A Simple Stochastic Algorithm for Structural Features Learning

Jan Mačák and Ondřej Drbohlav

Center for Machine Perception, Department of Cybernetics
Faculty of Elec. Eng., Czech Technical University in Prague
{macakj1,drbohlav}@cmp.felk.cvut.cz

Abstract. A conceptually very simple unsupervised algorithm for learning structure in the form of a hierarchical probabilistic model is described in this paper. The proposed probabilistic model can easily work with any type of image primitives such as edge segments, non-max-suppressed filter set responses, texels, distinct image regions, SIFT features, etc., and is even capable of modelling non-rigid and/or visually variable objects. The model has recursive form and consists of sets of simple and gradually growing sub-models that are shared and learned individually in layers. The proposed probabilistic framework enables to exactly compute the probability of presence of a certain model, regardless on which layer it actually is. All these learned models constitute a rich set of independent structure elements of variable complexity that can be used as features in various recognition tasks.

1 Introduction

Unsupervised learning of object appearance has been a challenging task since the very beginning of computer vision. There are many approaches to this problem, but considering a huge number of visual categories, low computational requirements, easy extension-ability requirements, the most promising object representations seem to be hierarchic/compositional ones [1, 2]. Focusing on these, there are various hierarchically organized models proposed, taking inspiration from different fields of science. There are nature inspired designs [3, 4], there are models using grammars [5–7], there are very successful approaches using neural networks [8, 9]. Learning strategies of such structures are similarly diverse, ranging from semi-automatic methods when the structure is given by human and only its parameters are learned [5] over sophisticated supervised/unsupervised methods of deep learning [10, 9, 8].

The core question this work shall answer is that of whether it is possible to learn both structure and parameters of a generative compositional probabilistic model using a very simple algorithm based on the *Expectation-Maximization* principle, that means by random initialization and iterative updating. Noticeable difference from the deep learning is that unlike learning deep belief [10] or convolutional neural network [8] this method gives an explicit structure model similar to image grammars and requires less hyper-parameter/design choices – there is actually just one hyper-parameter that needs to be set and that is the maximal allowed portion of non-modelled data. Furthermore, its individual learned compositions can be used as features in more sophisticated classification framework such as SVM or AdaBoost. Similar approach [11] has been recently shown to be very efficient in domain transfer task[1]. This indicates that structural approach has very good generalization capabilities.

2 Concepts

Given a dataset which consists of a set of detected features (denoted $\mathcal{D}$ further on) per image, the task is to construct a set of generative probabilistic models that would be able to generate the dataset. This construction is designed to work in an unsupervised manner. Following the idea of Occam's razor or its modern version represented by the *MDL* approach [12], the set of models is to be as simple as possible. Significant reduction of the model complexity can be achieved by sharing of model parts. This has also other practical side-effect benefits such as computation cost reduction, smaller memory consumption, etc. A natural form of model that allows for immediate and efficient sub-parts sharing is a hierarchical model where the root node (in this paper the root is always at the top) generates a number of children nodes, these children again generate sets of its children and this scheme is repeated until the last (lowest) layer is reached, the lower a node is the smaller its working radius is. The advantage of this form of the model organization also is the fact that it can very naturally model non-rigid data. For example, if the modelled object is a human body, the model can consist of rigid sub-models of individual limbs (more precisely their rigid pieces) on a certain layer and then on higher layers define their spatial relations including the rotations.

In the case that the learning shall proceed without supervision, it is reasonable to start from the most local properties and have the complexity grown while proceeding to higher layers. At each layer a set of *compositions* capable of generating the given data is acquired using random sampling and *Expectation-Maximization* parameter learning. The word *composition* is used to represent a single shareable model (actually on an arbitrary layer) which defines spatial relations between composition's root node and its children nodes.

2.1 Probabilistic framework

The form of the probabilistic model, which would be a member of the set of models representing the data, is shown in the Figure 1. In the illustration, there are actually two *compositions* explicitly shown, however, the nodes b_{12}, b_{13} are similar *compositions*. The model itself is a directed acyclic graph (*DAG*), the *compositions* on lower layers might have multiple parents though – in a sense that a *composition* can be shared by more than one higher layer models. Because the model structure is recursive, it is sufficient to describe only a single

[1] Training and classification of same object classes in different datasets

composition in detail. The following description refers to the model `comp.I.` in the Figure 1. It can be interpreted this way[2]: the node b with probability m_i generates *composition* at a random position c_i. There can be an arbitrary number of such structural components, but the reasonable number is ca 2–8 [13].

Mathematically, the node m_i has two discrete states

$$P(m_i|b) = \begin{cases} p_i \text{ if } m_i = 1 \\ 1 - p_i \text{ if } m_i = 0 \end{cases}, \tag{1}$$

where state $m_1 = 1$ means that i-th component is generated, the state $m_1 = 0$ means that nothing is generated in that branch. The probability model for the position of the generated underlying child *composition* is normal distribution

$$P(c_i|m_i = 1) = \mathcal{N}(\mu_i, \Sigma_i), \tag{2}$$

in the case that nothing is generated, the model branch is terminated with constant probability

$$P(\{\}|m_i = 0) = 1. \tag{3}$$

Except these so called *structural* components, each node is also equipped with ability of generating random patterns. This mechanism is incorporated in the right-most branch with the node e and double-bordered node c_e. The node e is again discreet-state node and its state can be any natural number k meaning that the model generates k independent random patterns – either arbitrary *composition* from lower layer or so called *non-structural random pattern* which basically means that it is a bunch of data that can not be explicitly modelled using the structural model. The probability model of k is a standard *Poisson distribution*

$$P(e|b) = \frac{\lambda^k}{k!} e^{-\lambda}. \tag{4}$$

The reason for the double border of the node c_e is that it denotes that there can actually be a number of such nodes, depending on the state k. However, each such node is of the same form, there is a distribution for its position

$$P(c_e|e) = \mathcal{N}(\mu_e, \Sigma_e) \tag{5}$$

and also the distribution over *compositions* that can be generated plus the *non-structural random pattern* ϵ is

$$P(c^t|c_e) = p^t, \ \sum_{i=1}^{|c|} p^i + p^\epsilon = 1 \tag{6}$$

Due to the fact that this internal state c^t is always marginalized, it is not drawn in the picture explicitly.

[2] For the sake of clarity, the unnecessary indices are ommited.

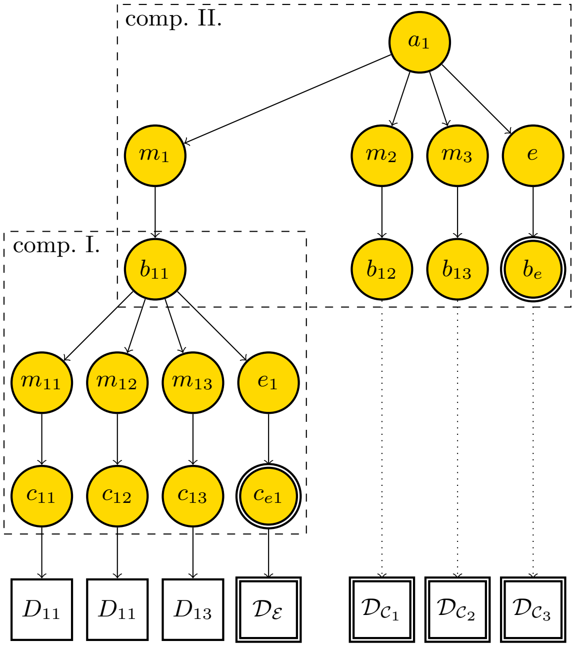


Fig. 1. The structure of the probabilistic model of a composition. The rectangular nodes denoted with letter D are individual input data, the round nodes (a, b, c, m, e) are internal nodes. Nodes b (c) model spatial relations of children and parent, m model the decision if the child is present. Nodes e are discrete-state and model the number of random structure fragments, nodes b_e and c_e model the location of each such fragment. Obviously, being dependent on the actual state of parent e node (non-negative number k), there are k such nodes, which is graphically denoted by double border. Similarly, the double border of the $\mathcal{D}$ nodes indicates that the nodes are actually disjoint sets of input data and for whole data holds $\mathcal{D} = D_{11} \cup D_{12} \cup D_{13} \cup \mathcal{D}_{\mathcal{E}} \cup \mathcal{D}_{\mathcal{C}_1} \cup \mathcal{D}_{\mathcal{C}_2} \cup \mathcal{D}_{\mathcal{C}_3}$. Dashed rectangles delimit exemplar individual compositions.

The marginalized probability for *non-structural random pattern* data generated by node b can be written as

$$P(\mathcal{D}_{\mathcal{E}}|b) = \sum_{k=1}^{\infty} \frac{\lambda^k}{k!} e^{-\lambda} \prod_{i=1}^{k} \int_{c_{e_k}} P(c_{e_k}|e) \sum_{t \in T} P(\mathcal{D}_{\mathcal{E}_i}|c^t) P(c^t|c_{e_k}) dc_{e_k} . \tag{7}$$

The marginalized probability of all data $\mathcal{D}_b$ generated by node b then is

$$P(\mathcal{D}_b|b) = \left(\prod_{i=1}^{3} \int_{c_i} P(D_i|c_i) P(c_i|m_i) P(m_i = 1|b) dc_i + P(m_i = 0|b) \right) P(\mathcal{D}_{\mathcal{E}}|b). \tag{8}$$

When evaluating higher layer *composition*, the formula is recurrent and structurally just the same, so it can be computed easily using the *Message Passing* algorithm [14].

2.2 Inference

As the probabilistic model models the simplest and smallest parts of the data at the lowest layer, it is reasonable to start inference from bottom and proceed up – if no bottom *compositions* are found then there is no reason for continuing in searching for more complex objects, since these are build of those simple ones. Due to the recursive character of the probabilistic model, the mechanism is just the same on each layer and therefore it is sufficient to describe only the transition from the layer $n-1$ to the layer n.

Suppose that the given data is organized into non-overlapping groups $\mathcal{A} = \{A_1, \dots, A_n\}$ and each group contains some instances:

$$\begin{aligned} &\mathcal{D}' = \mathcal{D}_{A_1} \cup \mathcal{D}_{A_2} \cup \dots \cup \mathcal{D}_{A_n} = \\ &= \left\{ {}_1\mathrm{d}_1^1, {}_1\mathrm{d}_2^1, {}_1\mathrm{d}_1^2, \dots, {}_1\mathrm{d}^\epsilon \right\} \cup \left\{ {}_2\mathrm{d}_1^1, {}_2\mathrm{d}_2^1, {}_2\mathrm{d}^\epsilon \right\} \cup \dots \left\{ {}_3\mathrm{d}_1^2, {}_3\mathrm{d}_2^2, {}_3\mathrm{d}_3^2, \dots, {}_3\mathrm{d}^\epsilon \right\} \end{aligned} \tag{9}$$

– the set of instances from the layer $n-1$, each of the known probability $P(\mathcal{D}_i | {}_i\mathrm{d}_k^c)$, where i is the data group index, c indicates which *composition* the instance is of and index k is the number of variant (there can be more than one instance of a *composition*) – and the set of *compositions* $\mathcal{C} = \left\{ c^1, c^2, \dots, c^n, c^\epsilon \right\}$, the task of inference algorithm is to find a set of instances $\mathcal{I} = \left\{ \mathrm{c}_1^1, \mathrm{c}_2^1, \mathrm{c}_1^2, \dots, \mathrm{c}^\epsilon \right\}$ of *compositions* that model the data with reasonably high probability $P(\mathcal{D}' | \mathrm{c}_k^c)$. Such scenario is advantageous from at least two viewpoints: i) if the *composition* instance c_k^c use any instance from each group of data, it is assured that the instance c_k^c models all assigned data, ii) as the data groups are mutually share-free, by choosing precisely one instance from each group, the algorithm can not produce cyclic structure.

Due to the limited maximal complexity of compositions at each layer and the limited receptive field, it is feasible to find the globally best instances by brute-force enumeration of all consistent proposals in each grouping. After inferring instances of the layer n, the grouping of the layer $n-1$ becomes obsolete.

This grouping approach does have a disadvantage, though. It is apparent that the grouping can not be optimal with respect to all candidate *compositions* and consequently the approach produces sub-optimal instance when the grouping is not in favour[3] of that particular *composition*. However, this problem can be overcome by involving a second mechanism – *top-down* optimization of instances of low probabilities caused by missing or dislocated components similar to [15, 16]. This *top-down* mechanism can either search for more suitable already existing instances or can come up with completely new instances. It is also capable of changing the layer n grouping when it is beneficial.

2.3 Learning

Besides the probability model already introduced, the core of this work is the learning method. First, it is necessary to return to the *grouping* mentioned in previous section, these groups are referred as *area* from now on. To give more

[3] The grouping is actually generated randomly.

```
1) get layer 1 instances from an image  for $i \leftarrow 1$ to $N$ do
    2) find random grouping of instances of previous layer;
    3) infer instances of current layer (bottom-up process);
    4) merge the partitioning in previous layer;
    5) improve the instances - find missing parts (top-down process);
end
```

Algorithm 1: The sketch of the inference algorithm.

precise description, an *area* is an artificial container which temporarily owns a subset of instances of *compositions* from the lower layer (see Eq. 9). As a consequence, an *area* always represents a well-defined subset of data, see the Figure 2 for an illustrative example. All inferred instances (the set $\mathcal{I}$) in the *area* has to have assigned all the content of the *area* – either as a part of the structural model or as a part of *non-structural random pattern*. This makes the inferred instances comparable and allows for computing the posterior probabilities of individual instances given the *area* content using the Bayes formula

$$P(\mathrm{c}_k^c|\mathcal{D}') = \frac{P(\mathcal{D}'|\mathrm{c}_k^c)P(c_c)}{\sum_{c,k} P(\mathcal{D}'|\mathrm{c}_k^c)P(c_c)}. \tag{10}$$

The learning itself is iterative and uses these probabilities $P(\mathrm{c}_k^c|\mathcal{D}')$ as weights for computing updated values of model parameters. First, the data is randomly partitioned into *areas* of pre-set size. Then a *composition* is randomly created by sampling from one of the *areas*. The inference algorithm over the whole data (all areas) is run as to get all instances of the *composition*. These instances are then used for update of the *composition* parameters in a *Maximum-likelihood* (further referred as *ML*) manner. The model branches are conditionally independent and therefore can be optimized separately. It can be shown that for example a *ML* estimate of any of *composition*'s component position is

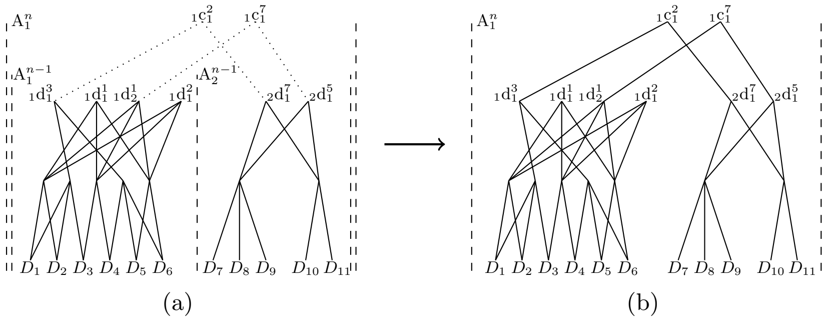
$$\mu_{\text{comp}} = \frac{\sum_{\mathcal{A},k} \mu_k P(b_k^c|\mathcal{D}_{\mathcal{A}})}{\sum_{\mathcal{A},k} P(b_k^c|\mathcal{D}_{\mathcal{A}})}, \tag{11}$$

where $\mathcal{A}$ is the set of all areas and k is the index of the set of instances of the *composition* c, and analogically for the other parameters. Only the parameter λ is learned differently. The *ML* estimate for this parameter of the *Poisson distribution* is the mean of the modelled values. In this case, it is the mean value of the number of lower layer *areas* contained in the current *areas*. This value is used directly as λ_ϵ, for each *composition* is the parameter lambda different and it is set as

$$\lambda^c = \max(\lambda_{\min}, \lambda_\epsilon - N), \tag{12}$$

where N is the number of components in the *composition*. This reflects the intent that each *composition* generates on average a similar amount of data.

Besides the *ML* updating the parameters of the already assigned *composition* components, also the positions and types of neighbouring non-explained



(a) (b)

Fig. 2. An illustration of groups and their life cycle. The (a) shows the situation before inference of new instances on the layer n, the (b) shows the same situation after inference. Noticeable changes are: i) two new instances, ${}_1\mathrm{c}_1^2$,${}_1\mathrm{c}_1^7$, were created, each of them is built of different lower layer instances (${}_1\mathrm{d}_1^3$,${}_2\mathrm{d}_1^7$) vs. (${}_1\mathrm{d}_2^1$,${}_2\mathrm{d}_1^5$), ii) partitioning on the layer $n-1$ became ineffective and iii) both new layer n instances model all underlying data.

instances are tracked. When a significant cluster is discovered, it can be added to *composition* as a new component. Analogously, if any of the components prove to be useless, it can be removed. These mechanisms enable *composition* to travel within the configuration space towards at least locally most stable and frequent form.

In the second and further iterations, the *compositions* are updated and one new random *composition* is always added, unless the fraction of non-modelled data drops below a given treshold. After this event has happened, the algorithm keeps running for a predefined number of iteratiors and in each iteration one new spare *composition* is sampled and if turns out to be more useful, it replaces the least useful *composition* from the final set. The estimated prior probability of a *composition* is taken as the usefulness measure. This can be viewed as a simple restarting scheme in order not to end up in the first local extrema. When the learning of one layer finishes, the final set of *compositions* $\mathcal{C}_{\text{final}}$ is selected as the N highest probability models following the condition

$$\sum_{c \in \mathcal{C}_{\text{final}}} p^c \geq T \tag{13}$$

and the learning proceeds to higher layer and ends when no new layer can be built. This happens when there is a single area in each piece of dataset (i.e. in an image), because nothing can be learned from such data.

When sampling an *area* to be taken as an initial *composition*, the candidates are weighted according to the probability $P(b^e|\mathcal{D}_{\mathcal{A}})$ – that is by the probability that the *area* is not well modelled by any *composition*. By this, the algorithm softly focuses on yet not-modelled data.

3 Experiments

3.1 Learning the alphabet

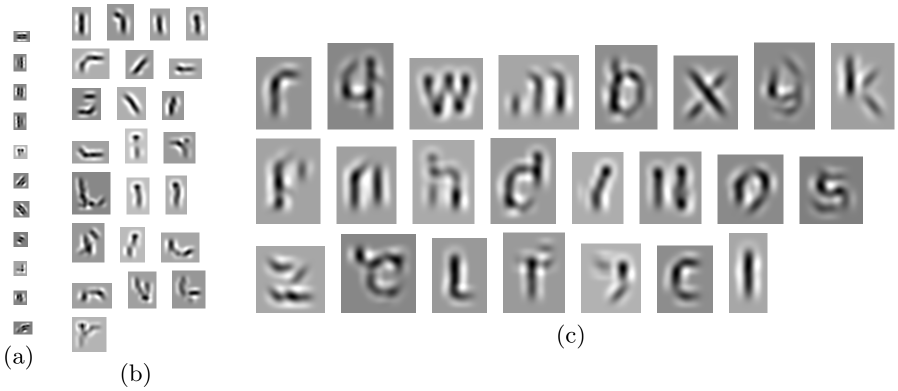


Fig. 3. The complete library of *compositions* that has been learned on the alphabet letters data. The library consists of three compositional layers in this case. The (a) shows the first compositional layer, the (b) middle layer and the (c) shows the top layer. All layers are plotted in the same scale, the apparent growing fuzziness is due to the increasing uncertainty of positions of components.

The functionality of the learning algorithm was studied on a simple yet interesting dataset of rendered letters. This dataset consisted of 26 black and white images of small alphabet letters, where each letter was present exactly once. The noticeable fact is, that some letters differ from each other by just a tiny part and while a human eye is very sensitive to these small differences, the proposed method does not have any information on the meaning or importance of individual parts of the content of images and works completely unsupervised. This observation indicates that alphabet letters exhibit a high degree of sharing of shape segments - there are actually only straight vertical, horizontal or slanted lines and arcs, so the learned library of *composition* shall be rather small. But it can not be too small, because if there is a strong stress on obtaining an as compact representation as possible, there is also an imminent danger of losing discriminative accuracy. Naturally, the smaller the library is the less discriminative the learned model is – meaning that the model is definitely not capable of telling some letters apart. If this is to happen on the alphabet dataset, the most expected candidates are the letters 'l', 'i' and possibly 'j'. Practical challenging aspect was the fact that statistical approaches like random partitioning might not work with so small number of images.

The data entering into the learning are not preprocessed in any way, except for finding the edges and their orientations and random selection of edgels reasonably distant from each other. The edgels orientations are discretized into

8 categories representing orientations of 0, 45, 90, 135, 180, 225, 270 and 315 degrees.

The output of the learning is shown in the Figure 3, learned *compositions* are shown in groups per layers. As can be seen, all letters do have a model and the result also follows the expectation that there might be some letters sharing one model. It happened for the letters 'i', 'j' and 'l' which are all represented as a vertical line segment. The Figure 4 shows an example of how the data is decomposed and hierarchically organized.

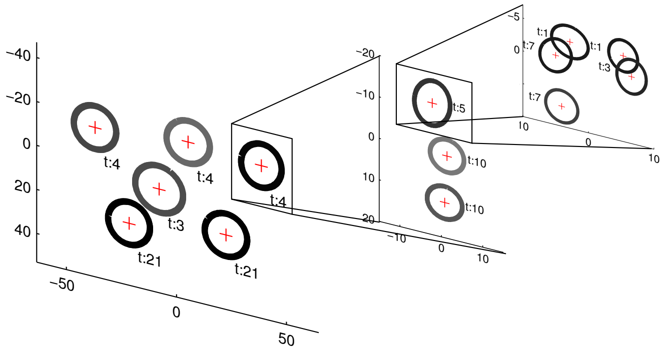
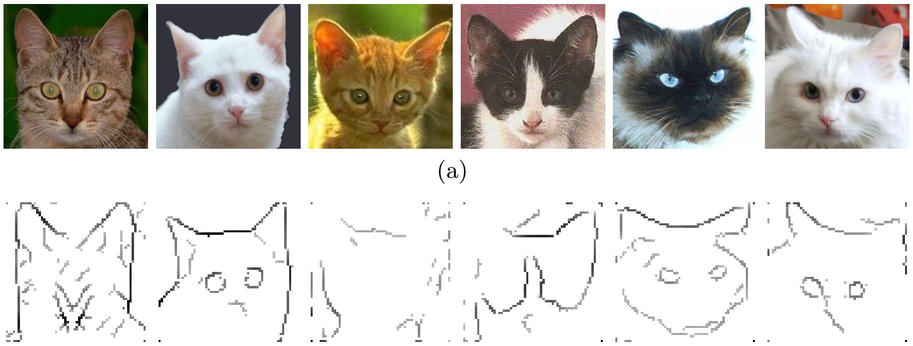


Fig. 4. A detailed view of one branch of the learned structure for the letter 'w', the ellipses show the covariance matrix encoding the position uncertainty of individual components, the gray-level of the ellipses depicts the probability that the corresponding component is present/generated.

3.2 Learning cat faces

Second experiment had a different scenario, it was done on a single category set of real images of cat faces from the LHI-Animal-Faces dataset [17]. The subset of cat faces consisted of 89 images roughly on a same scale.A small selection of images from this set is shown in the Figure 5(a).The images were preprocessed before entering the learning algorithm in order to extract oriented edge segments. This was done by convolving the image with a four-item set of *Gabor filters* and taking edges above a threshold of ($T = 0.25 \cdot \max_{x,y} I(x, y)$ – individual for each image). Also *nonmax-suppression* was applied to sparsify the data. The final edge maps for a few images is shown in the Figure 5(b). A dense edge segment set was acquired of this data and was used for learning. In fact, any other suitable edge/texture kernel or image descriptor could be used instead of the edge segments.

The learning algorithm succeeded and found a three-layered set of *compositions* capable of modelling a cat face which can be seen in the Figure 6 showing a



(a)

(b)

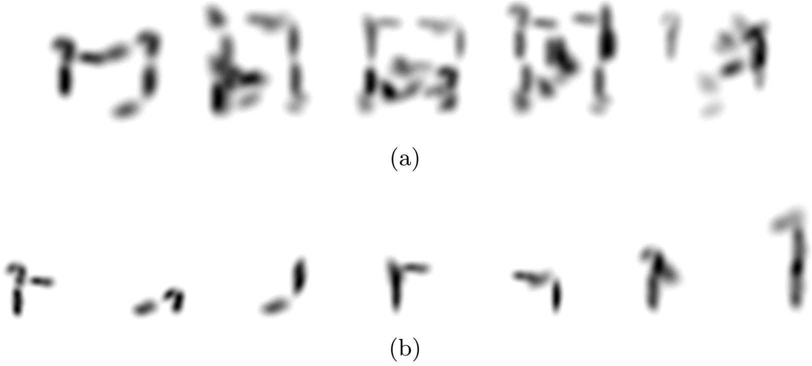
Fig. 5. An illustrative selection of the input data for the second experiment. In the (a) there are original pictures shown, what the learning algorithm really sees is shown in the (b).

few selected learned *compositions*. Looking at the over-all statistics of the second layer, there are 52 models of which about a 1/3 are models for ears, there is also a model for pair of eyes and some face fractions. On the top layer, there are 26 models of which about eight represent full faces, there are some non-complete faces and there are also a few models covering the 'random' fur texture patterns. This results indicate that the proposed probabilistic model would benefit from higher expresivity in a sense of allowing multiple mutually exclusive components in the individual *branches* – an analogy to the *OR* nodes in [5] – that would merge some *compositions* together yielding a more compact model, but at the risk of losing discriminability. This can be achieved by rather simple modification of the m node of the model, see Figure 1, more specifically, by changing it from two-state to multiple-state with appropriate probabilities.

4 Summary

In this paper, a very simple stochastic algorithm for unsupervised joint learning of structure and parameters is described. The probabilistic model is generative and its structure is compositional. The learning of the model is done gradually per individual layers of compositions exploiting the *maximum-likelihood* principle and *expectation-maximization*. The produced learned compositions can be seen as structure elements and can be used as features in various computer vision tasks.

The functionality is demonstrated on two experiments. The first was learning a compositional representation of alphabet letters – a compact representative of a dataset which exhibit both compositionality and presence of multiple categories. The algorithm succeeded in both aspects. The second experiment was on a single category dataset of cat faces with significant within-category visual diversity.



(a)

(b)

Fig. 6. Some of the learned cat faces are shown in the (a). In the (b), there is a selection of lower layer *compositions* plotted. The first three models from the left are the building blocks of the left-most head in the (a).

Acknowledgement. The authors were supported by the Czech Science Foundation under the Project P103/12/1578.

References

1. Tsotsos, J.K.: Analyzing vision at the complexity level. Behavioral and Brain Sciences **13** (1990) 423–445
2. Bienenstock, E., Geman, S., Potter, D.: Compositionality, MDL priors, and object recognition. In Mozer, M., Jordan, M.I., Petsche, T., eds.: NIPS, MIT Press (1996) 838–844
3. Fukushima, K.: Neocognitron: A hierarchical neural network capable of visual pattern recognition. Neural Networks **1** (1988) 119 – 130
4. Serre, T., Wolf, L., Bileschi, S., Riesenhuber, M., Poggio, T.: Robust object recognition with cortex-like mechanisms. IEEE Trans. Pattern Anal. Mach. Intell. **29** (2007) 411–426
5. Zhu, S.C., Mumford, D.: A stochastic grammar of images. Found. Trends. Comput. Graph. Vis. **2** (2006) 259–362
6. Zhu, L., Chen, Y., Yuille, A.L.: Learning a hierarchical deformable template for rapid deformable object parsing. IEEE Trans. Pattern Anal. Mach. Intell. **32** (2010) 1029–1043
7. Si, Z., Zhu, S.C.: Learning and-or templates for object recognition and detection. IEEE Transactions on Pattern Analysis and Machine Intelligence **35** (2013) 2189–2205
8. Lee, H., Grosse, R., Ranganath, R., Ng, A.Y.: Unsupervised learning of hierarchical representations with convolutional deep belief networks. Commun. ACM **54** (2011) 95–103
9. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C., Bottou, L., Weinberger, K., eds.: Advances in Neural Information Processing Systems 25. Curran Associates, Inc. (2012) 1097–1105

10. Hinton, G.E., Osindero, S.: A fast learning algorithm for deep belief nets. Neural Computation **18** (2006) 2006
11. Dai, J., Hong, Y., Hu, W., Zhu, S.C., Wu, Y.N.: Unsupervised learning of dictionaries of hierarchical compositional models. In: Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). (2014)
12. Grünwald, P.D.: The Minimum Description Length Principle. MIT Press (2007)
13. Fidler, S., Berginc, G., Leonardis, A.: Hierarchical statistical learning of generic parts of object structure. In: Proc. CVPR. (2006) 182–189
14. Pearl, J.: Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1988)
15. Fidler, S., Leonardis, A.: Towards scalable representations of object categories: Learning a hierarchy of parts. In: Proc. CVPR. (2007)
16. Zhu, L., Lin, C., Huang, H., Chen, Y., Yuille, A.L.: Unsupervised structure learning: Hierarchical recursive composition, suspicious coincidence and competitive exclusion. In Forsyth, D.A., Torr, P.H.S., Zisserman, A., eds.: ECCV (2). Volume 5303 of Lecture Notes in Computer Science., Springer (2008) 759–773
17. Si, Z., Zhu, S.C.: Learning hybrid image templates (HIT) by information projection. IEEE Transactions on Pattern Analysis and Machine Intelligence **34** (2012) 1354–1367