Gesture Recognition on Few Training Data using Slow Feature Analysis and Parametric Bootstrap

Patrick Koch, Wolfgang Konen, and Kristine Hein

Abstract— **Slow Feature Analysis (SFA) has been established as a robust and versatile technique from the neurosciences to learn slowly varying functions from quickly changing signals. Recently, the method has been also applied to classification tasks. Here we apply SFA for the first time to a time series classification problem originating from gesture recognition. The gestures used in our experiments are based on acceleration signals of the Bluetooth Wiimote controller (Nintendo). We show that SFA achieves results comparable to the well-known Random Forest predictor in shorter computation time, given a sufficient number of training patterns. However – and this is a novelty to SFA classification – we discovered that SFA requires the number of training patterns to be strictly greater than the dimension of the nonlinear function space. If too few patterns are available, we find that the model constructed by SFA severely overfits and leads to high test set errors. We analyze the reasons for overfitting and present a new solution based on parametric bootstrap to overcome this problem.**

I. Introduction

Slow Feature Analysis (SFA) is a new learning algorithm emerging from neuroscience which is capable of learning unsupervised new features or 'concepts' from multidimensional time signals. SFA has been originally developed in context of an abstract model of unsupervised learning of invariances in the visual system of vertebrates [Wis98] and is described in detail in [WS02], [Wis03]. Although SFA is inspired from neuroscience, it does not have the drawbacks of conventional ANNs such as long training times or strong dependencies on initial conditions. Instead, SFA is fast in training and it has the potential to find hidden features out of multidimensional signals, as has been shown impressively by [Ber05] for handwritten-digit recognition. For these reasons we apply SFA for the first time to a complex time series classification problem originating from gesture recognition, namely acceleration signals of the Nintendo Bluetooth Wiimote controller and observe several interesting facts when applying this method to small data sets.

The inclusion of accelerometers in electronic devices such as mobile phones or game controllers for gesture recognition has become more and more popular in consumer electronics. While in the nineties, the recognition of gestures placed a high demand on hardware and was only present in the labs of research institutes, gesture recognition has now made its way into the homes. With over 50 million units sold [Shi09] the reason for the success of the Nintendo Wii console can be connected to the novel type of controller handling, which clearly differs from traditional game controllers. But even though there exist a large number of games for the Nintendo Wii, recognition of complex gestures is still a challenging task. In spite of the large number of classification approaches using the Wiimote controller with the infrared device, e.g. the work from Lee [Lee08], we focus here on approaches where no infrared device is used. Although the task becomes more difficult then, we are not so dependent on the Wii structure and can easily transfer our approach to other applications. This can be of high importance when the use of infrared is not possible for any reason and only acceleration based sensors are available. Similar works exist, e.g., a recent study by Schlömer *et al.* [SPHB08], who present a classification tool based on a Hidden Markov Chain approach and Liu *et al.* [LW$^+$09] who apply personalized gesture recognition to a user authentication problem. Rehm *et al.* [RBA08] describe classification of Wii gestures with different methods and expose the influence of cultural behaviours in gestures. Malmestig and Sundberg [MS08] use the Wiimote to recognize gestures for sign language recognition with good results. The commercial product LiveMove2 [Inc09] integrates features of the Nintendo Motion Plus addon but no benchmark results are available for publicity.

The Slow Feature Analysis and Random Forest algorithms are briefly introduced in the next section. We also give detailed information about the acquisition and preparation of the gathered data. The experimental results are described in Sec. 3 and discussed in Sec. 4. We finish with a conclusion of the results and give an outlook of our future work in Sec. 5.

II. Methods

A. Slow Feature Analysis

Slow Feature Analysis has been invented by Wiskott and Sejnowski [WS02], [Wis03] and later extended by Berkes [Ber05] for classification. The original SFA approach for time series analysis is defined as follows: For a multivariate time series signal $\vec{x}(t)$ where t indicates time, find the set of real-valued output functions $g_1(\vec{x}), g_2(\vec{x}), ..., g_M(\vec{x})$, such that each output function

$$y_j(t) = g_j(\vec{x}(t)) \tag{1}$$

minimally changes in time[1]:

$$\Delta y_j(t) = \langle \dot{y_j}^2 \rangle_t \text{ is minimal} \tag{2}$$

To exclude trivial solutions we add some constraints:

$$\langle y_j \rangle_t = 0 \text{ (zero mean)} \tag{3}$$

Patrick Koch, Wolfgang Konen, and Kristine Hein are with the Department of Computer Sciences and Engineering Science, Cologne University of Applied Sciences, 51643 Gummersbach, Germany (email: {patrick.koch, wolfgang.konen, kristine.hein}@fh-koeln.de)

[1] $\langle \cdot \rangle_t$ means average over time and $\dot{y}$ indicates the time derivative.

$$\langle y_j^2 \rangle_t = 1 \text{ (unit variance)} \tag{4}$$

$$\langle y_k y_j \rangle_t = 0 \text{ (decorrelation for } k > j) \tag{5}$$

The third equation is only relevant from the second slow signal on and higher ones to prevent higher signals from learning features already represented by slower signals.

For arbitrary functions this problem is difficult to solve, but SFA finds a solution by expanding the input signal into a nonlinear function space by applying certain basis functions, e.g. monomials of degree d. This expanded signal is sphered to fulfill the constraints of equations (3), (4) and (5). Then SFA calculates the time derivative of the sphered expanded signal and determines from its covariance matrix the normalized eigenvector with the smallest eigenvalue. Finally the sphered expanded signal is projected onto this eigenvector to obtain the slowest output signal $y_1(t)$.

Berkes [Ber05] has extended this approach to classify a set of handwritten digits. The main idea of this extension is to create many small time series out of the class patterns. Let us assume that for a K-class problem each class $c_m \in \{c_1, ..., c_K\}$ has got N_m patterns. We then reformulate the Δ-objective function (2) for SFA with distinct indices k and l as the mean of the difference over all possible pairs:

$$\Delta(y_j) = \frac{1}{n_{pair}} \cdot \sum_{m=1}^{K} \sum_{\substack{k,l=1,\\ k<l}}^{N_m} \left(g_j(p_k^{(m)}) - g_j(p_l^{(m)}) \right)^2 \tag{6}$$

where n_{pair} denotes the total count of all pairs and $p_k^{(m)}$ and $p_l^{(m)}$ represent the k-th and l-th class pattern of class m. In our case such a pattern can be understood as a gesture, consisting of a concatenated signal from the 3 axis accelerometer of the Wiimote controller. Constraints (3), (4) and (5) can be reformulated then by substituting the average over time with the average over all patterns, such that the learned functions are going to have zero mean, unit variance and be decorrelated [Ber05].

As [Ber05] has shown, the $(K-1)$ slowest SFA output signals are expected to have a low intra-class variation, but usually a high inter-class variation. Therefore it is natural to train a standard Gaussian classifier on the slowest $(K-1)$ SFA outputs produced from the training records. The Gaussian classifier will seek an optimal position and shape of a Gauss function for each class in this $(K-1)$-dimensional space.

All SFA calculations were performed with the extended MATLAB toolkit `sfa-tk` V2.0 from [Kon09], [Ber05].

B. Random Forest

The Random Forest (RF) predictor by Breiman [Bre01] is an extension of the well-known classification and regression trees (CART) [BFSO84]. The method is a combination of several tree predictors and hence is comparable to other ensemble methods as bagging or boosting. We use here the R-implementation of Random Forest [LW02]. It has been shown in the past that the algorithm is quite robust to overfitting and noise, that is why we use it as a comparison to the SFA approach.

TABLE I

ABSOLUTE EXECUTION TIMES (IN SECONDS) FOR FIVE EXAMPLE GESTURES PERFORMED BY TEN TEST PERSONS. EACH GESTURE WAS RECORDED TEN TIMES.

Gesture	Minimum	Median	Maximum	Std.Dev.
Circle	0.773	1.607	4.914	0.343
Throw	0.416	0.957	2.306	0.471
Frisbee	0.376	0.883	2.036	0.353
Bowling	0.530	1.325	2.303	0.377
Z	0.980	1.607	4.280	0.453

Fig. 1. Wiimote Controller

C. Data Acquisition and Data Preparation

Nintendo has invented the Wiimote Controller (Fig. 1) as a Bluetooth compliant game controller for the Nintendo Wii console. In its basic version an acceleration sensor, a so called accelerometer, is implemented inside the device, as well as one infrared device for position determination of the controller during games. Additionally there exists an extension to the Wiimote, the Wii Motion Plus which contains two gyrometers for calculation of angular velocities. In order to keep things as simple as possible with respect to the sensor and hardware side we neither use the sensor information from the Wii infrared sensor (sensor bar) nor from the Wii Motion Plus sensors. As observed in other studies by Schlömer *et al.* [SPHB08] and Rehm *et al.* [RBA08] we expect to get comparably good and accurate recognition rates with the sensor data solely taken from the accelerometer.

We recorded five different gestures from ten test persons with the Avetana Bluetooth library together with a modified implementation of the Wiigee framework by Poppinga [Pop07b]. Accelerometer data were recorded with their timestamps at a rate of approximately 100 Hz, not equidistantly delivered from the Bluetooth interface. In a preliminary data preparation step we interpolated the accelerometer data at 300 equidistant time points between the first and the last timestamp. A first view on the data revealed that the patterns usually vary strongly in execution time and amplitude; even execution times from the same person seem to be rather volatile. In Tab. I we present a summary of the execution times of a single gesture set, resulting from the ten test persons. Due to the large variance we conduct a smoothing of the gestures by taking the mean of each 10 consecutive data points, resulting in a set of $n = 30$ points for each accelerometer dimension $x_{acc}(t)$, $y_{acc}(t)$ and $z_{acc}(t)$. Another preliminary observation was that the gestures from different persons usually vary in the size of the amplitude. Hence, as an additional operator we use an amplitude normalization by dividing the accelerometer data of each gesture by its standard deviation. In Fig. 2 the difference between the gesture signals before and after data preparation can be seen. It is clearly visible that despite the normalization steps taken, there is still considerable variation within the same gesture type from the same person. Although there is only a small difference between the time-normalized gestures in the middle column and the gestures with amplitude normalization in the right column, the results became slightly better with amplitude normalization for all classifiers.

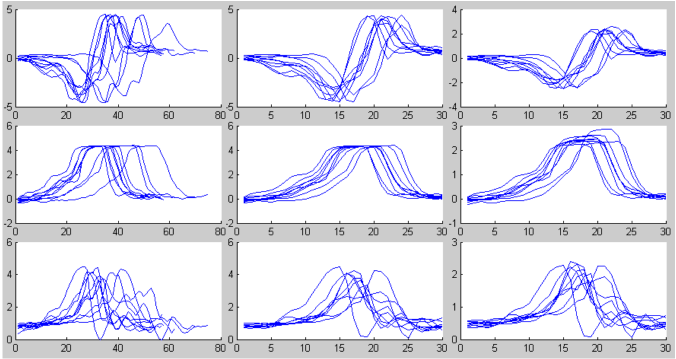


Fig. 2. Sensor data for all frisbee gesture patterns from one person. *Left column:* x_{acc}, y_{acc} and z_{acc} acceleration sensor values before time- and amplitude-normalization. *Middle column:* Sensor values after time normalization. *Right column:* Final curves for classification after amplitude normalization.

In order to produce one vector for each gesture we concatenate the multivariate time series data from the three acceleration sensors $x_{acc}(t)$, $y_{acc}(t)$, $z_{acc}(t)$ into a single pattern $\vec{X}(t) = (x_{acc}(t) \oplus y_{acc}(t) \oplus z_{acc}(t))$, where $\oplus$ denotes concatenation. $\vec{X}(t)$ has the dimension $3n = 90$. We concatenate the raw gesture execution time in seconds as the last dimension, making $\vec{X}(t)$ finally a $3n + 1 = 91$-dimensional vector.

Since 91 dimensions as input to SFA lead to very large processing times and memory requirements, we reduce the dimensions as a final data preparation step by PCA to the n_{pp} dimensions which carry most of the variance in the training data. Usually we let n_{pp} vary between 3 and 20, since higher values are computationally expensive and lower values have a negative impact on the classification. In the case of RF we do not need this dimension reduction.

III. Experimental Results

Experimental Setup

The gestures recorded for recognition were performed by each person ten times. The participants had to push and hold a button on the Wiimote while performing the gestures. The recorded gestures were passed through the data preparation steps as described in Sec. II-C. These preprocessed patterns were passed into the SFA and Random Forest classifiers. The gesture set used in our work is thought to be almost realistic for the application in games, since a set of five common gestures is used:

- Circle,
- Throw,
- Bowling,
- Frisbee,
- Z (the letter 'Z' painted in the air)

In future work we plan to extend this choice to a more complex gesture set in order to find out how well similar looking gestures can be classified. The acceleration values for several patterns of the frisbee gesture are exemplarily plotted in Fig. 2.

Both SFA and RF need some number of training and test data in advance. Since classification results usually depend highly on the chosen division of training and test data we differ in the following between different recognition tasks with respect to test and training sets:

A. *Random Sampling:* Classification on the recorded gestures with random sampled test and training set. 10-fold cross validation is used to certify the gathered results.
B. *Recognition of unseen persons:* Partitioning of training and test set by leaving all gestures from a certain person out of the training set. This partitioning is done for all persons sequentially.
C. *Small number of patterns:* Recognition of gestures, when only a marginal number of training patterns is available.

A. Random Sampling

In Fig. 3 and Tab. II we show results of 10-fold cross validation (CV) from ten independent runs on randomly sampled data. The 716 gestures were divided into 10 folds containing 71 or 72 records. Each fold in turn was considered as test data while the remaining data were training data. The test error rate on unseen gestures decreases quickly to values around 2% as the PCA-reduced dimension reaches $n_{pp} = 12$ or above. As a comparision we directly trained a Gaussian classifier on the same preprocessed input of dimension n_{pp} and tested its performance on the same unseen test data. For $n_{pp} \geq 12$ the pure Gaussian classifier is worse by a factor of 6, showing the strength of the feature combinations found unsupervisedly by SFA. We also show in Fig. 3 and Tab. II the results of RF (blue dash-dotted line). RF is slightly better for n_{pp} =12, but for n_{pp} =15 and above SFA and RF are very similar.

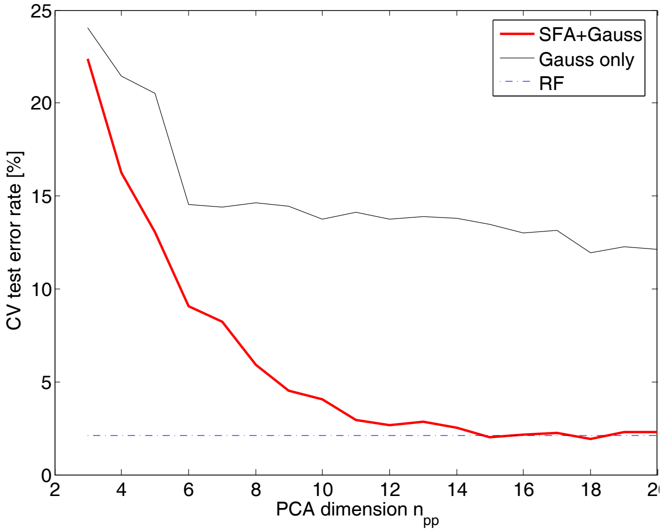


Fig. 3. Error rates achieved with SFA for different preprocessed dimensions n_{pp} used after PCA preprocessing. Shown are the averages out of 10 runs with different seeds for the CV fold generation. We show for comparision the mean RF error rate 2.09% from Tab. II (independent of n_{pp}).

TABLE II
ERROR RATES OBTAINED FROM TEN RUNS OF 10-FOLD CROSS VALIDATION ON RANDOM SAMPLED GESTURE DATA.

Classifier	Best	Mean	Worst	Std.Dev.
SFA(n_{pp} =12)	2.37	2.82	3.21	0.25
SFA(n_{pp} =15)	1.68	**2.03**	**2.24**	0.18
RF	**1.54**	2.09	2.37	0.30
Gauss	13.55	14.02	14.39	0.22

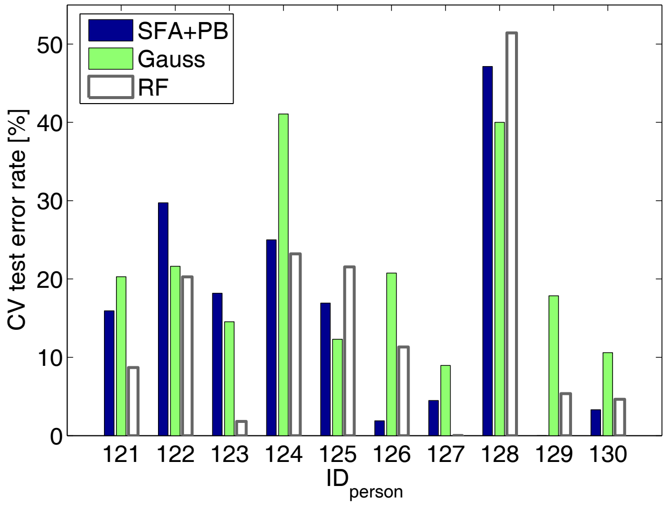


Fig. 4. Error rates when classifying gestures of persons unseen in the training set (SFA+PB see Sec. IV, Gauss: Gaussian Classifier, RF: Random Forest).

B. Classification of Gestures from New Persons

In classification with random sampling of training and test sets it is indirectly presumed that the training data is representative for the test set. However, in gesture recognition it is of high importance that gestures of persons who never occured in the training data can also be recognized by the classifier. Rehm *et al.* [RBA08] stressed that gestures are influenced by the expressivity of the user. Factors for expressivity are e.g. speed, space used for the gesture and the cultural background. Due to the large differences between several persons this may be a quite difficult task for any classifier. Nevertheless this experiment can be of high relevance for the game industry, because when a good classification is possible, no person-specific calibration will be needed.

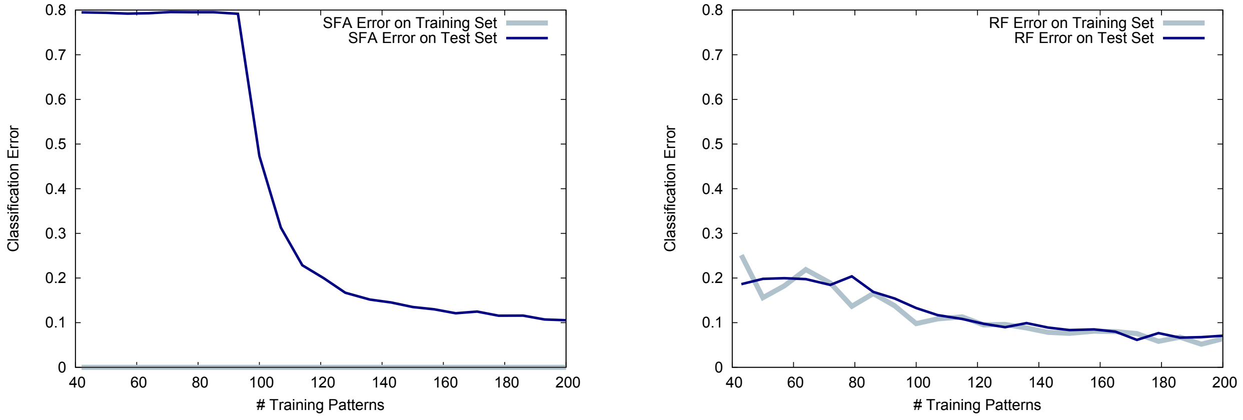
In Fig. 4 we show the results of a cross validation experiment for SFA with parametric bootstrap (SFA+PB, see Sec. IV), Gauss-Classifier and RF when the gestures of each person in turn are put completely into the test set. As expected, the error rates are considerably higher than with random sampling. Tab. III shows the error rate when averaging over all gestures which are $\{15.3,\ 14.8,\ 19.6\}\%$ for {SFA, RF, Gauss}, resp. But the most striking feature of Fig. 4 is the great variety of error rates between persons. Some persons, e.g. ID 127, 129, 130, are extremely well classifiable for SFA and RF, while others, most prominently the person with ID 128, are hard to classify for all algorithms. Presumably this person has very different characteristics when performing gestures, but further research is necessary to analyze this more quantitatively.

C. Marginal Training Data

Sometimes data is rare, e.g. for classification with few observations. In our case there were only ten patterns per person available, which makes training more difficult for function learning classifiers as SFA or ANNs. The generalization ability for such methods usually increases with more training examples. Surprisingly we encountered a SFA specific problem, namely when only a small number of training data is available for classification and this number is even smaller than the defined expanded function space denoted by D_{xp}.

To clarify this issue we applied SFA and RF as reference methods in five runs with different random seeds to the same set of gestures. The ratio of training and test set was varied starting with a high number of training data and low number of test data and then sequentially decreasing the training data and increasing the test data respectively. The final classification results for this experiment can be seen in Fig. 5(a) for SFA and in Fig. 5(b) for RF.

Both plots show that the classification rates of the two classifiers are promising with enough training data available (e.g. $>$ 120 training patterns). However, with less training patterns especially the SFA detection rates become quite unsatisfying. The error rate increases immensely for $<$ 90 training patterns, while the error on the training set constantly stays at zero level. The random forest predictor is not so largely affected by this problem, because the subspaces determined during classification should still be correct, though the results are not so accurate any longer. But as a peculiarity of the RF classifier the error rates on training and test set



(a) SFA error rates

(b) RF error rates

Fig. 5. (a) Error rates achieved with SFA in the interval of $[40, 200]$ training patterns (parameter setting: $n_{pp} = 12$). The test set error suddenly increases when the number of patterns is too small for a sufficient rank of the covariance matrix ($<$ 90 patterns), while the error on the training set stays constantly near zero. (b) The RF error rates also increase with fewer training patterns, but do not rise to an unnatural high level in contrast to SFA. Additionally, the training set error is a good predictor for the test set error on unseen data.

correlate with each other. This observation is an indicator for the robustness against overfitting and is well supported by the almost linear decreasing classification error in Fig. 5(b) when adding more training data.

Why is this not the case for the classification with SFA? The answer to this question becomes obvious when we look into SFA's computation of the output signal. The algorithm computes the covariance matrix of the expanded input signal and determines its eigenvalues. We show in the Appendix that a certain covariance matrix becomes rank deficient if too few training examples are available. This leads to an underdetermined linear system to be solved by the algorithm. As a necessary precondition to avoid rank deficiency in SFA, the following constraint has to be met (see Appendix):

$$N \geq D_{xp} + K \tag{7}$$

where N is the number of patterns available for training and K is the number of classes and D_{xp} is the dimension of the nonlinear expanded function space used by SFA.

Fig. 6(a) shows the results from SFA on a sufficient number of training records with 121 training records (from two persons) which is enough for $D_{xp} = 77$. There is some intra-class variation and a sharp inter-class separation, at least for the slower signals. A Gaussian classifier trained on $y_1, \ldots, y_4$ can learn quite robustly to separate the 5 classes. In contrast, Fig. 6(b) shows the results from SFA when there are too few training records. Here the number of 46 training records is much smaller than $D_{xp} + K = 82$ as required by Eq. (7). Consequently we get $\text{rank}(\text{Cov}(\vec{v})) = \text{rank}(\text{Cov}(\dot{\vec{v}})) = 46 - 1$, i. e. the matrices are rank deficient and the SFA ouput signals show absolutely *no intra-class variation*. The SFA model will select any of the $77 - 45$ dimensions not within the eigenvector basis of $\text{Cov}(\dot{\vec{v}})$, and almost surely this dimension will be meaningless for the test data. As a result we get high error rates (60%-100%) on independent test data.

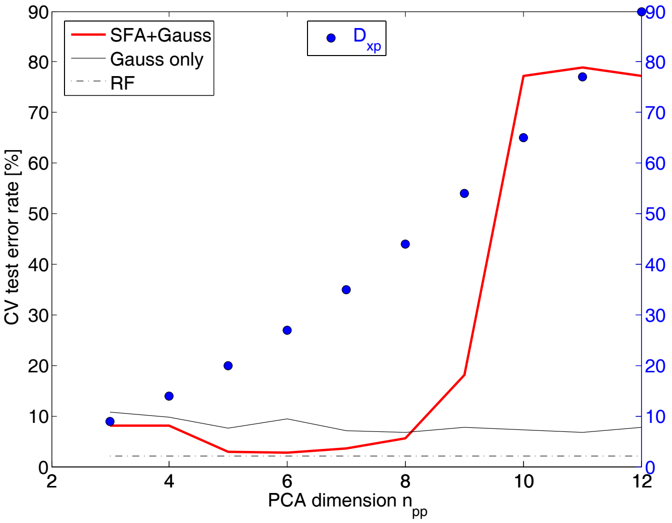


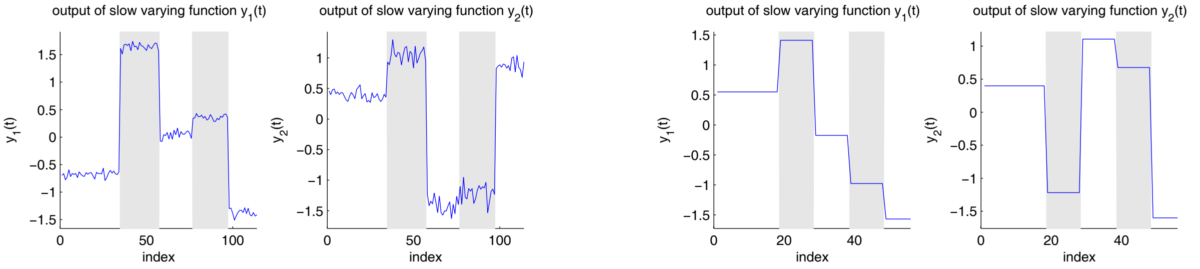
Fig. 7. SFA with marginal data: Too high n_{pp} yield unsatisfactory error rates. Experimental settings: 74 gestures from one person (ID 122), cross validation with 10 folds. Shown is the average out of 5 runs with different seeds for the fold generation.

IV. Discussion

While SFA works well in classification experiments with sufficient training data, where it achieves results comparable to the well-known Random Forest classifier, it shows severe limitations on cases with marginal training data. Are there possibilities to overcome these limitations? There are two options:

1) **Small n_{pp}:** We have shown that $N \geq D_{xp} + K$ (see Appendix or Eq. (7)) is a necessary condition to get a full-rank covariance matrix from N training records. For monomials of degree 2 the relation

$$D_{xp} = n_{pp} + \frac{n_{pp}(n_{pp} + 1)}{2} \tag{8}$$



(a) Rank sufficient

(b) Rank too small

Fig. 6. Training data output of the two slowest SFA signals for $D_{xp} = 77$ on the 5-class gesture classification problem when there are (a) sufficient training data (121 records) The x-axis shows the training record index. For better visualization the training records are ordered: first 35 class-1 records (leftmost white area), then 14 class-2 records (grey area) and so on up to class 5 (rightmost white area). (b) Too few training data (only 46 records, where at least $D_{xp} + K = 82$ records are required). The output on the training data shows severe overfitting.

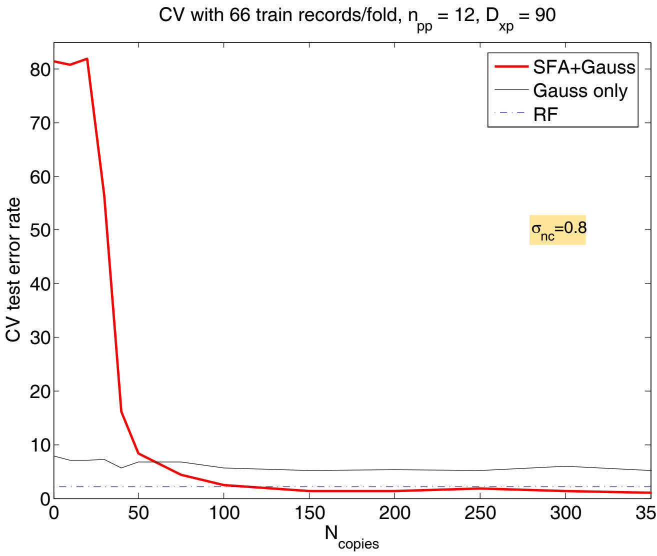


Fig. 8. Bootstrapping SFA in the case of marginal data: SFA with too few training data ($n_{train} = 66$, $N_{copies} = 0$) gets very high test set errors (approx. 80%). By applying a statistical bootstrap, which adds N_{copies} noisy replica to the 66 training data, we decrease the SFA test set error dramatically to values around 2% (thick line "SFA+Gauss"). This is by a factor of 3 smaller than the test set error from a Gaussian classifier (thin line "Gauss only"). Experimental settings: Same as Fig. 7, additionally $n_{pp} = 12$ (hence $D_{xp} = 90$) and $\sigma_{nc} = 0.8$.

holds. Therefore, one option is to decrease n_{pp} until the constraint of Eq. (7) is fulfilled. An example is shown in Fig. 7, where the task was to classify the gestures of one person: For each cross validation run we have 66 or 67 training data (90% of 74 gesture records in 10-fold cross validation). This leads to the necessary condition $D_{xp} \leq 66 - 5 = 59$ or $n_{pp} \leq 9$ acc. to Eq. (8), which is confirmed by the steep incline of the red curve between $n_{pp} = 9$ and 10 in Fig. 7. Best results are obtained with $n_{pp} \in \{5, 6\}$. – This option works, but it has the drawback that the number of information transferable to the classification algorithm is quite severely limited to 5 or 6 input dimensions.

2) **SFA+PB:** Another option is to keep n_{pp} at its desired value, but to enrich the training data by *parametric bootstrap* [HTF01, pp. 264]: This method increases the number of training instances by adding N_{copies} new records which are 'noisy copies' of original training records $\vec{X}$: Here we estimate first the centroid $\mu_i^{(m)}$ for each class m and the standard deviation $\sigma(X_i)$ for each component X_i of the data records $\vec{X}$. Then we form new patterns whose ith component is given by

$$X_i^{(nc)} = \mu_i^{(m)} + \sigma_{nc}\, Z_i \tag{9}$$

where Z_i is a random number drawn from the normal distribution $N(0, \sigma(X_i)^2)$ and σ_{nc} is a free strength parameter. Fig. 8 shows the resulting CV error rates as a function of N_{copies}. We expect to need at least $N_{copies} = 90 + 5 - 66 = 29$ additional records and we find from Fig. 8 that the steepest decline of the red curve is exactly at this value. However, to get a low error rate, N_{copies} should be higher, between 150 and 300. Note that the parametric bootstrap affects only the training data, and no changes to the test data are made. Therefore the CV test error rate remains realistic. – The option *parametric bootstrap* allows to put more of the original training information into the SFA model since we can work with n_{pp} =12 or higher and are not restricted to n_{pp} =6. We name this enhanced algorithm SFA+PB.

Parameter sensitivity We ran several tests with other values of parameter σ_{nc}. The algorithm is not very sensitive to this parameter, since we get nearly identical results if we halve or double the value, i.e. $\sigma_{nc} = 0.4$ or $\sigma_{nc} = 1.6$. But the right order of maginitude is important: With too small values, e.g. $\sigma_{nc} = 0.1$, the convergence as a function of N_{copies} is very slow, while with too large values, e.g. $\sigma_{nc} = 3.2$, the error rates rise after a small dip quickly to unsatisfactory high error rates of 25% and above.

Best results As a summary we show in Tab. III for the three tasks A), B) and C) (cf. Sec. III) and for our three algorithms the best error rates. (SFA-settings: $n_{pp} = 12$, $\sigma_{nc} = 0.8$ plus $N_{copies} = \{0, 200, 200\}$ in cases $\{$A),

TABLE III

BEST CROSS VALIDATION TEST SET ERROR RATES FOR THE THREE ALGORITHMS SFA+PB, RF AND GAUSSIAN CLASSIFIER.

Settings	CV test set error		
	SFA+PB	RF	Gaussian
A) random sampling (716 records)	(2.9±0.3)%	(2.1±0.3)%	(14.0±0.6)%
B) unseen persons (716 records)	(15.3±14.9)%	(14.8±15.3)%	(19.6±11.3)%
C) marginal data (74 records, 1 person)	(1.3±1.1)%	(2.8±0.4)%	(5.0±1.7)%

TABLE IV

CPU TIMES FOR SFA+PB, RF AND HIDDEN MARKOV [POP07A]. N IS THE NUMBER OF RECORDS FOR EACH TASK.

Task	N	CPU time [sec]			
		RF	SFA+PB		[Pop07a]
			n_{pp} =12	n_{pp} =15	
train	74	0.55	**0.52**	0.82	60
train	716	5.60	**5.16**	8.00	–
apply	64500	6.61	**1.09**	1.89	–

B), C)}, resp. RF settings: 500 trees, `mtry`=3). The shown results are averages and standard deviations from 10 runs with randomly different cross validation sets in cases A) and C). With the enhancement of parametric bootstrap the resulting CV test set errors from SFA+PB and RF are similar. They are in cases A) and C) considerably better than a Gaussian classifier (by a factor of 2-6).

Case B) has only one cross validation index (defined by the person ID). The standard deviation is in this case taken with respect to the different persons. It is very large (approximately as large as the CV error itself), because there is a large inter-person variation as already described in connection with Fig. 4.

As a surprise we found that case B) also benefits from the parametric bootstrap enhancement which was originally designed only for case C). If we run case B) without bootstrap ($N_{copies} = 0$ instead of 200) we get a CV error rate of 17.7% instead of 15.3%. We assume that the variation introduced by the noisy copies of the parametric bootstrap is beneficial for the generalization to unseen persons.

Other approaches We compared our results to another gesture recognition engine in order to exclude that our gesture set is perhaps 'easier' than others. The Hidden Markov model [SPHB08] is available as software library Wiigee [Pop07a] and we used it on a small set of our gestures (74 patterns). The error rate on this training set was 26% which has to be compared with the RF training set error of about 17% from Fig. 5(b) for the same training patterns (and with 15% for SFA+PB). Wiigee tests with larger training sets or on unseen test patterns were currently not possible due to memory and time restrictions. We conclude however that our gestures do not seem to be fundamentally 'easier' for other classifiers. – As another comparision we performed the "same-person CV error"-task of [SPHB08] on our data and with our SFA+PB model and got a $(2.3 \pm 1.5)\%$ CV error rate where [SPHB08] reports 10%.

Performance In Tab. IV we show the CPU times for SFA and RF. Both, RF and SFA, are quite fast algorithms, since the training takes only about 0.5 seconds for 74 records and 5 seconds for 716 records on a standard laptop processor[2]. This is faster by a factor of 100 compared to the Hidden Markov recognition engine Wiigee [Pop07a], which we tested with the same 74 patterns. The training time for the Gaussian classifier is negligible, actually it is contained in the reported SFA time. The times for SFA include the extra 200 parametric bootstrap records. They are based on an unoptimized MATLAB-implementation, so there may be some room for improvements. Since a trained SFA model has a very simple structure, we found that applying a trained model to new gestures (last line in Tab. IV) is for SFA 3–6 times faster than for RF.

Importance of Data Preparation We finally investigated the impact of the preliminary normalization steps described in Sec. II-C on the overall accuracy. As an example we skipped the amplitude normalization and repeated the experiments of Tab. III. The results of SFA+PB were 6.8% in case A and 3.2% in case C, i.e. they got considerably worse, roughly by a factor of 2. We conclude that data preparation is important, even if visually the difference between column 2 and column 3 in Fig. 2 appears small.

V. CONCLUSION AND OUTLOOK

In this paper we applied Slow Feature Analysis (SFA) to a time series classification problem originating from gesture recognition. SFA has proven to be considerably better (by a factor of 2-6) than a simple Gaussian classifier. Surprisingly we experienced that small problems are more difficult for SFA than larger problems. More precisely, the number of training records has a large impact on the classification performance for independent test records. If a sufficient number of data is available to train the classifier, the results of SFA are comparable to other state-of-the-art methods like the Random Forest (RF) predictor. But for marginal training data – more precisely: if the number of training patterns is lower than the dimension of the expanded function space of SFA – the algorithm severely overfits, which causes a high classification error on the test set.

We proposed here an enhancement to SFA for the case of marginal data, which is based on parametric bootstrapping. With that we got SFA results comparable or better than RF results on the same data. The parametric bootstrap was found to be beneficial for generalization to unseen persons, too.

We used here a very simple – however broadly applicable – parametric model for the bootstrap, namely a model based on a Gaussian noise distribution. For the gesture classification task we plan to investigate as future work a more specific parametric model where the creation of virtual patterns is based on gesture-specific geometric operators, e.g. rotations of real class patterns, or timeline operators like shift of start and stop point. We expect that with such virtual patterns

[2] Intel® Core2 Duo CPU T7700, 2.4GHz.

the generalization capabilities, especially towards unseen persons, can be enhanced.

A strength of SFA is its capability to analyze unsupervisedly continuous time streams. Therefore, a future application for SFA in gesture analysis is the automatic separation of gesture- and non-gesture-periods on the continuous timeline. At the moment, most gesture-recording devices require an additional button to be pressed while the gesture is performed. If SFA were able to distinguish gesture and non-gesture-periods this would permit continuous online gesture recognition inside applications. – A third line of research is to analyze in more detail, why some gestures / some persons are harder to classify than others.

In summary, the neuro-inspired algorithm SFA has shown to be fast and precise on classification tasks and it needs only few parameters. Due to its simple projection approach, the application of the trained model is 3–6 times faster than the already fast RF method. With SFA+PB, our new parametric-bootstrap extension, the algorithm can also deal with few training data, which was not possible for plain SFA.

ACKNOWLEDGEMENT

This work has been supported by the Bundesministerium für Bildung und Forschung (BMBF) under the grant SOMA (AIF FKZ 17N1009, ”Ingenieurnachwuchs”) and by the Cologne University of Applied Sciences under the research focus grant COSA. We are grateful to J. Piesk and M. Polak (Nurogames GmbH) as well as to P. Wagner for fruitful discussions and technical support.

APPENDIX: LOWER BOUND ON THE NUMBER OF TRAINING PATTERNS

Given a classification problem with N patterns $\vec{x}^{(m)}$ from K classes $m = 1, \ldots, K$: After SFA-expansion each pattern $\vec{x}^{(m)}$ is transformed to a point $\vec{v}^{(m)}$ in the nonlinear expanded function space of SFA with dimension D_{xp}. The SFA matrix $\boldsymbol{C} := \mathrm{Cov}(\Delta\vec{v})$ is formed from all intra-class difference vectors $\Delta\vec{v} = \vec{v}_i^{(m)} - \vec{v}_j^{(m)}$, where $\vec{v}_i^{(m)}$ and $\vec{v}_j^{(m)}$ are patterns belonging to the same class m. We show here that matrix $\boldsymbol{C}$ is rank deficient as soon as $N - K < D_{xp}$.

Lemma:

$$\mathrm{rank}(\boldsymbol{C}) \leq \min(D_{xp}, N - K) \tag{10}$$

Proof: Each of the N patterns belongs to one class m. If N_m is the number of patterns belonging to class $m = 1, \ldots, K$, we have

$$N_1 + N_2 + \ldots + N_K = N.$$

The difference vectors $\Delta\vec{v} = \vec{v}_i^{(m)} - \vec{v}_j^{(m)}$ for class m will span at most an $(N_m - 1)$-dimensional subspace, since the N_m points $\vec{v}_i^{(m)}$ can not span more than $N_m - 1$ dimensions. The matrix $\boldsymbol{C}$ is formed from these subspaces and thus can not have a rank larger than the direct sum of these subspaces:

$$\mathrm{rank}(\boldsymbol{C}) \leq N_1 - 1 + N_2 - 1 + \ldots + N_K - 1 = N - K.$$

Since on the other hand $\boldsymbol{C}$ is a square matrix with D_{xp} rows and columns, it can not have a rank larger than D_{xp}. In combination this proves the Lemma above.

Similarly, it is easy to see by spezializing to $K = 1$ that

$$\mathrm{rank}(\mathrm{Cov}(\vec{v})) \leq N - 1.$$

If $\mathrm{rank}(\boldsymbol{C}) = N - K < D_{xp}$ then $\boldsymbol{C}$ is rank deficient. There remain at least $D_{xp} - (N - K)$ dimensions perpendicular to all difference vectors. One of the directions in this perpendicular subspace is arbitrarily picked by SFA but it is almost surely not the best direction for slow variation. Therefore we will get 0% training set error (by construction), but with high probability a large test set error. In other words,

$$N = D_{xp} + K \tag{11}$$

is the minimum number of training records required to avoid rank deficiency and overfitting.

REFERENCES

[Ber05] P. Berkes. Pattern recognition with slow feature analysis. Cognitive Sciences EPrint Archive (CogPrint) 4104, http://cogprints.org/4104/, 2005.

[BFSO84] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and Regression Trees*. Chapman & Hall/CRC, January 1984.

[Bre01] L. Breiman. Random forests. In *Machine Learning*, pages 5–32, 2001.

[HTF01] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2001.

[Inc09] AiLive Inc. Livemove2: Motion recognition and tracking. http://www.ailive.net/lm2.html, 2009.

[Kon09] W. Konen. On the numeric stability of the SFA implementation sfa-tk. arXiv.org e-Print archive, http://arxiv.org/abs/0912.1064, December 2009.

[Lee08] J. C. Lee. Hacking the Nintendo Wii remote. *Pervasive Computing, IEEE*, 7(3):39–45, 2008.

[LW02] A. Liaw and M. Wiener. Classification and regression by randomForest. *R News*, 2(3):18–22, 2002.

[LW$^+$09] J. Liu, Z. Wang, et al. uWave: Accelerometer-based personalized gesture recognition and its applications. *IEEE Int. Conf. on Pervasive Computing and Communications*, 0:1–9, 2009.

[MS08] P. Malmestig and S. Sundberg. SignWiiver - implementation of sign language technology, 2008.

[Pop07a] B. Poppinga. Beschleunigungsbasierte 3D-Gestenerkennung mit dem Wii-Controller. Carl von Ossietzky Universität Oldenburg, Germany, Internal Reports, 2007.

[Pop07b] B. Poppinga. Wiigee, a Java-based gesture recognition library for the Wii remote. http://www.wiigee.org/, 2007.

[RBA08] M. Rehm, N. Bee, and E. André. Wave like an egyptian – accelerometer based gesture recognition for culture specific interactions. In *Procedings of HCI 2008 Culture, Creativity, Interaction*, 2008.

[Shi09] M. Shiels. Nintendo Wii sales hit 50 million. BBC News website, San Francisco, 2009.

[SPHB08] T. Schlömer, B. Poppinga, N. Henze, and S. Boll. Gesture recognition with a Wii controller. In *Proc. of TEI'08 Conference Tangible and Embedded Interaction*, pages 11–14, 2008.

[Wis98] L. Wiskott. Learning invariance manifolds. In *Proc. of the 5th Joint Symp. on Neural Computation*, volume 8, pages 196–203, San Diego, CA, 1998. Univ. of California.

[Wis03] L. Wiskott. Estimating driving forces of nonstationary time series with slow feature analysis. arXiv.org e-Print archive, http://arxiv.org/abs/cond-mat/0312317/, December 2003.

[WS02] L. Wiskott and T. Sejnowski. Slow feature analysis: Unsupervised learning of invariances. *Neural Computation*, 14(4):715–770, 2002.