Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

Learning Features for Activity Recognition with Shift-invariant Sparse Coding

Christian Vollmer, Horst-Michael Groß, Julian Eggert

2013

Preprint:

This is an accepted article published in Int. Conf. on Artificial Neural Networks (ICANN). The final authenticated version is available online at: https://doi.org/[DOI not available]

Learning Features for Activity Recognition with Shift-invariant Sparse Coding

Christian Vollmer[1], Horst-Michael Gross[1], and Julian P. Eggert[2]

[1]Ilmenau University of Technology,
Neuroinformatics and Cognitive Robotics Lab,
98684 Ilmenau, Germany
christian.vollmer@tu-ilmenau.de
[2]Honda Research Institute Europe GmbH
63073 Offenbach/Main, Germany
julian.eggert@honda-ri.de

Abstract. In activity recognition, traditionally, features are chosen heuristically, based on explicit domain knowledge. Typical features are statistical measures, like mean, standard deviation, etc., which are tailored to the application at hand and might not fit in other cases. However, Feature Learning techniques have recently gained attention for building approaches that generalize over different application domains. More conventional approaches, like Principal Component Analysis, and newer ones, like Deep Belief Networks, have been studied so far and yielded significantly better results than traditional techniques. In this paper we study the potential of Shift-invariant Sparse Coding (SISC) as an additional Feature Learning technique for activity recognition. We evaluate the performance on several publicly available activity recognition data sets and show that classification based on features learned by SISC outperforms other previously presented Feature Learning techniques.

Keywords: feature learning, activity recognition, sparse coding

1 Introduction

Context-aware computing provides intelligent systems with the ability to perceive the world from the user's perspective and allows to provide smart assistance where necessary. Human activity is an important cue for inferring the state and context of a user. In recent years, activity recognition has gained increased attention due to its usefulness and practical success in application domains such as medical diagnosis, rehabilitation, elderly care, assembly-line work, and human behavior modeling in general. As a result, a number of successful approaches have been built for recognition of a wide range of activities.

In most cases, activities are recognized by their body movements, since they are clearly defined by the motion and relative position of the user's body parts. Sensors, attached to the body or embedded into objects that are utilized throughout the activities, are used to capture those movements. One of the main issues

in activity recognition is that the sensor readings are typically noisy and often ambiguous. By applying signal processing and pattern recognition techniques, those data can be automatically analyzed, yielding a real-time classification of the activities.

In activity recognition the goal is to detect and classify contiguous portions of sensor data that contain the activities of interest. A widely adopted approach is the sliding window technique, where overlapping frames of the incoming multidimensional signal stream are extracted and a set of features is computed over each frame. The features are then categorized by means of some classifier. Popular features computed from the signal are mean, variance or standard deviation, energy, entropy, correlation between axes or discrete FFT coefficients (see e.g. [2] for a good comprehension). Common methods for classification based on the extracted features include Naive Bayes, Decision Trees, K-Nearest-Neighbors, and Support Vector Machines (see e.g. [6, 7]).

As mentioned by Plötz et al. in [7], feature extraction is usually a heuristic process that is driven by domain knowledge about the application at hand. This process has to be repeated for every new application domain and sometimes even for new sensor setups in the same domain. Thus, conventional approaches are usually tailored to specific applications. One way to overcome this restriction and build a general approach is to find methods to automatically discover useful features in the data or, in other words, adapt the features to the data.

Activity recognition techniques are usually built of two main components: (i) a feature extraction technique and (ii) a classifier. Most approaches are using the above mentioned standard features. Only recently, first attempts have been made in applying machine learning techniques to learn features from the data. Plötz et al. [7] use two feature learning techniques, namely PCA and Deep Belief Networks, to automatically discover features. By applying feature learning as a preprocessing step, the authors argue, a universal feature representation is created that captures the core characteristics of the data. The authors show that classification based on the discovered features yields significantly better results, then traditional techniques.

We would like to contribute to that line of research by investigating Shift-invariant Sparse Coding as another technique for feature learning in the area of activity recognition. The idea of Sparse Coding is that the data can be represented as a composition of sparsely distributed features. The data is imagined as consisting of two hidden components, (i) the set of features and (ii) activations of those features that mark, when a feature occurs in the data. The goal is to learn the features as well as their activations from the data in an unsupervised manner. The learning problem can be decomposed into two subproblems, the coding problem and the dictionary learning problem. In the coding problem, the features are assumed to be given and fixed. Here, the goal is to find a minimal set of activations such that the input data is best reconstructed, i.e. the error between input data and linear superposition of the features according to the activations is minimized. In the dictionary learning problem, the activations are assumed to be fixed and the goal is to adapt the features to the data given a known set of

activations. By initializing dictionary and activations randomly and alternating the two steps iteratively, one can learn both components simultaneously.

In the domain of time series processing, Sparse Coding has been mainly used for auditory signal coding. In [9], the authors aim at computing a sparse representation of natural audio signals in form of spike trains, where the spikes mark activations of a set of basis functions (or features), which are also learned, and represent an optimal dictionary. The authors argue that such a representation provides a very efficient encoding and uncovers the underlying event-like structure of the signal. More recently, Sparse Coding has been applied to find patterns in movement data, like walking cycles of human legs [4].

We use an approach similar to that published in our earlier work [10], where Sparse Coding has been utilized to learn features from handwriting data and generate handwritten characters using the features and statistics of their typical occurrence. The contribution of this work is the use of that framework for learning features from general activity data, which is much more diverse, and to build a simple classifier using those features.

We compare our results to those published by Plötz et al. [7], where PCA and Deep Belief Networks have been compared to conventional approaches for feature extraction. The authors evaluate their approaches on four publicly available activity recognition data sets. To be comparable, we will evaluate our approach on the same data. The rest of this work is organized as follows. We describe our approach in detail in Sec. 2. In Sec. 3, we compare our approach to previously published approaches on a number of activity recognition data sets. Finally, we discuss our work in Sec. 4 and give a brief outlook on its potential in activity recognition.

2 Method

Feature Learning We formalize Feature Learning as a Sparse Coding (SC) problem. In general, given an input signal, the goal in SC is to find features (or *basis vectors* in SC terms) and a sparse set of feature occurrences (or *activations* in SC terms) that, when linearly superimposed, reconstruct the input. We use a special kind of SC formulation as a Non-negative Matrix Factorization (NMF) problem. As detailed later, this problem can be solved by minimizing an energy function on the error between reconstruction and input plus a penalty on the activations. By imposing a non-negativity constraint, i.e. basis vectors and activations have to be non-negative, and a sparseness constraint on the activations, the resulting basis vectors are interpretable as features that constitute an alphabet underlying the data [5]. We further use a variant of NMF called Shift-NMF [1]. Shift-NMF introduces translation-invariant basis vectors. Thus, a basis vector can occur anywhere in the input, which is necessary for temporal signals with reoccurring features. In the following, we will give a formal description of the problem and the update equations.

As mentioned earlier, consecutive, overlapping frames are extracted from the input signal before feature extraction. Feature learning is then performed over

all frames simultaneously. Let $\mathbf{V}^d \in \mathbb{R}^{N \times T}$ denote the matrix of the N training frames of frame length T, where d indexes the dimensions of the signal. For ease of notation, we separate the dimensions of the signal into distinct matrices, indexed by d. A single frame is denoted as $\mathbf{V}_n^d$ and the scalar elements by $V_{n,t}^d$. Let $\mathbf{W}^d \in \mathbb{R}^{K \times L}$ be the matrix of K basis vectors of length L, with elements $W_{k,l}^d$. We denote the single basis vectors by $\mathbf{W}_k^d$. Let $\mathbf{H} \in \mathbb{R}^{N \times K \times T}$ be the tensor of activations $H_{n,k,t}$ of the k-th basis vector at time t for frame n.

In NMF the input, basis vectors, and activations are constrained to be non-negative. Thus, for NMF to be applicable, the input signal has to be made non-negative. We do this by doubling the number of input dimensions and projecting the negation of its negative parts to the new dimensions. The non-negative input $\tilde{\mathbf{V}}^{\tilde{d}}$ as used in the calculations below is then given by

$$\tilde{\mathbf{V}}^{2d} = \max(\mathbf{V}^d, 0), \quad \tilde{\mathbf{V}}^{2d+1} = \max(-\mathbf{V}^d, 0) \ . \tag{1}$$

For ease of notation, we resubstitute $\tilde{\mathbf{V}}^{\tilde{d}}$ with $\mathbf{V}^d$ again. However, please keep in mind, that $\mathbf{V}^d$ denotes the non-negative input from now on.

We learn $\mathbf{W}^d$ and $\mathbf{H}$ with NMF by minimizing the following energy function

$$F = \frac{1}{2} \sum_d \left\| \mathbf{V}^d - \mathbf{R}^d \right\|_2^2 + \lambda \left\| \mathbf{H} \right\|_1 \ . \tag{2}$$

The matrices $\mathbf{R}^d \in \mathbb{R}^{N \times T}$ are the reconstructions of the frames by activation of the basis vectors $\mathbf{W}^d$ through activations $\mathbf{H}$, which can be formalized as

$$R_{n,t}^d = \sum_k \mathrm{conv}_{\mathbf{H}_{n,k}, \overline{\mathbf{W}}_k^d}(t) \ , \tag{3}$$

where $\mathrm{conv}_{X,Y}(t)$ denotes temporal convolution of X with filter Y at time t. Here, we introduced *normalized basis vectors* $\overline{\mathbf{W}}_k^d$, where the normalization is done jointly over all dimensions d. This normalization is necessary during learning to avoid scaling problems as described in [1].

The energy function in eq. 2 formalizes the standard approximation scheme commonly used for Sparse Non-negative Matrix Factorization. The first term is the distance measure and the second term is a penalization of the overall sum of activations, weighed by the sparseness weight λ. Due to lack of space, we refer to a more detailed explanation in our earlier work [10].

This optimization problem can be solved by alternating the update of one of the factors $\mathbf{H}$ or $\mathbf{W}^d$, while holding the other fixed. Due to the non-negativity of the two factors, the update can be formulated as an exponentiated gradient descent with better convergence properties then pure gradient descent (see e.g. [5]). For a detailed description of the optimization procedure, we refer to [10].

After applying NMF to the data, we have a representation of the input in terms of learned basis vectors and activations. We interpret the basis vectors as features and their corresponding activations as temporal occurrences of those features. For illustrations of the resulting representation, we refer to [10] again.

The full procedure as described above is applied in the training phase for learning the features. During application phase, this optimization is applied again, but the features are given and held fixed and only the activations are updated. Thus only steps 1 to 3 have to be iterated and step 4 is left out.

Classification As mentioned earlier, we compare our Feature Learning technique to the ones presented in Plötz et al. [7]. To compare the performance of the different techniques, Plötz et al. use the K-Nearest-Neighbor algorithm as a simple classifier. To get comparable results, we also adopt this technique. The K-Nearest-Neighbor algorithm represents a simple but effective standard approach that simply stores all feature vectors from a training set and assigns to a new feature vector the label of the majority of its k nearest neighbors in the feature space. We emphasize that we do not aim at presenting the best possible classifier, but merely want to compare our feature extraction technique to the ones presented earlier.

Applying the classifier to the activations for frame n $\mathbf{H}_n$ (which encodes the temporal positions of features within the frame) directly would yield bad results, because instances of the same class generally differ slightly in the temporal positions of features. Thus, to be temporally invariant within a frame, we sum the activations over the temporal dimension of the frame, yielding the summed activations for each feature as a feature vector that is passed to the classifier.

3 Experiments

Data sets We evaluate our method in comparison to the results presented Plötz et al. [7]. The authors tested their methods on four publicly available data sets, which will be described briefly in the following.

Pham et al. [6] describe the data set "Ambient Kitchen 1.0" (AK) consisting of food preparation routines with sensor-equipped kitchen utensils. 20 Persons either prepared a sandwich or a salad, using two kinds of knifes, a spoon, and a scoop, the handle of each of which was equipped with a tri-axial accelerometer. In total, the data consist of 4 hours of recording, sampled at 40Hz, where about 50% cover ten typical food preparation activities.

Huynh et al. [3] describe the dataset "Darmstadt Daily Routines" (DA) consisting of 35 activities of daily living (e.g. brushing teeth, setting the table), captured by two tri-axial accelerometers (one wrist-worn, the other in the pocket) in a lab-like environment. After preprocessing, they yield a sampling frequency of 2.5Hz. In [7] only results for the pocket sensor are presented, hence we also use only the pocket sensor.

Zappi et al. [11] describe the dataset "Skoda Mini Checkpoint" (Skoda) consisting of activities of an assembly-line worker, wearing a number of accelerometers on his arms, while performing ten tasks of quality checks for correct assembly of car parts. The data consists of three hours of recording, sampled at 96Hz. As in [7] we only use a single accelerometers at the right arm.

Roggen et al. [8] describe the dataset for the "Opportunity Gesture Challenge" (Opp) consisting of activities of daily living in a kitchen environment,

recorded with multiple accelerometers, body-worn or embedded into objects. Multiple subjects have been recorded on different days. As in [7] we only consider the data of the right arm of the subjects. Also as in [7] we only consider 10 low-level activities and one *unknown* activity class. The data is sampled at 64Hz.

In [7] only a small excerpt of the data is used, consisting of recordings of one subject, because the full data set was not published yet at the time of the publication. Since we have the full data set and the subject used by [7] is left unspecified, it is difficult to get a fair comparison. Thus, for the comparison to [7] to be fairer, we evaluate our method on each single person separately and present the minimum accuracy over all subjects in the dataset.

Before applying SISC, we normalized all datasets by PCA and resampled them to 10Hz, which seemed to be sufficient for activity recognition.

Features In Plötz et al., four feature extraction techniques are presented, namely Statistical, FFT, PCA, RBM. Further a preprocessing technique based on the empirical cumulative distribution function (ECDF) is used to normalize the data. ECDF is combined with PCA and RBM and called PCA+ECDF and RBM+ECDF. We will describe the methods very briefly here. Please refer to [7] for a deeper explanation.

The method *Statistical* refers to the most commonly used feature extraction method, which is to extract statistical measures, like mean, standard deviation, energy, and entropy over the whole frame. For each sensor, Plötz et al. use x, y, z, pitch, and roll and compute the statistics over each channel independently plus the pairwise correlation between x, y, and z, resulting in a 23-D feature vector for each frame.

The method *FFT* is also widely used and consists of computing for each channel independently the Fourier coefficients through the Discrete Fourier Transform (DFT). Usually only a subset of the resulting Fourier coefficients is used. In [7] the first 10 are used.

In the method *PCA*, features are learned by Principal Component Analysis (PCA). The Eigenvectors corresponding to the largest Eigenvalues are kept as features. In [7] the 30 largest Eigenvectors are used.

The method *RBM* refers to a technique based on Deep Belief Networks. Deep Belief Networks are auto-encoders that use a hierarchy of Restricted Boltzman Machines (RBM) for extracting useful features. It has been shown that Deep Belief Networks can uncover features in the data, which, in turn, can be used for classification. In [7] an architecture consisting of four layers with 1024 Units in each hidden layer and 30 units in the output layer is used.

Additionally to the methods described above, we present results using Shift-invariant Sparse Coding (SISC). For SISC a number of parameters have to be chosen. We applied grid search to find a single set of parameters that gave the highest average classification accuracy for all data sets. The final parameters are as follows: the frame size T is 7 seconds, the width of the basis vectors L is 2.5 seconds (note however that the effective width, i.e. the part of the basis vector

that is actually used and above zero, can vary), the number of basis vectors K is 20, and the sparseness parameter λ is 0.1.

In the classification stage of our method we used a K-Nearest-Neighbor classifier with K set to 5. Higher values had no significant impact on the results.

We validated our results by class-balanced 10-fold cross-validation. Regarding computational performance, the learning phase takes up to half an hour on a 2.6GHz Quad-Core CPU for the larger data sets. The application phase takes about 30 milliseconds per frame, which is well within real-time boundaries.

Results We conducted one experiment devoted to the classification accuracy using the respective feature extraction techniques. In Fig. 1 the classification accuracies for the seven techniques are shown. The results of the first six techniques

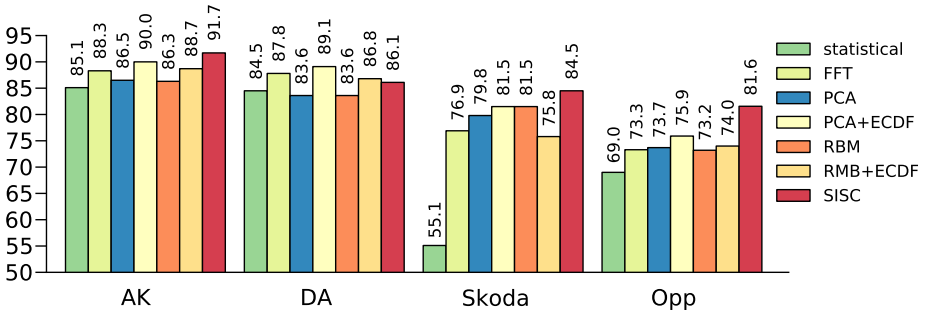

Fig. 1: Classification accuracies of the seven approaches for the four datasets.

are taken directly from [7]. The authors state, that these results are comparable with those published earlier for those data sets. The seventh technique *SISC* is our shift-invariant Sparse Coding approach. Interestingly, *SISC* yields significantly better results on three of the four data sets. We reason that this is mainly due to the shift-invariant nature of this coding technique, which learns features independently of their position in a particular frame and can, in turn, detect a feature even if it is shifted slightly in a frame. Because when a pattern is slightly shifted in a frame, the sum over activations, and hence the feature vector, does not change, which is not the case for PCA and RBM.

In summary, one can use SISC to successfully learn features in an unsupervised manner without prior domain knowledge. Further, SISC outperforms PCA and Deep Belief Networks as a Feature Learning technique in some cases. The caveat, however, is that PCA and Deep Belief Networks are faster during application phase, since they can be applied by simple matrix multiplication, while in SISC a few iterative steps have to be computed for each frame. But, for small frames sizes of up to a few seconds the computational time lies well within real-time boundaries.

4 Conclusion

We have presented Shift-invariant Sparse Coding (SISC) as a Feature Learning technique for activity recognition. We compared our method to traditional methods for feature extraction and to two recent approaches for Feature Learning, namely PCA and Deep Belief Networks. The evaluation was performed on four publicly available data sets. The results show that SISC outperforms all other methods on three of the four data sets. Thus, SISC has great potential for application in activity recognition.

Plötz et al. [7] mention that Feature Learning techniques can potentially be used for further sub-frame analysis, which is important if one wants, e.g., to assess certain properties, like the quality of the activities performed. SISC is particularly suited for that task, because the sparse nature of the representation and the shift-invariance allows features to be well localized in time. Thus, the exact position of the features or the relative positions of different features in a frame can be used as a cue for sub-frame analysis.

References

1. Eggert, J., Wersing, H., Korner, E.: Transformation-invariant representation and NMF. In: Proc. of the 2004 IEEE Int. Joint Conf. on Neural Networks, 2004. vol. 4, pp. 2535–2539. IEEE (2004)
2. Huynh, T., Schiele, B.: Analyzing features for activity recognition. Proc. of the 2005 Joint Conf. on Smart Objects and Ambient Intelligence (2005)
3. Huynh, T., Fritz, M., Schiele, B.: Discovery of activity patterns using topic models. In: Proc. of the 10th Int. Conf. on Ubiquitous computing. pp. 10–19. ACM Press, New York, New York, USA (2008)
4. Kim, T., Shakhnarovich, G., Urtasun, R.: Sparse Coding for Learning Interpretable Spatio-Temporal Primitives. In: Proc. Neural Inf. Process. Syst. (NIPS). vol. 22 (Dec 2010)
5. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. Nature 401(6755), 788–91 (Oct 1999)
6. Pham, C., Olivier, P.: Slice&dice: Recognizing food preparation activities using embedded accelerometers. Ambient Intelligence pp. 34–43 (2009)
7. Plötz, T., Hammerla, N.Y., Olivier, P.: Feature Learning for Activity Recognition in Ubiquitous Computing. In: Proc. of the Twenty-Second Int. Joint Conf. on Artificial Intelligence. pp. 1729–1734 (2011)
8. Roggen, D.e.a.: Collecting complex activity datasets in highly rich networked sensor environments. 2010 Seventh Int. Conf. on Networked Sensing Systems (INSS) pp. 233–240 (Jun 2010)
9. Smith, E., Lewicki, M.S.: Efficient coding of time-relative structure using spikes. Neural Computation 17(1), 19–45 (2005)
10. Vollmer, C., Eggert, J., Groß, H.M.: Generating Motion Trajectories by Sparse Activation of Learned Motion Primitives. In: Artificial Neural Networks and Machine Learning (ICANN 2012), vol. 7552, pp. 637–644. Springer Berlin (2012)
11. Zappi, P., Lombriser, C., Stiefmeier, T.: Activity recognition from on-body sensors: accuracy-power trade-off by dynamic sensor selection. Wireless Sensor Networks pp. 17–33 (2008)