g., same number of layers, nodes, expansion function, data dimensions, receptive fields). Similarly to experiment 1, we directly set the number of slow features preserved by the iGSFA nodes, which in this experiment varies depending on the layer from 7 to 20 features (these values have been roughly tuned using a run with a random seed not used for testing). These parameters used to construct the networks are shown in Table 2.

5.2.3 Training graphs that encode pose parameters

In order to train HGSFA and HiGSFA one needs a training graph (i.e., a structure that contains the samples and the vertex and edge weights). A few efficient predefined graphs have already been proposed (Escalante-B. and Wiskott 2013), allowing training of GSFA with a complexity of $\mathcal{O}(NI^2 + I^3)$, which is of the same order as SFA, making this type of graphs competitive in terms of speed. One example of a training graph for classification is the clustered graph (see Sect. 5.3.2) and one for regression is the serial training graph, described below.

Serial training graph (Escalante-B. and Wiskott 2013) The features extracted using this graph typically approximate a monotonic function of the original label and its higher frequency harmonics. To solve a regression problem and generate label estimates in the appropriate domain, a few slow features (e.g., extracted using HGSFA or HiGSFA) are post-processed by an explicit regression step. There are more training graphs suitable for regression (e.g., mixed graph), but this one has consistently given good results in different experiments.

Table 2 Description of the HiGSFA and HGSFA networks used for pose estimation

Layer	Size	Node fan-in	Stride	Output dim. per node	Size of slow part
0	64 × 64 pixels	–	–	–	–
1	16 × 16 nodes	4 × 4	4 × 4	13	7
2	8 × 16 nodes	2 × 1	2 × 1	20	8
3	8 × 8 nodes	1 × 2	1 × 2	35	10
4	4 × 8 nodes	2 × 1	2 × 1	60	14
5	4 × 4 nodes	1 × 2	1 × 2	60	20
6	2 × 4 nodes	2 × 1	2 × 1	60	20
7	2 × 2 nodes	1 × 2	1 × 2	60	20
8	1 × 2 nodes	2 × 1	–	60	20
9	1 × 1 nodes	1 × 2	–	60	20

Both networks have the same structure. Layer 0 denotes the input image, whereas layer 9 is the top node. The parameter "size of slow part" (used only by HiGSFA) indicates the number of slow features preserved by the nodes of the respective layer and eliminates parameter Δ_T

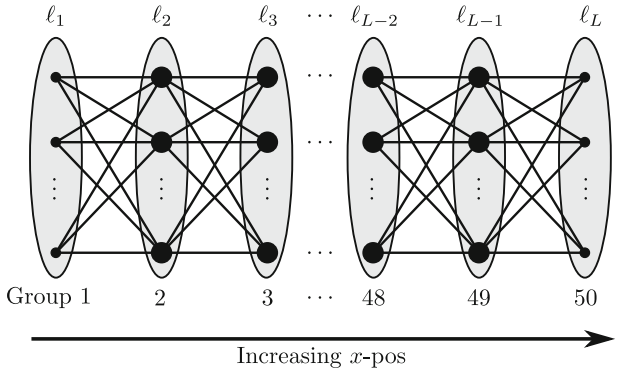

Fig. 8 Illustration of a serial training graph used to learn the position of the face center (x-pos). The training images are first ordered by increasing x-pos and then grouped into $L = 50$ groups of $N_g = 2200$ images each. Each dot represents an image, edges represent connections, and ovals represent the groups. The images of the first and last group have weight 1 and the remaining images have weight 2 (image weights are represented by smaller/bigger dots). All edges have a weight of 1. The serial graphs for learning x-pos, y-pos, angle, and scale are constructed in the same way.

Figure 8 illustrates a serial graph useful to learn x-pos. In general, a serial graph is constructed by ordering the samples by increasing label. Then, the samples are partitioned into L groups of size $N_g = N/L$. A representative label $\in \{\ell_1, \ldots, \ell_L\}$ is assigned to each group, where $\ell_1 < \ell_2 < \cdots < \ell_L$. Edges connect all pairs of samples from two consecutive groups with group labels (ℓ_l and ℓ_{l+1}). Thus, all connections are inter-group, no intra-group connections are present. Notice that since any two vertices of the same group are adjacent to exactly the same neighbors, they are likely to be mapped to similar outputs by GSFA. We use this procedure to construct four serial graphs $\mathcal{G}_{x\text{-pos}}$, $\mathcal{G}_{y\text{-pos}}$, $\mathcal{G}_{\text{angle}}$, and $\mathcal{G}_{\text{scale}}$ that encode the x-pos, y-pos, angle, and scale label, respectively. All of them have the same parameters: $L = 50$ groups, $N = 110{,}000$ samples, and $N_g = 2200$ samples per group.

Combined graph to learn all pose parameters One disadvantage of current pre-defined graphs is that they allow to learn only a single (real-valued or categorical) label. In order to learn

various labels simultaneously, we resort to a method proposed earlier that allows the combination of graphs (Escalante-B. and Wiskott 2016). We use this method here for the first time on *real data* to create an efficient graph that encodes all four labels combining $\mathcal{G}_{x\text{-pos}}$, $\mathcal{G}_{y\text{-pos}}$, $\mathcal{G}_{\text{angle}}$, and $\mathcal{G}_{\text{scale}}$. The combination preserves the original samples of the graphs, but the vertex and edge weights are added, which is denoted $\mathcal{G}'_{4\text{L}} \stackrel{\text{def}}{=} \mathcal{G}_{x\text{-pos}} + \mathcal{G}_{y\text{-pos}} + \mathcal{G}_{\text{angle}} + \mathcal{G}_{\text{scale}}$.

The combination of graphs guarantees that the slowest optimal free responses of the combined graph span the slowest optimal free responses of the original graphs, as long as three conditions are fulfilled: (1) all graphs have the same samples and are consistent, (2) all graphs have the same (or proportional) node weights, and (3) optimal free responses that are slow in one graph ($\Delta < 2.0$) should not be fast ($\Delta > 2.0$) in any other graph. Since the labels (i.e., pose parameters) have been computed randomly and independently of each other, these conditions are fulfilled on average (e.g., for the $\mathcal{G}_{x\text{-pos}}$ graph, any feature $\mathbf{y}$ that solely depends on y-pos, scale, or angle has $\Delta_{\mathbf{y}}^{\mathcal{G}_{x\text{-pos}}} \approx 2.0$).

However, the naive graph $\mathcal{G}'_{4\text{L}}$ does not take into account that the 'angle' and 'scale' labels are more difficult to estimate compared to the x-pos and y-pos labels. Thus, the feature representation is dominated by features that encode the easier labels (and their harmonics) and only few features encode the difficult labels, making their estimation even more difficult. To solve this problem, we determine weighting factors such that each label is represented at least once in the first five slow features. Features that are more difficult to extract have higher weights to avoid an implicit focus on easy features. The resulting graph and weighting factors are: $\mathcal{G}_{4\text{L}} \stackrel{\text{def}}{=} \mathcal{G}_{x\text{-pos}} + 1.25\mathcal{G}_{y\text{-pos}} + 1.5\mathcal{G}_{\text{angle}} + 1.75\mathcal{G}_{\text{scale}}$.

Supervised post-processing We extract 20 slow features from the training dataset to train four separate Gaussian classifiers (GC), one for each label. Ground truth classes are generated by discretizing the labels into 50 values representing class 1–50. After the GC has been trained with this data, the pose estimate (on the test patches) is computed using the soft-GC method (Escalante-B. and Wiskott 2013), which exploits class membership probabilities of the corresponding classifier: Let $\mathbf{P}(C_{\ell_l}|\mathbf{y})$ be the estimated class probability that the input sample $\mathbf{x}$ with feature representation $\mathbf{y} = \mathbf{g}(\mathbf{x})$ belongs to the group with average label ℓ_l. Then, the estimated label is

$$\tilde{\ell} \stackrel{\text{def}}{=} \sum_{l=1}^{50} \ell_l \cdot \mathbf{P}(C_{\ell_l}|\mathbf{y})\,. \tag{14}$$

Equation (14) has been designed to minimize the root mean squared error (RMSE), and although it incurs an error due to the discretization of the labels, the soft nature of the estimation has provided good accuracy and low percentage of misclassifications.

5.2.4 Results: Accuracy of HiGSFA for pose estimation

The HiGSFA and HGSFA networks were trained using graph $\mathcal{G}_{4\text{L}}$ described above. Table 3 shows the estimation error of each pose parameter. The results show that HiGSFA yields more accurate estimations than HGSFA for all pose parameters. We would like to remark that the HiGSFA network has the same structure as the HGSFA network, which has been tuned for the current problem. However, one may adapt the network structure specifically for HiGSFA to further exploit the advantages of this algorithm. Concretely, the improved robustness of HiGSFA allows to handle more complex networks (e.g., by increasing the output dimensionality of the nodes or using more complex expansion functions). In the following

Table 3 Estimation errors (RMSE) of HGSFA and HiGSFA for each pose parameter

Estimation errors (RMSE)	HGSFA ($\mathcal{G}_{4L}$)	HiGSFA ($\mathcal{G}_{4L}$)	Chance level
x-pos	2.281 ± 0.030 (17F)	**1.843** ± 0.020 (14F)	11.547 ± 0.000
y-pos	1.874 ± 0.017 (12F)	**1.753** ± 0.017 (12F)	5.921 ± 0.006
Angle	6.678 ± 0.029 (9F)	**6.102** ± 0.021 (9F)	12.970 ± 0.023
Scale	0.0614 ± 0.0002 (9F)	**0.0594** ± 0.0002 (9F)	0.0829 ± 0.0001

Best results for each parameter are shown in bold
The number of slow features that were post-processed by the soft-GC regression step is indicated in parenthesis. This hyperparameter was tuned for each algorithm and pose parameter using two random runs different from those of the evaluation. All results were averaged over five runs and include the standard error of the mean

experiment on age estimation, the network structure of HiGSFA and its hyperparameters are tuned, yielding higher accuracy.

5.3 Experiment 3: Age estimation from frontal face photographs

Systems for age estimation from photographs have many applications in areas such as human-computer interaction, group-targeted advertisement, and security. However, age estimation is a challenging task, because different persons experience facial aging differently depending on several intrinsic and extrinsic factors.

The first system for age estimation based on SFA was a four-layer HSFA network that processes raw images without prior feature extraction (Escalante-B. and Wiskott 2010). The system was trained on synthetic input images created using special software for 3D-face modeling. However, the complexity of the face model was probably too simple, which allowed linear SFA (in fact linear GSFA) to achieve good performance, and left open the question of whether SFA/GSFA could also be successful on real photographs.

This subsection first describes the image pre-processing method. Then, a training graph used to learn age, race, and gender simultaneously is presented. Finally, an HiGSFA network is described and evaluated according to three criteria: feature slowness, age estimation error (compared with state-of-the-art algorithms), and linear reconstruction error.

5.3.1 Image database and pre-processing

The MORPH-II database (i.e. MORPH, Album 2, Ricanek Jr. and Tesafaye 2006) is a large database suitable for age estimation. It contains 55,134 images of about 13,000 different persons with ages ranging from 16 to 77 years. The images were taken under partially controlled conditions (e.g. frontal pose, good image quality and lighting), and include variations in head pose and expression. The database annotations include age, gender (M or F), and "race" ("black", "white", "Asian", "Hispanic", and "other", denoted by B, W, A, H, and O, respectively) as well as the coordinates of the eyes. The procedure used to assign the race label does not seem to be documented. Most of the images are of black (77%) or white races (19%), making it probably more difficult to generalize to other races, such as Asian.

We follow the evaluation method proposed by Guo and Mu (2014), which has been used in many other works. In this method, the input images are partitioned in 3 disjoint sets S_1 and S_2 of 10, 530 images, and S_3 of 34,074 images. The racial and gender composition of S_1 and S_2 is the same: about 3 times more images of males than females and the same number

of white and black people. Other races are omitted. More exactly, $|MB| = |MW| = 3980$, $|FB| = |FW| = 1285$. The remaining images constitute the set S_3, which is composed as follows: $|MB| = 28{,}872$, $|FB| = 3187$, $|MW| = 1$, $|FW| = 28$, $|MA| = 141$, $|MH| = 1667$, $|MO| = 44$, $|FA| = 13$, $|FH| = 102$ and $|FO| = 19$. The evaluation is done twice by using either S_1 and S_1-test $\stackrel{\text{def}}{=} S_2 + S_3$ or S_2 and S_2-test $\stackrel{\text{def}}{=} S_1 + S_3$ as training and test sets, respectively.

We pre-process the input images in two steps: pose normalization and face sampling (Fig. 2). The pose-normalization step fixes the position of the eyes ensuring that: (a) the eye line is horizontal, (b) the inter-eye distance is constant, and (c) the output resolution is 256×260 pixels. After pose normalization, a face sampling step selects the head area only, enhances the contrast, and scales down the image to 96×96 pixels.

In addition to the S_1, S_2, and S_3 datasets, three extended datasets (DR, S, and T) are defined in this work: A DR-dataset is used to train HiGSFA to perform dimensionality reduction, an S-dataset is used to train the supervised step on top of HiGSFA (a Gaussian classifier), and a T-dataset is used for testing. The DR and S-datasets are created using the same set of training images (either S_1 or S_2), and the T-dataset using the corresponding test images, either S_1-test or S_2-test.

The images of the DR and S-datasets go through a random distortion step during face sampling, which includes a small random translation of max ±1.4 pixels, a rotation of max ±2 degrees, a rescaling of ±4%, and small fluctuations in the average color and contrast. The exact distortions are sampled uniformly from their respective ranges. Although these small distortions are frequently imperceptible, they teach HiGSFA to become invariant to small errors during image normalization and are necessary due to its feature specificity to improve generalization to test data. Other algorithms that use pre-computed features, such as BIF, or particular structures (e.g., convolutional layers, max pooling) are mostly invariant to such small transformations by construction (e.g., Guo and Mu 2014).

Distortions allow us to increase the number of training images. The images of the DR-dataset are used 22 times, each time using a different random distortion, and those of the S-dataset 3 times, resulting in 231,660 and 31,590 images, respectively. The images of the T-dataset are not distorted and used only once.

5.3.2 A multi-label training graph for learning age, gender, and race

We create an efficient training graph by combining three pre-defined graphs: a serial graph for age estimation and two clustered graphs (one for gender and the other for race classification).

Clustered training graphs The clustered graph generates features useful for classification that are equivalent to those of FDA (see Klampfl and Maass 2010, also compare Berkes 2005a and Berkes 2005b). This graph is illustrated in Fig. 9. The optimization problem associated with this graph explicitly demands that samples from the same class should be mapped to similar outputs. If C is the number of classes, $C - 1$ output (slow) features can be extracted and passed to a standard classifier, which computes the final class estimate.

The graph used for gender classification is a clustered graph that has only two classes (female/male) of $N_F = 56{,}540$ and $N_M = 175{,}120$ samples, respectively. The graph used for race classification is similar to the graph above: Only two classes are considered (B and W), and the number of samples per class is $N_B = N_W = 115{,}830$.

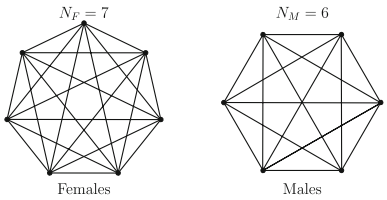

Fig. 9 Illustration of a *clustered* training graph for gender classification with 7 images of females and 6 of males. Each vertex represents an image and edges represent transitions. In general, pairs of images $\mathbf{x}(n)$ and $\mathbf{x}(n')$, with $n \neq n'$, that belong to the same class $s \in \{\text{F}, \text{M}\}$ are connected with an edge weight $\gamma_{n,n'} = 1/(N_s - 1)$, where N_s is the number of images in class s. This results in 2 fully connected subgraphs. Images of different genders are not connected. The weight of all vertices is equal to one. For the actual experiments, we use $N_F = 56{,}540$ and $N_M = 175{,}120$.

Serial training graph for age estimation Serial graphs have been described in Sect. 5.2.3. To extract age-related features, we create a serial graph with $L = 32$ groups, where each group has 7238 images.

Efficient graph for age, race, and gender estimation We use again the method for multiple label learning (Escalante-B. and Wiskott 2016) to learn age, race, and gender labels, by constructing a graph $\mathcal{G}_{3\text{L}}$ that combines a serial graph for age estimation, a clustered graph for gender, and a clustered graph for race. Whereas the vertex weights of the clustered graph are constant, the vertex weights of the serial graph are not (first and last groups have smaller vertex weights), but we believe this does not affect the accuracy of the combined graph significantly. For comparison purposes, we also create a serial graph $\mathcal{G}_{1\text{L}}$ that only learns age.

The graph combination method yields a compact feature representation. For example, one can combine a clustered graph for gender (M or F) estimation and another for race (B or W). The first 2 features learned from the resulting graph are then enough for gender *and* race classification. Alternatively, one could create a clustered graph with four classes (MB, MW, FB, FW), but to ensure good classification accuracy one must keep 3 features instead of 2. Such a representation would be impractical for larger numbers of classes. For example, if the number of classes were $C_1 = 10$ and $C_2 = 12$, one would need to extract $C_1 C_2 - 1 = 119$ features, whereas with the proposed graph combination, one would only need to extract $(C_1 - 1) + (C_2 - 1) = 20$ features.

Supervised post-processing We use the first 20 or fewer features extracted from the S-dataset to train three separate Gaussian classifiers, one for each label. For race and gender only two classes are considered (B, W, M, F). For age, the images are ordered by increasing age and partitioned in 39 classes of the same size. This hyperparameter has been tuned independently of the number of groups in the age graph, which is 32. The classes have average ages of $\{16.6, 17.6, 18.4, \ldots, 52.8, 57.8\}$ years. To compute these average ages, as well as to order the samples by age in the serial graph, the exact birthday of the persons is used, representing age with a day resolution (e.g., an age may be expressed as 25.216 years).

The final age estimation (on the T-dataset) is computed using the soft-GC method (14), except that 39 groups are used instead of 50. Moreover, to comply with the evaluation protocol, we use integer ground-truth labels and truncate the age estimates to integers.

Table 4 Description of the HiGSFA and HGSFA networks

Layer	Size	Node fan-in	Stride	Output dim. per node	
				HGSFA	HiGSFA
0	96×96 pixels	–	–	–	–
1	31×31 nodes	6×6	3×3	14	18
2	15×31 nodes	3×1	2×1	20	27
3	15×15 nodes	1×3	1×2	27	37
4	7×15 nodes	3×1	2×1	49	66
5	7×7 nodes	1×3	1×2	60	79
6	3×7 nodes	3×1	2×1	61	88
7	3×3 nodes	1×3	1×2	65	88
8	1×3 nodes	3×1	1×1	65	93
9	1×1 nodes	1×3	–	66	95
10	1×1 nodes	1×1	–	75	75

Both networks have the same number of nodes and general structure, but they differ in the type of nodes and in the number of features preserved by them, individually optimized for best performance. Layer 0 denotes the input image, whereas layer 10 is the top node

5.3.3 Evaluated algorithms

We compare HiGSFA to other algorithms: HGSFA, PCA, and state-of-the-art age-estimation algorithms. The structure of the HiGSFA and HGSFA networks is described in Table 4. In both networks, the nodes are simply an instance of iGSFA or GSFA preceded by different linear or nonlinear expansion functions, except in the first layer, where PCA is applied to the pixel data to preserve 20 out of 36 principal components prior to the expansion. The method used to scale the slow features is the sensitivity method, described in "Appendix B". The hyperparameters have been hand-tuned to achieve best accuracy on age estimation using educated guesses, sets S_1, S_2 and S_3 different to those used for the evaluation, and fewer image multiplicities to speed up the process.

The proposed HGSFA/HiGSFA networks are different in several aspects from SFA networks used so far (e.g., Franzius et al. 2007). For example, to improve feature specificity at the lowest layers, no weight sharing is used. Moreover, the input to the nodes (fan-in) originates mostly from the output of 3 nodes in the preceding layer (3×1 or 1×3). Such small fan-ins reduce the computational cost because they minimize the input dimensionality. The resulting networks have 10 layers.

The employed expansion functions consist of different nonlinear functions on subsets of the input vectors and include: (1) The identity function $\mathrm{I}(\mathbf{x}) = \mathbf{x}$, (2) quadratic terms $\mathrm{QT}(\mathbf{x}) \stackrel{\text{def}}{=} \{x_i x_j\}_{i,j=1}^{I}$, (3) a normalized version of QT: $\mathrm{QN}(\mathbf{x}) \stackrel{\text{def}}{=} \{\frac{1}{1+||\mathbf{x}||^2} x_i x_j\}_{i,j=1}^{I}$, (4) the terms $0.8\mathrm{ET}(\mathbf{x}) \stackrel{\text{def}}{=} \{|x_i|^{0.8}\}_{i=1}^{I}$ of the 0.8Exp expansion, and (5) the function $\mathrm{MAX2}(\mathbf{x}) \stackrel{\text{def}}{=} \{\max(x_i, x_{i+1})\}_{i=1}^{I-1}$. The MAX2 function is proposed here inspired by state-of-the-art CNNs for age estimation (Yi et al. 2015; Xing et al. 2017) that include max pooling or a variant of it. As a concrete example of the nonlinear expansions employed by the HiGSFA network, the expansion of the first layer is $\mathrm{I}(x_1, \ldots, x_{18})\,|0.8\mathrm{ET}(x_1, \ldots, x_{15})\,|\,\mathrm{MAX2}(x_1, \ldots, x_{17})\,|\,\mathrm{QT}(x_1, \ldots, x_{10})$, where | indicates vector concatenation. The expansions used in the remaining layers can be found in the available source code.

Table 5 Number of output features passed to the supervised step, a Gaussian classifier

Algorithm	Age	Race	Gender
HiGSFA ($\mathcal{G}_{3L}$)	5	6	4
HGSFA ($\mathcal{G}_{3L}$)	5	5	7
PCA	54	54	60

The parameter Δ_T of layers 3 to 10 is set to 1.96. Δ_T is not used in layers 1 and 2, and instead the number of slow features is fixed to 3 and 4, resp. The number of features given to the supervised algorithm, shown in Table 5, has been tuned for each DR algorithm and supervised problem.

Since the data dimensionality allows it, PCA is applied directly (it was not resorted to hierarchical PCA) to provide more accurate principal components and smaller reconstruction errors.

5.3.4 Experimental results

The results of HiGSFA, HGSFA and PCA (as well as other algorithms, where appropriate) are presented from three angles: feature slowness, age estimation error, and reconstruction error. Individual scores are reported as $a \pm b$, where a is the average over the test images (S_1-test and S_2-test), and b is the standard error of the mean (i.e., half the absolute difference).

Feature slowness The weighted Δ values of GSFA (Eq. 1) are denoted here as $\Delta_j^{\mathrm{DR},\mathcal{G}_{3L}}$ and depend on the graph $\mathcal{G}_{3L}$, which in turn depends on the training data and the labels. To measure slowness (or rather fastness) of test data (T), standard Δ values are computed using the images ordered by increasing age label, $\Delta_j^{\mathrm{T,lin}} \stackrel{\mathrm{def}}{=} \frac{1}{N-1}\sum_n (y_j(n+1) - y_j(n))^2$. The last expression is equivalent to a weighted Δ value using a linear graph (Fig. 3b). In all cases, the features are normalized to unit variance before computing their Δ values to allow for fair comparisons in spite of the feature scaling method.

Table 6 shows $\Delta_{1,2,3}^{\mathrm{DR},\mathcal{G}_{3L}}$ (resp. $\Delta_{1,2,3}^{\mathrm{T,lin}}$), that is, the Δ values of the three slowest features extracted from the DR-dataset (resp. T-dataset) using the graph $\mathcal{G}_{3L}$ (resp. a linear graph). HiGSFA maximizes slowness better than HGSFA. The $\Delta^{\mathrm{T,lin}}$ values of the PCA features are larger, which is not surprising, because PCA does not optimize for slowness. Since $\Delta^{\mathrm{DR},\mathcal{G}_{3L}}$ and $\Delta^{\mathrm{T,lin}}$ are computed from different graphs, they should not be compared with each other. $\Delta^{\mathrm{T,lin}}$ considers transitions between images with the same or very similar ages but arbitrary race and gender. $\Delta^{\mathrm{DR},\mathcal{G}_{3L}}$ only considers transitions between images having at least one of a) the same gender, b) the same race, or c) different but consecutive age groups.

Age estimation error We treat age estimation as a regression problem with estimates expressed as an integer number of years, and use three metrics to measure age estimation accuracy: (1) the mean absolute error (MAE) (see Geng et al. 2007), which is the most frequent metric for age estimation, (2) the root mean squared error (RMSE), which is a common loss function for regression. Although it is sensitive to outliers and has been barely used in the literature on age estimation, some applications might benefit from its stronger penalization of large estimation errors. And (3) cumulative scores (CSs, see Geng et al. 2007), which indicate the fraction of the images that have an estimation error below a given threshold. For instance, CS(5) is the fraction of estimates (e.g., expressed as a percentage) having an error of at most 5 years w.r.t. the real age.

Table 6 Average delta values of the first three features extracted by PCA, HGSFA, and HiGSFA on either training (DR) or test (T) data

	PCA	HGSFA ($\mathcal{G}_{3L}$)	HiGSFA ($\mathcal{G}_{3L}$)
$\Delta_1^{\mathrm{DR},\mathcal{G}_{3L}}$	–	1.23	**1.17**
$\Delta_2^{\mathrm{DR},\mathcal{G}_{3L}}$	–	1.46	**1.38**
$\Delta_3^{\mathrm{DR},\mathcal{G}_{3L}}$	–	1.56	**1.53**
$\Delta_1^{\mathrm{T,lin}}$	1.99	0.45	**0.38**
$\Delta_2^{\mathrm{T,lin}}$	1.93	1.12	**0.99**
$\Delta_3^{\mathrm{T,lin}}$	1.99	**1.90**	**1.90**

Smallest delta values for each row have been highlighted in bold
Delta values for test data are computed using a linear graph (lin) with samples ordered by age, thus smaller values imply better age discrimination. The first feature extracted (both by HiGSFA and HGSFA) has the smallest delta value, indicating that it is the main feature encoding age. For comparison, the Δ value of unit-variance i.i.d. noise is 2.0.

Table 7 Accuracy in years of state-of-the-art algorithms for age estimation on the MORPH-II database (test data)

Algorithm	Age (MAE)	Age (RMSE)	Race (CR)	Gender (CR)
BIF+3Step (Guo and Mu 2010)	4.45 ± 0.01	–	$98.80\% \pm 0.04$	$97.84\% \pm 0.16$
BIF+KPLS (Guo and Mu 2011)	4.18 ± 0.03	–	$98.85\% \pm 0.05$	$98.20\% \pm 0.00$
BIF+rKCCA (Guo and Mu 2014)	3.98 ± 0.03	–	$99.00\% \pm 0.00$	$98.45\% \pm 0.05$
BIF+rKCCA+SVM (Guo and Mu 2014)	3.92 ± 0.02	–	–	–
Baseline CNN (Yi et al. 2015)	4.60 ± 0.05	–	–	–
MCNN no align (Yi et al. 2015)	3.79 ± 0.09	–	–	–
MCNN only age (Yi et al. 2015)	3.63 ± 0.00	–	–	–
MCNN (Yi et al. 2015)	3.63 ± 0.09	–	$98.6\% \pm 0.05$	$97.9\% \pm 0.1$
MRCNN (Liu et al. 2017)	3.48	–	–	–
$\mathrm{Net}_{\mathrm{hybrid}}^{\mathrm{VGG}}$ (Xing et al. 2017)	**2.96** ± 0.01	–	99.2%	**98.7%**
PCA (control)	6.804 ± 0.007	8.888 ± 0.000	$96.75\% \pm 0.06$	$91.54\% \pm 0.24$
HGSFA ($\mathcal{G}_{3L}$) (control)	3.921 ± 0.018	5.148 ± 0.049	$98.60\% \pm 0.08$	$96.40\% \pm 0.12$
HiGSFA ($\mathcal{G}_{1L}$) (control)	3.605 ± 0.001	4.690 ± 0.000	–	–
HiGSFA ($\mathcal{G}_{3L}$) (proposed)	3.497 ± 0.008	4.583 ± 0.000	$99.15\% \pm 0.01$	$97.70\% \pm 0.01$
HiGSFA-mirroring ($\mathcal{G}_{3L}$)	3.412 ± 0.004	**4.524** ± 0.014	**99.26%** ± 0.01	$98.51\% \pm 0.07$
Chance level	9.33	10.95	87.58%	86.73%

Top results for each column are shown in bold
Classification rates (CR) for race and gender estimation are also provided. The chance level is the best possible performance when the estimation is constant

The accuracies are summarized in Table 7. The MAE of HGSFA is 3.921 years, which is better than that of BIF+3Step, BIF+KPLS, BIF+rKCCA, and a baseline CNN, similar to BIF+rKCCA+SVM, and worse than the MCNNs. The MAE of HiGSFA is 3.497 years, clearly better than HGSFA and also better than MCNNs. At first submission of this publication, HiGSFA achieved the best performance, but newer CNN-based methods have now improved the state-of-the-art performance. In particular, MRCNN yields an MAE of 3.48 years, and $\mathrm{Net}_{\mathrm{hybrid}}^{\mathrm{VGG}}$ an MAE of only 2.96 years. In contrast, PCA has the largest MAE, namely 6.804 years.

Table 8 Percentual cumulative scores (the larger the better) for various maximum allowed errors ranging from 0 to 30 years

Algorithm	cs (0)	cs (1)	cs (2)	cs (3)	cs (4)	cs (5)	cs (6)	cs (7)	cs (8)	cs (9)	cs (10)
HGSFA	8.80	26.42	42.41	55.80	66.38	74.86	81.31	86.37	90.12	92.93	94.96
HiGSFA	9.87	29.16	46.23	60.14	71.04	79.56	85.70	90.04	93.12	95.45	96.92
Algorithm	cs (11)	cs (12)	cs (14)	cs (16)	cs (18)	cs (20)	cs (22)	cs (24)	cs (26)	cs (28)	cs (30)
HGSFA	96.43	97.42	98.67	99.34	99.69	99.85	99.92	99.97	99.97	99.98	99.99
HiGSFA	97.91	98.56	99.34	99.70	99.84	99.91	99.94	99.96	99.97	99.98	99.99

MCNN denotes a multi-scale CNN (Yi et al. 2015) that has been trained on images decomposed as 23 48×48-pixel image patches. Each patch has one out of four different scales and is centered on a particular facial landmark. A similar approach was followed by Liu et al. (2017) to propose the use of a multi-region CNN (MRCNN). Xing et al. (2017) evaluated several CNN architectures and loss functions and propose the use of special-purpose hybrid architecture ($\text{Net}^{\text{VGG}}_{\text{hybrid}}$) with five VGG-based branches. One branch estimates a distribution on demographic groups (black female, black male, white female, white male), and the remaining four branches estimate age, where each branch is specialized in a single demographic group. The demographic distribution is used to combine the outputs of the four branches to generate the final age estimate.

In an effort to improve our performance, we also tried support vector regression (SVR) as supervised post-processing and computed an average age estimate using the original images and their mirrored version (HiGSFA-mirroring). This slightly improved the estimation to 3.412 years, becoming better than MRCNN. Mirroring is also done by MCNN and $\text{Net}^{\text{VGG}}_{\text{hybrid}}$.

Detailed cumulative scores for HiGSFA and HGSFA are provided in Table 8, facilitating future comparisons with other methods. The RMSE of HGSFA on test data is 5.148 years, whereas HiGSFA yields an RMSE of 4.583 years, and PCA an RMSE of 8.888 years. The RMSE of other approaches does not seem to be available.

The poor accuracy of PCA for age estimation is not surprising, because principal components might lose wrinkles, skin imperfections, and other information that could reveal age. Another reason is that principal components are too unstructured to be properly untangled by the soft GC method, in contrast to slow features, which have a very specific and simple structure.

The behavior of the estimation errors of HiGSFA is plotted in Fig. 10 as a function of the real age. On average, older persons are estimated much younger than they really are. This is in part due to the small number of older persons in the database, and because the oldest class used in the supervised step (soft-GC) has an average of about 58 years, making this the largest age that can be estimated by the system. The MAE is surprisingly low for persons below 45 years. The most accurate estimation is an MAE of only 2.253 years for 19-year-old persons.

Reconstruction error A reconstruction error is a measure of how much information of the original input is contained in the output features. In order to compute it, we assume a linear global model for input reconstruction.

Let $\mathbf{X}$ be the input data and $\mathbf{Y}$ the corresponding set of extracted features. A matrix $\mathbf{D}$ and a vector $\mathbf{c}$ are learned from the DR-dataset using linear regression (ordinary least squares) such that $\hat{\mathbf{X}} \stackrel{\text{def}}{=} \mathbf{DY} + \mathbf{c}\mathbf{1}^T$ approximates $\mathbf{X}$ as closely as possible, where $\mathbf{1}$ is a vector of N ones.

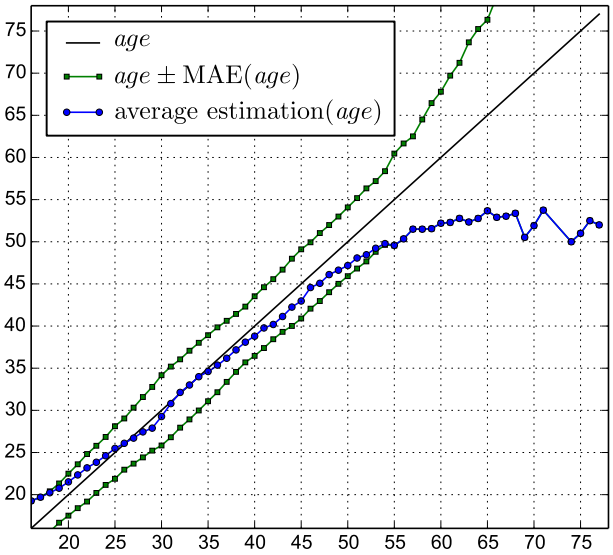

Fig. 10 The average age estimates of HiGSFA are plotted as a function of the real age. The MAE is also computed as a function of the real age and plotted as $age \pm \mathrm{MAE}(age)$.

Table 9 Reconstruction errors on test data using 75 features extracted by various algorithms

	Chance level	HGSFA	HiGSFA	PCA
e_{rec}	1.0	0.818	0.338	**0.201**

The best reconstruction error is shown in bold
The original dimensionality is $96^2 = 9216$ components

Thus, $\hat{\mathbf{X}}$ contains the reconstructed samples (i.e. $\hat{\mathbf{x}}_n \stackrel{\mathrm{def}}{=} \mathbf{D}\mathbf{y}_n + \mathbf{c}$ is the reconstruction of the input $\mathbf{x}_n$ given its feature representation $\mathbf{y}_n$). Figure 2 shows examples of face reconstructions using features extracted by different algorithms.

The model is linear and global, which means that output features are mapped to the input domain linearly. For PCA this gives the same result as the usual multiplication with the transposed projection matrix plus image average. An alternative (local) approach for HiGSFA would be to use the linear reconstruction algorithm of each node to perform reconstruction from the top of the network to the bottom, one node at a time. However, such a local reconstruction approach is less accurate than the global one.

The normalized reconstruction error, computed on the T-dataset, is then defined as

$$e_{\mathrm{rec}} \stackrel{\mathrm{def}}{=} \frac{\sum_{n=1}^{N} ||(\mathbf{x_n} - \hat{\mathbf{x}_n})||^2}{\sum_{n=1}^{N} ||(\mathbf{x_n} - \bar{\mathbf{x}})||^2}, \tag{15}$$

which is the ratio between the energy of the reconstruction error and the variance of the test data except for a factor $N/(N-1)$.

Reconstruction errors of HGSFA, HiGSFA and PCA using 75 features are given in Table 9. The constant reconstruction $\bar{\mathbf{x}}$ (chance level) is the baseline with an error of 1.0. As expected, HGSFA does slightly better than chance level, but worse than HiGSFA, which is closer to PCA. PCA yields the best possible features for the given linear global reconstruction method, and is better than HiGSFA by 0.127. For HiGSFA, from the 75 output features, 8 of them

Table 10 Performance of HiGSFA on the MORPH-II database using different Δ_T (default value is $\Delta_T = 1.96$)

Δ_T	Age MAE	Age RMSE	e_{rec}	Race CR	Gender CR	#Features L3
1.92	3.506	4.583	0.330	99.15	97.58	2.87
1.94	3.499	4.583	0.333	99.15	97.64	3.22
1.96	3.497	4.583	0.338	99.15	97.70	3.66
1.98	3.530	4.637	0.359	99.09	94.72	4.14

The reported results are the age estimation errors (MAE and RMSE), the reconstruction error (e_{rec}), the percentual classification rate for race and gender, and the average length of the slow part in the features computed by the nodes of the third HiGSFA layer. All metrics have been computed on test data.

are slow features (slow part), and the remaining 67 are reconstructive. If one uses 67 features instead of 75, PCA yields a reconstruction error of 0.211, which is still better because the PCA features are computed globally.

HiGSFA network with HGSFA hyperparameters We verify that the performance of HiGSFA for age estimation is better than that of HGSFA not simply due to different hyperparameters by evaluating the performance of an HiGSFA network using the hyperparameters of the HGSFA network (the only difference is the use of iGSFA nodes instead of GSFA nodes). The hyperparameter Δ_T, not present in HGSFA, is set as in the tuned HiGSFA network. As expected, the change of hyperparameters affected the performance of HiGSFA: The MAE increased to 3.72 years, and the RMSE increased to 4.80 years. Although the suboptimal hyperparameters increased the estimation errors of HiGSFA, it was still clearly superior to HGSFA.

Sensitivity to the delta threshold Δ_T The influence of Δ_T on estimation accuracy and numerical stability is evaluated by testing different values of Δ_T. For simplicity, the same Δ_T is used from layers 3 to 10 in this experiment (Δ_T is not used in layers 1 and 2, where the number of features in the slow part is constant and equal to 3 and 4 features, respectively). The performance of the algorithm as a function of Δ_T is shown in Table 10. The Δ_T yielding minimum MAE and used in the optimized architecture is 1.96.

The average number of slow features in the third layer changes moderately depending on the value of Δ_T, ranging from 2.87 to 4.14 features, and the final metrics change only slightly. This shows that the parameter Δ_T is not critical and can be tuned easily.

Evaluation on the FG-NET database The FG-NET database (Cootes 2004) is a small database with 1002 facial images taken under uncontrolled conditions (e.g., many are not frontal) and includes identity and gender annotations. Due to its small size, it is unsuitable to evaluate HiGSFA directly. However, FG-NET is used here to investigate the capability of HiGSFA to generalize to a different *test* database. The HiGSFA ($\mathcal{G}_{3\text{L}}$) network that has been trained with images of the MORPH-II database (either with the set S_1 or S_2) is tested using images of the FG-NET database. For this experiment, images outside the original age range from 16 to 77 years are excluded.

For age estimation, the MAE is 7.32 ± 0.08 years and the RMSE is 9.51 ± 0.13 years (using 4 features for the supervised step). For gender and race estimation, the classification rates (5 features) are $80.85\% \pm 0.95\%$ and $89.24\% \pm 1.06\%$, resp. The database does not include race annotations, but all inspected subjects appear to be closer to white than to black. Thus, we assumed that all test persons have white race.

The most comparable cross-database experiment known to us is a system (Ni et al. 2011) trained on a large database of images from the internet and tested on FG-NET. By restricting the ages to the same 16–77 year range used above, their system achieves an MAE of approximately 8.29 years.

6 Discussion

In this article, we propose the use of *information preservation* to enhance HSFA and HGSFA. The resulting algorithms are denoted by HiSFA and HiGSFA, respectively, and significantly improve global slowness, input reconstruction and in supervised learning settings also label estimation accuracy.

We have analyzed the advantages and limitations of HSFA and HGSFA networks, particularly the phenomena of unnecessary information loss and poor input reconstruction. Unnecessary information loss occurs when a node in the network prematurely discards information that would have been useful for slowness maximization in another node higher up in the hierarchy. Poor input reconstruction refers to the difficulty of approximating an input accurately from its feature representation. We show unnecessary information loss is a consequence of optimizing slowness locally, yielding suboptimal global features.

HiSFA and HiGSFA improve the extracted features to address these shortcomings. In the conclusions below we focus on HiGSFA for simplicity, although most of them also apply to HiSFA. The feature vectors computed by iGSFA nodes in an HiGSFA network have two parts: a slow and a reconstructive part. The features of the slow part follow a slowness optimization goal and are slow features (in the sense of SFA) transformed by a linear scaling. The features of the reconstructive part follow the principle of information preservation (i.e. maximization of mutual information between the output and input data), which we implement in practice as the minimization of a reconstruction error. A parameter Δ_T (Δ-threshold) balances the lengths of the slow and reconstructive parts, consisting of J and $D - J$ features, respectively, where D is the output dimensionality and J is the number of slow features selected having $\Delta < \Delta_T$.

Parameter Δ_T thus controls the composition of the feature vector. A small Δ_T results in more reconstructive features and a large Δ_T results in more slow features. In particular, when $\Delta_T < 0$, iGSFA becomes equivalent to PCA, and when $\Delta_T \geq 4.0$, iGSFA becomes equivalent to GSFA except for a linear transformation (this assumes positive edge weights and a consistent graph). Theory justifies fixing Δ_T slightly smaller than 2.0 (Sect. 3.2), resulting in some features being similar to those of GSFA and other features being similar to those of PCA (on residual data).

Experiment 1 on unsupervised feature extraction from the view of a simulated rat shows that HiSFA yields significantly slower features than HSFA, and it is especially resistant to overfitting, allowing the extraction of slower features using much fewer training samples. Due to its improved feature slowness, HiSFA may be useful to improve simulations for neuroscience based on HSFA and SFA.

A method proposed by Escalante-B. and Wiskott (2016) for the combination of training graphs is used for the first time on real data to estimate either pose or subject (age/race/gender) attributes by combining pre-defined training graphs for the individual attributes into a single training graph, allowing efficient and accurate learning of multiple labels. In Experiment 2 on pose estimation, we experienced the problem that the HGSFA and HiGSFA networks concentrated too much on the easiest labels, yielding very few features that convey infor-

mation about the difficult labels. This was easily solved by increasing the weight of the graphs corresponding to less represented labels. For this experiment HiGSFA yielded better label estimation accuracy than HGSFA even though it was constrained to hyperparameters previously tuned for HGSFA.

Experiment 3 on age estimation (where the hyperparameters are unconstrained) shows that HiGSFA is superior to HGSFA in terms of feature slowness, input reconstruction and label estimation accuracy. Moreover, HiGSFA offers a competitive accuracy for age estimation, surpassing approaches based on bio-inspired features and some convolutional neural networks, such as a multi-scale CNNs (MCNN), which HiGSFA outperforms by 48.5 days ($\approx$ 1.5 months) mean absolute error. However, HiGSFA is not as accurate as current state-of-the-art CNNs for age estimation (Xing et al. 2017), which have a problem-specific structure. The improvement of HiGSFA over HGSFA is large: 154 days ($\approx$ 5 months). This is a significant technical and conceptual advance.

The next sections provide additional insights—mostly conceptual—into the proposed approach, the obtained results, and future work.

6.1 Remarks on the approach

We have shown that a single GSFA node in a network cannot always identify from its input which aspects or features contribute to the computation of *global* slow features. Hence, some of these features may be incorrectly discarded if the data dimensionality is reduced. Therefore, we resort in HiGSFA to a reconstruction goal to compress the local input and increase the chances that the local output features include information relevant to extract global slow features.

Besides the method used in HiGSFA to combine the slowness principle and information preservation, another method is to optimize a single objective function that integrates both criteria, favoring directions that are slow and have a large variance. However, we found in previous experiments that balancing these two criteria is difficult in practice. In addition, splitting the two types of features allows to keep SFA nonlinear and PCA linear.

HiGSFA should not be seen merely as a more sample efficient version of HGSFA, because it also provides a better feature representation and yields slower features. Even assuming unlimited training data and computational resources, features extracted by HGSFA would not necessarily reach the slowness achieved by HiGSFA. This hypothetical scenario is free of overfitting, but information loss would only decrease partially in HGSFA, because the main cause of this problem is not overfitting but blind optimization of slowness locally. One can observe this phenomenon in Table 1 (40 k), where HSFA does not reach the feature slowness of HiSFA even though overfitting is minimal.

6.2 Network hyperparameters and training times

By selecting the training graph and network structure appropriately, the computational complexity of HiGSFA (and other hierarchical versions of SFA) is linear w.r.t. the number of samples and their dimensionality, resulting in feasible training times, see "Appendix A". Training a single HiGSFA network (231,660 images of 96×96 pixels) takes only 10 hours, whereas HiSFA takes about 6 hours (including the time needed for data loading, the supervised step, training, and testing) on a single computer (24 virtual cores Xeon-X7542 @ 2.67GHz and 128 GB of RAM) without GPU computing. However, the algorithm could also benefit from GPUs or distributed computing, because nodes within the same layer can be

trained independently. For comparison, the system of Guo and Mu (2014) takes 24.5 hours (training and testing) for fewer images and lower resolution. Training times of the various CNNs were not included in the publications.

HiGSFA is more accurate than HGSFA even when using output dimensionalities and network hyperparameters that have been tuned for the latter network. However, HiGSFA can yield even higher accuracies if one uses larger output dimensionalities. This can be explained by various factors: (a) In both networks, the input dimensionality of a single GSFA node is I' (the expanded dimension). In HiGSFA the input dimensionality of a single PCA node is I, where typically $I \ll I'$. Hence, the reconstructive features of HiGSFA may overfit less than the slow features of HGSFA that they replace. (b) In HGSFA, the features of all the nodes are attracted to the same optimal free responses, whereas in HiGSFA the reconstructive part is attracted to the local principal components, which are different for each node. Thus, HGSFA layers might potentiate overfitting more than HiGSFA layers. (c) HGSFA networks may benefit less from additional features computed by the nodes, because additional quickly varying features are frequently noise-like and cause more overfitting in later nodes without improving slowness or reconstruction significantly.

In order to train iSFA and iGSFA, one must choose the Δ_T hyperparameter. A too small value $\Delta_T \approx 0.0$ results in only PCA features, whereas a too large value $\Delta_T \approx 4.0$ results in only slow features. The recommendation is to start with a value slightly smaller than 2.0. One can then tune this hyperparameter to improve the target metric. The control experiment in Sect. 5.3.4 shows that this tuning is straightforward. Alternatively, one can directly specify the number of slow features J'.

6.3 Remarks on experiment 3

Age estimation from adult facial photographs appears to be an ideal problem to test the capabilities of HiGSFA. PCA is not very useful for this problem because PCs represent wrinkles, skin texture and other higher-frequency features poorly. Therefore, it is counter-intuitive that feature slowness can be improved by incorporating PCs in HiGSFA. Improvements on feature slowness on other supervised learning problems (such as digit and traffic sign recognition, and the estimation of gender, race, and face pose from images) are less conclusive, because for such problems a few PCs encode discriminative information relatively well.

To estimate age, race, and gender simultaneously, we construct a graph $\mathcal{G}_{3\mathrm{L}}$ that combines three pre-defined graphs, encoding sensitivity to the particular labels, and favoring invariance to any other factor. Out of the 75 features extracted in the top-most node, 8 are slow and 67 are reconstructive. The best number of features passed to the supervised step ranges from 4 to 7 slow features, and the reconstructive part is not used. This shows that HiGSFA and HGSFA concentrate the label information in the first features. One can actually replace the iGSFA node on the top of the HiGSFA network by a regular GSFA node, so that all features are slow, without affecting the performance. The superiority in age estimation of HiGSFA over HGSFA is thus not due to the use of principal components in the final supervised step but to the higher quality of the slow features.

The performance of HiGSFA for age estimation on the MORPH-II database is competitive with an MAE of 3.497 years (or 3.412 years for HiGSFA-mirroring). Previous state-of-the-art results include an MAE of 3.63 years using a multi-scale CNN (Yi et al. 2015) and 3.92 using BIF+rKCCA+SVM (Guo and Mu 2014). During the preparation of the manuscript, newer results have improved the state of the art to 3.48 years (Liu et al. 2017) and 2.96 years (Xing et al. 2017).

The main goal of this research is not to provide best performance on any particular application. Our specific goal is to improve HSFA and HGSFA. In this sense, the central claim is that the new extensions are better regarding feature slowness, input reconstruction and (in the case of HiGSFA) label estimation accuracy.

6.4 Input reconstruction from slow features

Experiment 3 confirms that PCA is more accurate than HiGSFA at reconstruction using the same number of features (here 75), which was expected because PCA features are optimal when reconstruction is linear. From the 75 features extracted by HiGSFA, 67 are reconstructive (8 fewer than in PCA) and they are computed hierarchically (locally), in contrast to the PCA features, which are global. Thus, it is encouraging that the gap between PCA and HiGSFA at input reconstruction is moderate. In turn, HiGSFA is much more accurate than HGSFA, because reconstruction is the secondary goal of HiGSFA, whereas HGSFA does not pursue reconstruction.

Since the HiGSFA network implements a nonlinear transformation, it is reasonable to employ nonlinear reconstruction algorithms. Nonlinear reconstruction can provide more accurate reconstructions in theory, but in our experience it is difficult to train such type of algorithms well enough to perform better on test data than the simpler global linear reconstruction algorithm. An algorithm for nonlinear reconstruction that minimizes the feature error e_{feat} has been described in Sect. 4.5. However, since the number of dimensions is reduced by the network, one can expect many samples with feature error $e_{\text{feat}} = 0$ (or e_{feat} minimal) that differ from any valid input sample fundamentally (i.e., do not belong to the face manifold). To correct this problem, one might need to consider the input distribution and limit the range of valid reconstructions.

Another option is to resort to generative adversarial networks (Goodfellow et al. 2014; Radford et al. 2015; Denton et al. 2015), which have been used in the context of CNNs to generate random inputs with excellent image quality. For HiGSFA networks, such a promising approach might also be applicable to do nonlinear reconstruction, if properly adapted.

6.5 Future work

Age and pose estimation accuracy may be improved by using more training samples and more complex hierarchical networks, for instance by increasing the overlap of the receptive fields and using more complex nonlinearities. For age estimation one could use true face-distortion methods instead of simple image transformations.

One key factor for the performance of MCNN and MRCNN is the use of receptive fields centered at specific facial points (e.g., see entries ‘MCNN no align’ and ‘MCNN’ in Table 7). For $\text{Net}_{\text{hybrid}}^{\text{VGG}}$ the use of a face normalization procedure and an image distortion method may also be relevant. These ideas could also be applied to HiGSFA, and might particularly boost generalization.

Another direction of research is to investigate a possible connection between information preservation in HiGSFA and other methods used in CNNs, particularly the use of shortcut connections in ResNets (He et al. 2016) and Dense Networks (Huang et al. 2017). These connections provide a new information channel that allows a layer access to the unmodified input of a previous layer. Although these methods are different and have been motivated by other ideas, it appears they may also be justified by the information preservation principle.

6.6 Conclusion

We believe it is possible to develop successful learning algorithms based on a few simple but strong learning principles and heuristics, and this is the approach that we try to pursue with HiGSFA. An algorithm that might be strong but cannot be understood and justified analytically would be of less interest to us.

HiGSFA follows two of the suggestions by Krüger et al. (2013) based on findings from neuroscience regarding the primate visual system useful for successful computer vision, namely, hierarchical processing and information-channel separation. The slow and reconstructive parts of the extracted features can be seen as two information channels: The first one encodes information representing the slow parameters, and the second one encodes information representing the remaining aspects of the input. The slow part can be further decomposed. For example, in Experiment 3 one can observe one or more features for each label; the 3rd slowest feature is mostly related to race, the 4th one to gender, and all the remaining features to age.

This work explores the potential of bottom-up training and shows that it can be surprisingly accurate, making this type of architectures attractive for further research by exploring new heuristics and extensions.

The proposed algorithm is general purpose (e.g., it does not know anything about face geometry), but it still provides competitive accuracy, at least for the age/race/gender estimation problem. The results show the improved versatility and robustness of the algorithm and make it a good candidate for many other problems of computer vision on high-dimensional data, particularly those lying at the intersection of image analysis, nonlinear feature extraction, and supervised learning.

A Complexity of a quadratic HSFA network

Although existing systems based on HSFA have resorted to hierarchical processing for efficiency reasons, apparently its actual asymptotic complexity has not yet been formally established. In this section, we compute the computational complexity of a concrete quadratic HSFA (QHSFA) network illustrated in Fig. 11. This network operates on data with a 1D structure (i.e., vectors), where all nodes of the network perform quadratic SFA (QSFA, a quadratic expansion followed by linear SFA), and has two important parameters L and k. Parameter L indicates the number of layers and k is used for various purposes and is assumed to be fixed. In the first layer, the nodes have a fan-in and stride of k input values. In the remaining layers, the nodes have a fan-in and stride of two nodes. Each node of this network reduces the dimensionality of the data from k input components to $k/2$ output components. We denote the total input dimensionality by I.

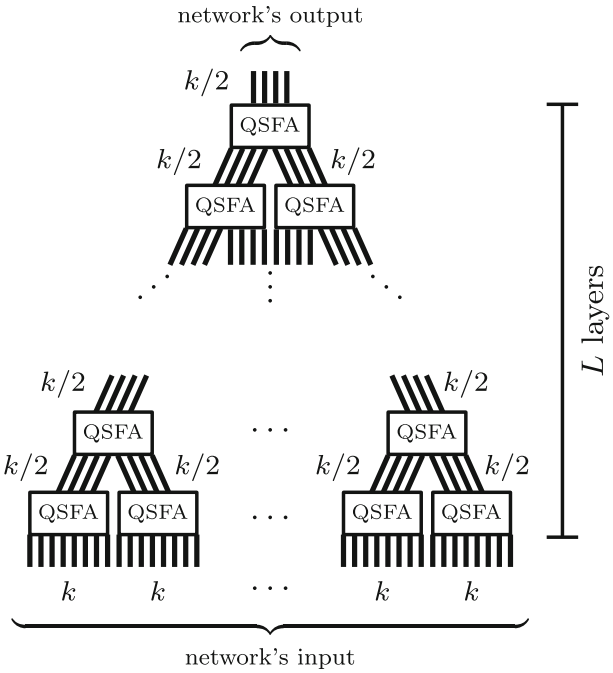

Fig. 11 Example of a 1D QHSFA network with binary fan-ins and no overlap. Each node performs quadratic SFA and reduces the dimensionality from k to $k/2$ components. The use of such a small fan-in results in a network with a large number L of layers, although L only grows logarithmically with the input dimension I

From the structure described above, it follows that the receptive fields of the nodes are non-overlapping, the network's output has $k/2$ components, the number of layers L is related to I and k:

$$I \;=\; k2^{L-1}, \tag{16}$$

and the total number of nodes in the network is

$$M \;=\; 2^L - 1. \tag{17}$$

Internally, all the QSFA nodes of the network have the same structure. The input data to a single node has k dimensions, which are increased by the quadratic expansion to $k(k+3)/2$ components. Afterwards, linear SFA reduces the dimensionality to $k/2$. Thus, the complexity of training a single (nonlinear) node is

$$T_{\mathrm{QSFA}}(N,k) \overset{(5)}{=} \mathcal{O}(N(k(k+3)/2)^2 + (k(k+3)/2)^3) \tag{18}$$

$$= \mathcal{O}(Nk^4 + k^6). \tag{19}$$

On the other hand, the number of nodes is $M \overset{(16,17)}{=} \mathcal{O}(I/k)$. Therefore, the complexity of training the whole network is

$$T_{\mathrm{QHSFA}}(N,I,k) \overset{(19)}{=} \mathcal{O}((Nk^4 + k^6)I/k) = \mathcal{O}(INk^3 + Ik^5)\,. \tag{20}$$

Thus, the complexity of the complete QHSFA network above is linear w.r.t. the input dimension I, whereas the complexity of direct QSFA (on I-dimensional data) is

$$T_{\mathrm{QSFA}}(N, I) \overset{(19)}{=} \mathcal{O}(NI^4 + I^6), \tag{21}$$

which is linear w.r.t. I^6. This shows that the QHSFA network is computationally much more efficient than direct QSFA (given that $k \ll I$).

Since each layer in the QHSFA network is quadratic, in general the output features of the l-th layer can be written as polynomials of degree 2^l on the input values. In particular, the output features of the whole network are polynomials of degree 2^L. However, the actual feature space spanned by the network does not include all polynomials of this degree but only a subset of them due to the restricted connectivity of the network. In contrast, direct QSFA only contains quadratic polynomials (although all of them).

One could try to train direct SFA on data expanded by a polynomial expansion of degree 2^L (to encompass the feature space of QHSFA), but the complexity would be prohibitive due to the large expanded dimensionality $\sum_{d=0}^{2^L} \binom{d+I-1}{I-1}$. Training SFA with such a high-dimensional data appears to be exponential in I.

We are also interested in analyzing the memory complexity of the QHSFA network. Memory (space) complexity is denoted by S. Thus, the memory complexity of linear SFA is

$$S_{\mathrm{SFA}}(N, I) = \mathcal{O}(NI + I^2)\,, \tag{22}$$

where the term NI is due to the input data, and I^2 is due to the covariance matrices. If a quadratic expansion is added, SFA becomes QSFA and the memory complexity becomes

$$S_{\mathrm{QSFA}}(N, I) = \mathcal{O}(NI + I^4)\,. \tag{23}$$

One can reduce these complexities by using HSFA/QHSFA and by training the nodes sequentially, one at a time, independently of whether an expansion has been applied or not. In particular, the memory complexity of the QHSFA network is only

$$S_{\mathrm{QHSFA}}(N, I, k) = \mathcal{O}(NI + k^4)\,. \tag{24}$$

The excellent computational and memory complexity of the QHSFA network is not exclusive to this simple architecture. It is possible to design more sophisticated networks and preserve a similar computational complexity. For example, training the HiGSFA network proposed in Sect. 5, which has overlapping receptive fields, larger fan-ins, and a 2D structure, has complexity also linear in I and N if one adds more pairs of layers with a fan-in of 1×3 and 3×1 to match the size of the input data as needed (concrete analysis not provided).

B Sensitivity-based scaling

As mentioned in Sect. 4.4, the QR scaling method (10)–(12) is useful to give the features in the slow part a meaningful scale, but it has the disadvantage that it mixes the slow features. This is irrelevant when one combines a polynomial expansion with SFA because the extracted features are invariant to invertible linear transformations of the input, including mixtures, i.e., $\mathrm{SFA}(\mathrm{QExp}(\mathbf{U}\mathbf{x})) \equiv \mathrm{SFA}(\mathrm{QExp}(\mathbf{x}))$, where $\mathbf{U}$ is any invertible matrix and QExp is the quadratic expansion). In other words, polynomial SFA can extract the same features from $\mathbf{s}'$ or $\mathbf{y}'$.

However, other expansions, including the 0.8EXP (13) expansion, do not have this property. When the 0.8EXP expansion is combined with SFA, the resulting algorithm is only invariant to scalings of the input components but not to their mixing, i.e., SFA(0.8EXP($\mathbf{\Lambda x}$)) $\equiv$ SFA(0.8EXP($\mathbf{x}$)), where $\mathbf{\Lambda}$ is a diagonal matrix with diagonal elements $\lambda_i \neq 0$, but SFA(0.8EXP($\mathbf{Ux}$)) $\not\equiv$ SFA(0.8EXP($\mathbf{x}$)) in general. Thus, using QR scaling would change the features extracted in later nodes fundamentally. Furthermore, the 0.8EXP expansion has been motivated by a model where the input slow features are noisy harmonics of increasing frequency of a hidden parameter and the expansion should be applied to these features directly. Thus, mixing the slow features would break the assumed model and might compromise slowness extraction in the next layers in practice.

Technically, feature mixing by QR scaling could be reverted in the next layer (e.g., by an additional application of linear SFA before the expansion), but such a step would add unnecessary complexity. Therefore, as an alternative to the QR scaling, we also propose a *sensitivity based scaling*, which scales the slow features without mixing them, as follows.

$$\mathbf{y}' = \mathbf{\Lambda s}' \,, \tag{25}$$

where $\mathbf{\Lambda}$ is a diagonal matrix with diagonal elements $\lambda_j \stackrel{\text{def}}{=} ||\mathbf{M}_j||_2$ (the L_2-norm of the j-th column vector of $\mathbf{M}$). Thus,

$$\mathbf{a}(t) \stackrel{(10,25)}{=} \mathbf{M\Lambda}^{-1}\mathbf{y}'(t) + \mathbf{d} \,. \tag{26}$$

Clearly, the transformation (25) does not mix the slow features, it only scales them. From the two key reconstruction properties of PCA mentioned in Sect. 4.4 (adding noise of certain variance to either one or many features increases the variance of the reconstruction error by the same amount), the first one (noise on a single feature) is fulfilled, because the columns of $\mathbf{M\Lambda}^{-1}$ have unit norm since $\mathbf{\Lambda}^{-1} = \text{Diag}(1/\lambda_1, \ldots, 1/\lambda_J)$. The second property is not fulfilled, because $\mathbf{M\Lambda}^{-1}$ is in general not orthogonal (in contrast to $\mathbf{Q}$).

At first glance, it seems like multi-view learning (data fusion, e.g., Xia et al. 2010) might be an alternative for the scaling methods used here. However, multi-view learning actually solves a different problem since we do not join two information channels but actually split them.

References

Berkes, P. (2005a). Handwritten digit recognition with nonlinear Fisher discriminant analysis. In *ICANN, LNCS* (Vol. 3697, pp. 285–287). Berlin: Springer.

Berkes, P. (2005b). Pattern recognition with Slow Feature Analysis. *Cognitive Sciences EPrint Archive (CogPrints)*. Retrieved February 24, 2012 from http://cogprints.org/4104/.

Berkes, P., & Wiskott, L. (2005). Slow Feature Analysis yields a rich repertoire of complex cell properties. *Journal of Vision*, *5*(6), 579–602.

Blaschke, T., Zito, T., & Wiskott, L. (2007). Independent Slow Feature Analysis and nonlinear blind source separation. *Neural Computation*, *19*(4), 994–1021.

Cootes, T. (2004). Face and gesture recognition research network (FG-NET) aging database. Retrieved June 17, 2009 from http://www-prima.inrialpes.fr/FGnet/.

Denton, E. L., Chintala, S., Szlam, A., & Fergus, R. (2015). Deep generative image models using a Laplacian pyramid of adversarial networks. *Advances in Neural Information Processing Systems*, *28*, 1486–1494.

Escalante-B, A. N., & Wiskott, L. (2010). 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* (pp. 240–249). Germany: Dortmund.

Escalante-B, A. N., & Wiskott, L. (2012). Slow Feature Analysis: Perspectives for technical applications of a versatile learning algorithm. *Künstliche Intelligenz [Artificial Intelligence]*, *26*(4), 341–348.

Escalante-B, A. N., & Wiskott, L. (2013). How to solve classification and regression problems on high-dimensional data with a supervised extension of Slow Feature Analysis. *Journal of Machine Learning Research*, *14*, 3683–3719.

Escalante-B, A. N., & Wiskott, L. (2016). Theoretical analysis of the optimal free responses of graph-based SFA for the design of training graphs. *Journal of Machine Learning Research*, *17*(157), 1–36.

Fink, M., Fergus, R., & Angelova, A. (2003). Caltech 10,000 web faces. Retrieved September 6, 2010 from http://www.vision.caltech.edu/Image_Datasets/Caltech_10K_WebFaces/.

Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, *7*, 179–188.

Földiák, P. (1991). Learning invariance from transformation sequences. *Neural Computation*, *3*(2), 194–200.

Franzius, M., Sprekeler, H., & Wiskott, L. (2007). Slowness and sparseness lead to place, head-direction, and spatial-view cells. *PLoS Computational Biology*, *3*(8), 1605–1622.

Franzius, M., Wilbert, N., & Wiskott, L. (2011). Invariant object recognition and pose estimation with slow feature analysis. *Neural Computation*, *23*(9), 2289–2323.

Gao, W., Cao, B., Shan, S., Chen, X., Zhou, D., Zhang, X., et al. (2008). The CAS-PEAL large-scale Chinese face database and baseline evaluations. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, *38*(1), 149–161.

Geng, X., Zhou, Z. H., & Smith-Miles, K. (2007). Automatic age estimation based on facial aging patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*(12), 2234–2240.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., et al. (2014). Generative adversarial nets. *Advances in Neural Information Processing Systems*, *27*, 2672–2680.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. Cambridge: MIT Press.

Guo, G., & Mu, G. (2010). Human age estimation: What is the influence across race and gender? In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (pp 71–78).

Guo, G., & Mu, G. (2011). Simultaneous dimensionality reduction and human age estimation via kernel partial least squares regression. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition* (pp 657–664).

Guo, G., & Mu, G. (2014). A framework for joint estimation of age, gender and ethnicity on a large database. *Image and Vision Computing*, *32*(10), 761–770.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016* (pp 770–778).

Hinton, G. E. (1989). Connectionist learning procedures. *Artificial Intelligence*, *40*(1–3), 185–234.

Huang, G., Liu, Z., van der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*

Huang, G. B., Ramesh, M., Berg, T., & Learned-Miller, E. (2007). Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Tech. Rep. 07-49, University of Massachusetts, Amherst

Klampfl, S., & Maass, W. (2010). Replacing supervised classification learning by Slow Feature Analysis in spiking neural networks. In *Proc. of NIPS 2009: Advances in Neural Information Processing Systems* (Vol. 22, pp. 988–996), MIT Press.

Koch, P., Konen, W., & Hein, K. (2010). Gesture recognition on few training data using slow feature analysis and parametric bootstrap. In *International Joint Conference on Neural Networks* (pp. 1–8).

Kompella, V. R., Luciw, M. D., & Schmidhuber, J. (2012). Incremental slow feature analysis: Adaptive low-complexity slow feature updating from high-dimensional input streams. *Neural Computation*, *24*(11), 2994–3024.

Krüger, N., Janssen, P., Kalkan, S., Lappe, M., Leonardis, A., Piater, J., et al. (2013). Deep hierarchies in the primate visual cortex: What can we learn for computer vision? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *35*(8), 1847–1871.

Kuhnl, T., Kummert, F., & Fritsch, J. (2011). Monocular road segmentation using slow feature analysis. In *Intelligent Vehicles Symposium* (pp. 800–806), IEEE

Kumar, N., Belhumeur, P. N., & Nayar, S. K. (2008). FaceTracer: A search engine for large collections of images with faces. In *European Conference on Computer Vision (ECCV)* (pp 340–353).

Liu, H., Lu, J., Feng, J., & Zhou, J. (2017). Group-aware deep feature learning for facial age estimation. *Pattern Recognition*, *66*, 82–94.

Mitchison, G. (1991). Removing time variation with the anti-Hebbian differential synapse. *Neural Computation*, *3*(3), 312–320.

Mohamed, N. M., & Mahdi, H. (2010). A simple evaluation of face detection algorithms using unpublished static images. In *10th International Conference on Intelligent Systems Design and Applications* (pp. 1–5).

Ni, B., Song, Z., & Yan, S. (2011). Web image and video mining towards universal and robust age estimator. *IEEE Transactions on Multimedia*, *13*(6), 1217–1229.
Phillips, P. J., Flynn, P. J., Scruggs, T., Bowyer, K. W., Chang, J., Hoffman, K., Marques, J., Min, J., & Worek, W. (2005). Overview of the face recognition grand challenge. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition, IEEE Computer Society* (Vol. 1, pp. 947–954)
Radford, A., Metz, L., & Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. E-print arXiv:1511.06434
Ricanek Jr, K., & Tesafaye, T. (2006). Morph: A longitudinal image database of normal adult age-progression. In *Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition*, IEEE Computer Society, FGR '06 (pp. 341–345)
Rish, I., Grabarnik, G., Cecchi, G., Pereira, F., & Gordon, G. J. (2008). Closed-form supervised dimensionality reduction with generalized linear models. In *Proc. of the 25th ICML* (pp. 832–839), ACM.
Schönfeld, F., & Wiskott, L. (2013). Ratlab: An easy to use tool for place code simulations. *Frontiers in Computational Neuroscience*, *7*, 104. https://doi.org/10.3389/fncom.2013.00104.
Sprekeler, H. (2011). On the relation of slow feature analysis and Laplacian eigenmaps. *Neural Computation*, *23*(12), 3287–3302.
Sprekeler, H., Michaelis, C., & Wiskott, L. (2007). Slowness: An objective for spike-timing-dependent plasticity? *PLoS Computational Biology*, *3*(6), 1136–1148.
Sprekeler, H., Zito, T., & Wiskott, L. (2014). An extension of Slow Feature Analysis for nonlinear blind source separation. *Journal of Machine Learning Research*, *15*, 921–947.
Sugiyama, M. (2006). Local Fisher discriminant analysis for supervised dimensionality reduction. In *Proc. of the 23rd ICML* (pp. 905–912).
Sugiyama, M., Idé, T., Nakajima, S., & Sese, J. (2010). Semi-supervised local Fisher discriminant analysis for dimensionality reduction. *Machine Learning*, *78*(1–2), 35–61.
Tang, W., & Zhong, S. (2007). Computational methods of feature selection, Chapman and Hall/CRC, chap pairwise constraints-guided dimensionality reduction
Wilbert, N. (2012). Hierarchical slow feature analysis on visual stimuli and top-down reconstruction. PhD thesis, Humboldt-Universität zu Berlin, Mathematisch-Naturwissenschaftliche Fakultät I
Wiskott, L. (1998). Learning invariance manifolds. In *Proc. of 5th Joint Symposium on Neural Computation*, San Diego, CA, USA, Univ. of California (Vol. 8, pp. 196–203).
Wiskott, L., & Sejnowski, T. (2002). Slow feature analysis: Unsupervised learning of invariances. *Neural Computation*, *14*(4), 715–770.
Xia, T., Tao, D., Mei, T., & Zhang, Y. (2010). Multiview spectral embedding. *Transactions on Systems, Man, and Cybernetics, Part B*, *40*(6), 1438–1446.
Xing, J., Li, K., Hu, W., Yuan, C., & Ling, H. (2017). Diagnosing deep learning models for high accuracy age estimation from a single image. *Pattern Recognition*, *66*, 106–116.
Yi, D., Lei, Z., & Li, S. (2015). Age estimation by multi-scale convolutional network. In *Computer Vision—ACCV 2014, Lecture Notes in Computer Science* (Vol. 9005, pp. 144–158).
Zhang, D., Zhou, Z. H., & Chen, S. (2007) Semi-supervised dimensionality reduction. In *Proc. of the 7th SIAM International Conference on Data Mining*
Zhang, Z., & Tao, D. (2012). Slow feature analysis for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(3), 436–450.
Zito, T., Wilbert, N., Wiskott, L., & Berkes, P. (2009). Modular toolkit for data processing (MDP): A python data processing framework. *Frontiers in Neuroinformatics*, *2*, 8.