Hierarchical Feature Extraction for Compact Representation and Classification of Datasets

Markus Schubert and Jens Kohlmorgen

Fraunhofer FIRST.IDA
Kekuléstr. 7, 12489 Berlin, Germany
{markus,jek}@first.fraunhofer.de
http://ida.first.fraunhofer.de

Abstract. Feature extraction methods do generally not account for hierarchical structure in the data. For example, PCA and ICA provide transformations that solely depend on global properties of the overall dataset. We here present a general approach for the extraction of feature hierarchies from datasets and their use for classification or clustering. A hierarchy of features extracted from a dataset thereby constitutes a compact representation of the set that on the one hand can be used to characterize and understand the data and on the other hand serves as a basis to classify or cluster a collection of datasets. As a proof of concept, we demonstrate the feasibility of this approach with an application to mixtures of Gaussians with varying degree of structuredness and to a clinical EEG recording.

1 Introduction

The vast majority of feature extraction methods does not account for hierarchical structure in the data. For example, PCA [1] and ICA [2] provide transformations that solely depend on global properties of the overall data set. The ability to model the hierarchical structure of the data, however, might certainly help to characterize and understand the information contained in the data. For example, neural dynamics are often characterized by a hierarchical structure in space and time, where methods for hierarchical feature extraction might help to group and classify such data. A particular demand for these methods exists in EEG recordings, where slow dynamical components (sometimes interpreted as internal "state" changes) and the variability of features make data analysis difficult.

Hierarchical feature extraction is so far mainly related to 2-D pattern analysis. In these approaches, pioneered by Fukushima's work on the Neocognitron [3], the hierarchical structure is typically a priori hard-wired in the architecture and the methods primarily apply to a 2-D grid structure. There are, however, more recent approaches, like local PCA [4] or tree-dependent component analysis [5], that are promising steps towards structured feature extraction methods that derive also the structure from the data. While local PCA in [4] is not hierarchical and tree-dependent component analysis in [5] is restricted to the context of ICA, we here present a general approach for the extraction of feature hierarchies and

their use for classification and clustering. We exemplify this by using PCA as the core feature extraction method.

In [6] and [7], hierarchies of two-dimensional PCA projections (using probabilistic PCA [8]) were proposed for the purpose of visualizing high-dimensional data. For obtaining the hierarchies, the selection of sub-clusters was performed either manually [6] or automatically by using a model selection criterion (AIC, MDL) [7], but in both cases based on two-dimensional projections. A 2-D projection of high-dimensional data, however, is often not sufficient to unravel the structure of the data, which thus might hamper both approaches, in particular, if the sub-clusters get superimposed in the projection. In contrast, our method is based on hierarchical clustering in the original data space, where the structural information is unchanged and therefore undiminished. Also, the focus of this paper is not on visualizing the data itself, which obviously is limited to 2-D or 3-D projections, but rather on the extraction of the hierarchical structure of the data (which can be visualized by plotting trees) and on replacing the data by a compact hierarchical representation in terms of a tree of extracted features, which can be used for classification and clustering. The individual quantity to be classified or clustered in this context, is a tree of features representing a *set* of data points. Note that classifying sets of points is a more general problem than the well-known problem of classifying individual data points. Other approaches to classify sets of points can be found, e.g., in [9, 10], where the authors define a kernel on sets, which can then be used with standard kernel classifiers.

The paper is organized as follows. In section 2, we describe the hierarchical feature extraction method. In section 3, we show how feature hierarchies can be used for classification and clustering, and in section 4 we provide a proof of concept with an application to mixtures of Gaussians with varying degree of structuredness and to a clinical EEG recording. Section 5 concludes with a discussion.

2 Hierarchical Feature Extraction

We pursue a straightforward approach to hierarchical feature extraction that allows us to make any standard feature extraction method hierarchical: we perform *hierarchical clustering* of the data prior to feature extraction. The feature extraction method is then applied locally to each *significant* cluster in the hierarchy, resulting in a representation (or replacement) of the original dataset in terms of a *tree of features*.

2.1 Hierarchical Clustering

There are many known variants of hierarchical clustering algorithms (see, e.g., [11, 12]), which can be subdivided into *divisive* top-down procedures and *agglomerative* bottom-up procedures. More important than this procedural aspect, however, is the dissimilarity function that is used in most methods to quantify the dissimilarity between two clusters. This function is used as the criterion to

determine the clusters to be split (or merged) at each iteration of the top-down (or bottom-up) process. Thus, it is this function that determines the clustering result and it implicitly encodes what a "good" cluster is. Common agglomerative procedures are single-linkage, complete-linkage, and average-linkage. They differ simply in that they use different dissimilarity functions [12].

We here use Ward's method [13], also called the minimum variance method, which is agglomerative and successively merges the pair of clusters that causes the smallest increase in terms of the total sum-of-squared-errors (SSE), where the error is defined as the Euclidean distance of a data point to its cluster mean. The increase in square-error caused by merging two clusters, D_i and D_j, is given by

$$d\left(D_i, D_j\right) = \sqrt{\frac{n_i n_j}{n_i + n_j}}\, \|\mathbf{m}_i - \mathbf{m}_j\|, \tag{1}$$

where n_i and n_j are the number of points in each cluster, and $\mathbf{m}_i$ and $\mathbf{m}_j$ are the means of the points in each cluster [12]. Ward's method can now simply be described as a standard agglomerative clustering procedure [11, 12] with the particular dissimilarity function d given in Eq. (1). We use Ward's criterion, because it is based on a *global* fitness criterion (SSE) and in [11] it is reported that the method outperformed other hierarchical clustering methods in several comparative studies. Nevertheless, depending on the particular application, other criteria might be useful as well.

The result of a hierarchical clustering procedure that successively splits or merges two clusters is a binary tree. At each hierarchy level, $k = 1, ..., n$, it defines a partition of the given n samples into k clusters. The leaf node level consists of n nodes describing a partition into n clusters, where each cluster/node contains exactly one sample. Each hierarchy level further up contains one node with edges to the two child nodes that correspond to the clusters that have been merged.

The tree can be depicted graphically as a *dendrogram*, which aligns the leaf nodes along the horizontal axis and connects them by lines to the higher level nodes along the vertical axis. The position of the nodes along the vertical axis could in principle correspond linearly to the hierarchy level k. This, however, would reveal almost nothing of the structure in the data. Most of the structural information is actually contained in the dissimilarity values. One therefore usually positions the node at level k vertically with respect to the dissimilarity value of its two corresponding child clusters, D_i and D_j,

$$\delta(k) = d\left(D_i, D_j\right). \tag{2}$$

For $k = n$, there are no child clusters, and therefore $\delta(n) = 0$ [11]. The function δ can be regarded as *within-cluster dissimilarity*. By using δ as the vertical scale in a dendrogram, a large gap between two levels, for example k and $k+1$, means that two very dissimilar clusters have been merged at level k.

2.2 Extracting a Tree of Significant Clusters

As we have seen in the previous subsection, a hierarchical clustering algorithm always generates a tree containing $n - 1$ non-singleton clusters. This does not

necessarily mean that any of these clusters is clearly separated from the rest of the data or that there is any structure in the data at all. The identification of clearly separated clusters is usually done by visual inspection of the dendrogram, i.e. by identifying large gaps. For an automatic detection of significant clusters, we use the following straightforward criterion

$$\frac{\delta(parent(k))}{\delta(k)} > \alpha, \quad \text{for } 1 < k < n, \tag{3}$$

where $parent(k)$ is the parent cluster level of the cluster obtained at level k and α is a significance threshold. If a cluster at level k is merged into a cluster that has a within-cluster dissimilarity which is more than α times higher than that of cluster k, we call cluster k a significant cluster. That means that cluster k is significantly more compact than its merger (in the sense of the dissimilarity function). Note that this does not necessarily mean that the sibling of cluster k is also a significant cluster, as it might have a higher dissimilarity value than cluster k. The criterion directly corresponds to the relative increase of the dissimilarity value in a dendrogram from one merger level to the next. For small clusters that contain only a few points, the relative increase in dissimilarity can be large just because of the small sample size. To avoid that these clusters are detected as being significant, we require a minimum cluster size M for significant clusters.

After having identified the significant clusters in the binary cluster tree, we can extract the tree of significant clusters simply by linking each significant cluster node to the next highest significant node in the tree, or, if there is none, to the root node (which is just for the convenience of getting a tree and not a forest). The tree of significant clusters is generally much smaller than the original tree and it is not necessarily a binary tree anymore. Also note that there might be data points that are not in any significant cluster, e.g., outliers.

The criterion in (3) is somewhat related to the criterion in [14], which is used to *take out* clusters from the merging process in order to obtain a plain, non-hierarchical clustering. The criterion in [14] accounts for the relative change of the absolute dissimilarity increments, which seems to be somewhat less intuitive and unnecessarily complicated. This criterion might also be overly sensitive to small variations in the dissimilarities.

2.3 Obtaining a Tree of Features

To obtain a representation of the original dataset in terms of a *tree of features*, we can now apply any standard feature extraction method to the data points in each significant cluster in the tree and then replace the data points in the cluster by their corresponding features. For PCA, for example, the data points in each significant cluster are replaced by their mean vector and the desired number of principle components, i.e. the eigenvectors and eigenvalues of the covariance matrix of the data points. The obtained hierarchy of features thus constitutes a compact representation of the dataset that does not contain the individual data points anymore, which can save a considerable amount of memory. This

representation is also independent of the size of the dataset. The hierarchy can on the one hand be used to analyze and understand the structure of the data, on the other hand – as we will further explain in the next section – it can be used to perform classification or clustering in cases where the individual input quantity to be classified (or clustered) is an entire dataset and not, as usual, a single data point.

3 Classification of Feature Trees

The classification problem that we address here is not the well-known problem of classifying individual *data points* or vectors. Instead, it relates to the classification of objects that are *sets* of data points, for example, time series. Given a "training set" of such objects, i.e. a number of datasets, each one attached with a certain class label, the problem consists in assigning one class label to each new, unlabeled dataset. This can be accomplished by transforming each individual dataset into a tree of features and by defining a suitable distance function to compare each pair of trees. For example, trees of principal components can be regarded as (hierarchical) mixtures of Gaussians, since the principal components of each node in the tree (the eigenvectors and eigenvalues) describe a normal distribution, which is an approximation to the true distribution of the underlying data points in the corresponding significant cluster. Two mixtures (sums) of Gaussians, f and g, corresponding to two trees of principal components (of two datasets), can be compared, e.g., by using the the squared L_2-Norm as distance function, which is also called the integrated squared error (ISE),

$$ISE(f,g) = \int (f-g)^2 \, d\mathbf{x}. \tag{4}$$

The ISE has the advantage that the integral is analytically tractable for mixtures of Gaussians.

Note that the computation of a tree of principal components, as described in the previous section, is in itself an interesting way to obtain a mixture of Gaussians representation of a dataset: without the need to specify the number of components in advance and without the need to run a maximum likelihood (gradient ascent) algorithm like, for example, expectation–maximization [15], which is prone to get stuck in local optima.

Having obtained a distance function on feature trees, the next step is to choose a classification method that only requires pairwise distances to classify the trees (and their corresponding datasets). A particularly simple method is first-nearest-neighbor (1-NN) classification. For 1-NN classification, the tree of a test dataset is assigned the label of the nearest tree of a collection of trees that were generated from a labeled "training set" of datasets. If the generated trees are sufficiently different among the classes, first- (or k-) nearest-neighbor classification can already be sufficient to obtain a good classification result, as we demonstrate in the next section.

In addition to classification, the distance function on feature trees can also be used to *cluster* a collection of datasets by clustering their corresponding trees. Any clustering algorithm that uses pairwise distances can be used for this purpose [11, 12]. In this way it is possible to identify homogeneous groups of datasets.

4 Applications

4.1 Mixtures of Gaussians

As a proof of concept, we demonstrate the feasibility of this approach with an application to mixtures of Gaussians with varying degree of structuredness. From three classes of Gaussian mixture distributions, which are exemplarily shown in Fig. 1(a)-(c), we generated 10 training samples for each class, which constitute the training set, and a total of 100 test samples constituting the test set. Each sample contains 540 data points. The mixture distribution of each test sample was chosen with equal probability from one of the three classes.

Next, we generated the binary cluster tree from each sample using Ward's criterion. Examples of the corresponding dendrograms for each class are shown in Fig. 1(d)-(f) (in gray). We then determined the significant clusters in each tree, using the significance factor $\alpha = 3$ and the minimum cluster size $M = 40$. In Fig. 1(d)-(f), the significant clusters are depicted as black dots and the extracted trees of significant clusters are shown by means of thick black lines. The cluster of each node in a tree of significant clusters was then replaced by the principle components obtained from the data in the cluster, which turns the tree of clusters into a tree of features. In Fig. 1(g)-(i), the PCA components of all significant clusters are shown for the three example datasets from Fig. 1(a)-(c).

Finally, we classified the feature trees obtained from the test samples, using the integrated squared error (Eq. (4)) and first-nearest-neighbor classification. We obtained a nearly perfect accuracy of 98% correct classifications (i.e.: only two misclassifications), which can largely be attributed to the circumstance that the structural differences between the classes were correctly exposed in the tree structures. This result demonstrates that an appropriate representation of the data can make the classification problem very simple.

4.2 Clinical EEG

To demonstrate the applicability of our approach to real-world data, we used a clinical recording of human EEG. The recording was carried out in order to screen for pathological features, in particular the disposedness to epilepsy. The subject went through a number of experimental conditions: eyes open (EO), eyes closed (EC), hyperventilation (HV), post-hyperventilation (PHV), and, finally, a stimulation with stroboscopic light of increasing frequency (PO: photic on). During the photic phase, the subject kept the eyes closed, while the rate of light flashes was increased every four seconds in steps of 1 Hz, from 5 Hz to 25 Hz.

The obtained recording was subdivided into 507 epochs of fixed length (1s). For each epoch, we extracted four features that correspond to the power in specific frequency bands of particular EEG electrodes.[1] The resulting set of four-dimensional feature vectors was then analyzed by our method. For the hierarchical clustering, we used Ward's method and found the significant clusters depicted in Fig. 2. The extracted tree of significant clusters consists of a two-level hierarchy. As expected, the majority of feature vectors in each sub-cluster corresponds to one of the experimental conditions. By applying PCA to each sub-cluster and replacing the data of each node with its principle components, we obtain a tree of features, which constitutes a compact representation of the original dataset. It can then be used for comparison with trees that arise from normal or various kinds of pathological EEG, as outlined in section 3.

5 Discussion

We proposed a general approach for the extraction of feature hierarchies from datasets and their use for classification or clustering. The feasibility of this approach was demonstrated with an application to mixtures of Gaussians with varying degree of structuredness and to a clinical EEG recording. In this paper we focused on PCA as the core feature extraction method. Other types of feature extraction, like, e.g., ICA, are also conceivable, which then should be complemented with an appropriate distance function on the feature trees (if used for classification or clustering). The basis of the proposed approach is hierarchical clustering. The quality of the resulting feature hierarchies thus depends on the quality of the clustering. Ward's criterion tends to find compact, hyperspherical clusters, which may not always be the optimal choice for a given problem. Therefore, one should consider to adjust the clustering criterion to the problem at hand. Our future work will focus on the application of this method to classify normal and pathological EEG. By comparing the different tree structures, the hope is to gain a better understanding of the pathological cases.

Acknowledgements: This work was funded by the German BMBF under grant 01GQ0415 and supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778.

[1] In detail: (I.) the power of the α-band (8–12 Hz) at the electrode positions O1 and O2 (according to the international 10–20 system), (II.) the power of 5 Hz and its harmonics (except 50 Hz) at electrode F4, (III.) the power of 6 Hz and its harmonics at electrode F8, and (IV.) the power of the 25–80 Hz band at F7.

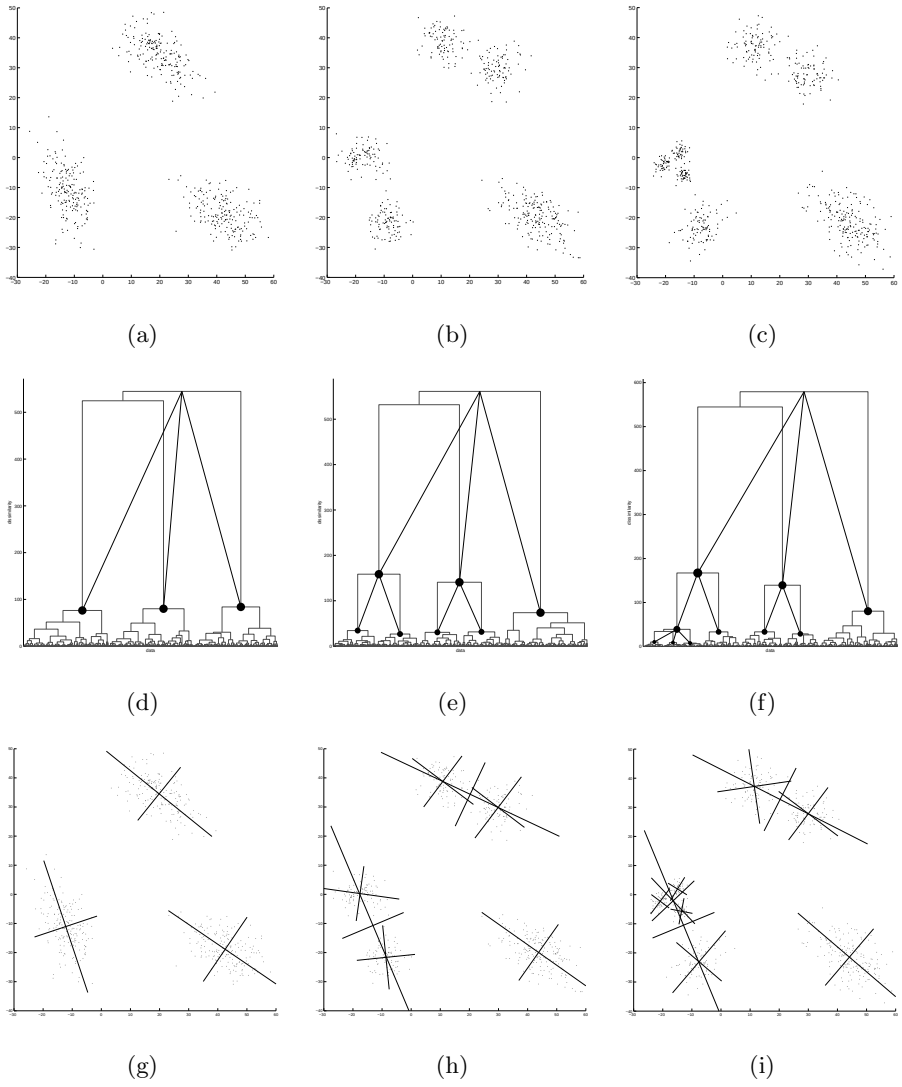

Fig. 1. (a)-(c) Example datasets for the three types of mixture distributions used in the application. (d)-(f) The corresponding dendrograms for each example dataset (gray) and the extracted trees of significant clusters (black). Note that the extracted tree structure exactly corresponds to the structure in the data. (g)-(i) The PCA components of all significant clusters. The components are contained in the tree of features.

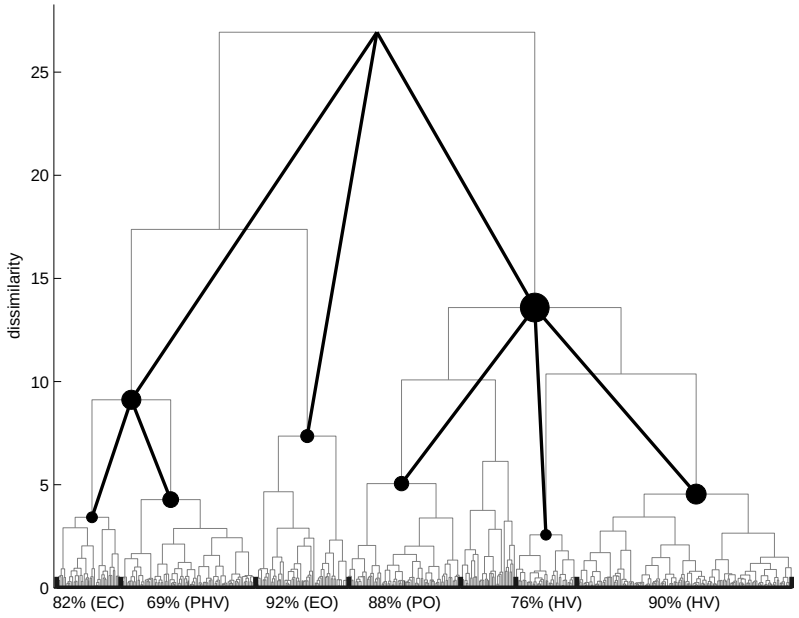

Fig. 2. The tree of significant clusters (black), obtained from the underlying dendrogram (gray) for the EEG data. The data in each significant sub-cluster largely corresponds to one of the experimental conditions (indicated in %): eyes open (EO), eyes closed (EC), hyperventilation (HV), post-hyperventilation (PHV), and 'photic on' (PO).

Bibliography

[1] Jolliffe, I.: Principal Component Analysis. Springer-Verlag, New York (1986)

[2] Hyvarinen, A., Karhunen, J., Oja, E.: Independent Component Analysis. Wiley (2001)

[3] Fukushima, K.: Neural network model for a mechanism of pattern recognition unaffected by shift in position — neocognitron. Transactions IECE **62-A**(10) (1979) 658–665

[4] Bregler, C., Omohundro, S.: Surface learning with applications to lipreading. In Cowan, J., Tesauro, G., Alspector, J., eds.: Advances in Neural Information Precessing Systems. Volume 6., San Mateo, CA, Morgan Kaufmann Publishers (1994) 43–50

[5] Bach, F., Jordan, M.: Beyond independent components: Trees and clusters. Journal of Machine Learning Research **4** (2003) 1205–1233

[6] Bishop, C., Tipping, M.: A hierarchical latent variable model for data visualization. IEEE Transactions on Pattern Analysis and Machine Intelligence **20**(3) (1998) 281–293

[7] Wang, Y., Luo, L., Freedman, M., Kung, S.: Probabilistic principal component subspaces: A hierarchical finite mixture model for data visualization. IEEE Transactions on Neural Networks **11**(3) (2000) 625–636

[8] Tipping, M., Bishop, C.: Probabilistic principal component analysis. Journal of the Royal Statistical Society: Series B **61**(3) (1999) 611–622

[9] Kondor, R., Jebara, T.: A kernel between sets of vectors. In Fawcett, T., Mishra, N., eds.: Proceedings of the ICML, AAAI Press (2003) 361–368

[10] Desobry, F., Davy, M., Fitzgerald, W.: A class of kernels for sets of vectors. In: Proceedings of the ESANN. (2005) 461–466

[11] Jain, A., Dubes, R.: Algorithms for Clustering Data. Prentice Hall, Inc. (1988)

[12] Duda, R., Hart, P., Stork, D.: Pattern Classification. Wiley–Interscience (2000)

[13] Ward, J.: Hierarchical grouping to optimize an objective function. Journal of the American Statistical Association **58** (1963) 236–244

[14] Fred, A., Leitao, J.: Clustering under a hypothesis of smooth dissimilarity increments. In: Proceedings of the ICPR. Volume 2. (2000) 190–194

[15] Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society: Series B **39** (1977) 1–38