Submitted to the Annals of Statistics
arXiv: arXiv:0000.0000

A THEORY OF GENERATIVE CONVNET

BY JIANWEN XIE*, YANG LU*, SONG-CHUN ZHU AND YING NIAN WU

University of California, Los Angeles

The convolutional neural network (ConvNet or CNN) is a powerful discriminative learning machine. In this paper, we show that a generative random field model that we call generative ConvNet can be derived from the discriminative ConvNet. The probability distribution of the generative ConvNet model is in the form of exponential tilting of a reference distribution. Assuming rectified linear units and Gaussian white noise reference distribution, we show that the generative ConvNet model contains a representational structure with multiple layers of binary activation variables. The model is piecewise Gaussian, where each piece is determined by the binary activation variables, which reconstruct the mean of the Gaussian piece. The Langevin dynamics for synthesis is driven by the reconstruction error, and the corresponding gradient descent dynamics converges to a local energy minimum that is auto-encoding. As for learning, we show that the contrastive divergence learning tends to reconstruct the observed images. Finally, we show that the maximum likelihood learning algorithm can generate realistic natural images.

1. Introduction.

1.1. *Recent development.* Fueled by the big datasets such as ImageNet [5] and improved computer power brought by the graphical processing units (GPUs), the convolutional neural network (ConvNet or CNN) [16, 15] has recently become the most successful discriminative or predictive learning machine.

The turning event for the resurgence of the ConvNet was its resounding victory in a competition on the ImageNet dataset [5] in 2012. The ImageNet dataset was first released in 2009. Starting from 2010, there has been an annual competition on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [28]. One of the tasks is image classification, which is to assign each image to an object category. There are roughly 1.2 million training images, 50,000 validation images, and 100,000 testing images, from 1,000 object categories. In ILSVRC 2012, the ConvNet [15] became the runaway winner of the image classification competition. The winning network has 60 million parameters and 650,000 hidden nodes. It consists of 5 convolutional layers (some of them are followed by sub-sampling and max-pooling layers) and 3 fully-connected layers. Since then, the ConvNet and related

*Equal contributions.

MSC 2010 subject classifications: Primary 62A01, 62A01; Secondary 62A01

Keywords and phrases: Auto-encoder, Generative models, Langevin dynamics, Markov random fields

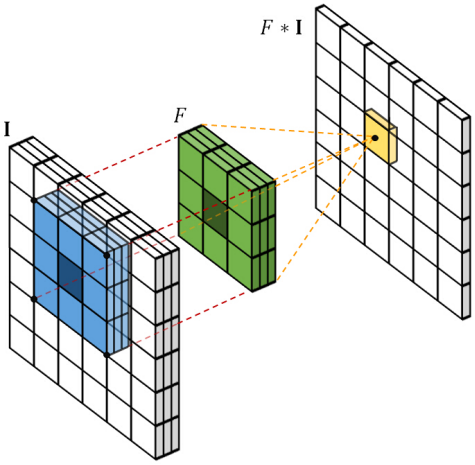

Fig 1: Filtering or convolution: applying a filter of the size $3 \times 3 \times 3$ on an image of the size $6 \times 6 \times 3$ to get a filtered image or feature map of 6×6 (with proper boundary handling). Each pixel of the filtered image is computed by the weighted sum of the $3 \times 3 \times 3$ pixels of the input image centered at this pixel. There are 3 color channels (R, G, B), so both the input image and the filter are three-dimensional.

deep learning methods have been adopted for many tasks in artificial intelligence, such as those in computer vision, speech recognition, natural language processing, etc., and have achieved state of the art performances, sometimes super-human performances, on these tasks.

1.2. *ConvNet as unfolded GLM.* For statisticians, a ConvNet can be considered an unfolded version of the generalized linear model (GLM). A GLM is characterized by a weighted sum of input variables followed by a one-dimensional non-linear link function. A ConvNet, often applied to image, video or speech data, unfolds the GLM structure along two directions. (1) Convolutional: the weighted sum is computed locally around each pixel of the image, mapping an input image to an output image called the filtered image or the feature map. The operation is called convolution or linear filtering. See Fig. 1 for an illustration of linear filtering. (2) Hierarchical: there are multiple layers of linear filtering and element-wise non-linear transformation, as well as sub-sampling that makes the filtered images smaller. After a number of layers, the feature maps are reduced to 1×1 due to repeated sub-sampling. The final layer of features are then used for classification via a multinomial logistic regression. See Fig. 2 for an illustration.

The convolutional and hierarchical unfolding of the GLM gives rise to a ConvNet structure that is both simple and rich. It defines a rich class of functions that

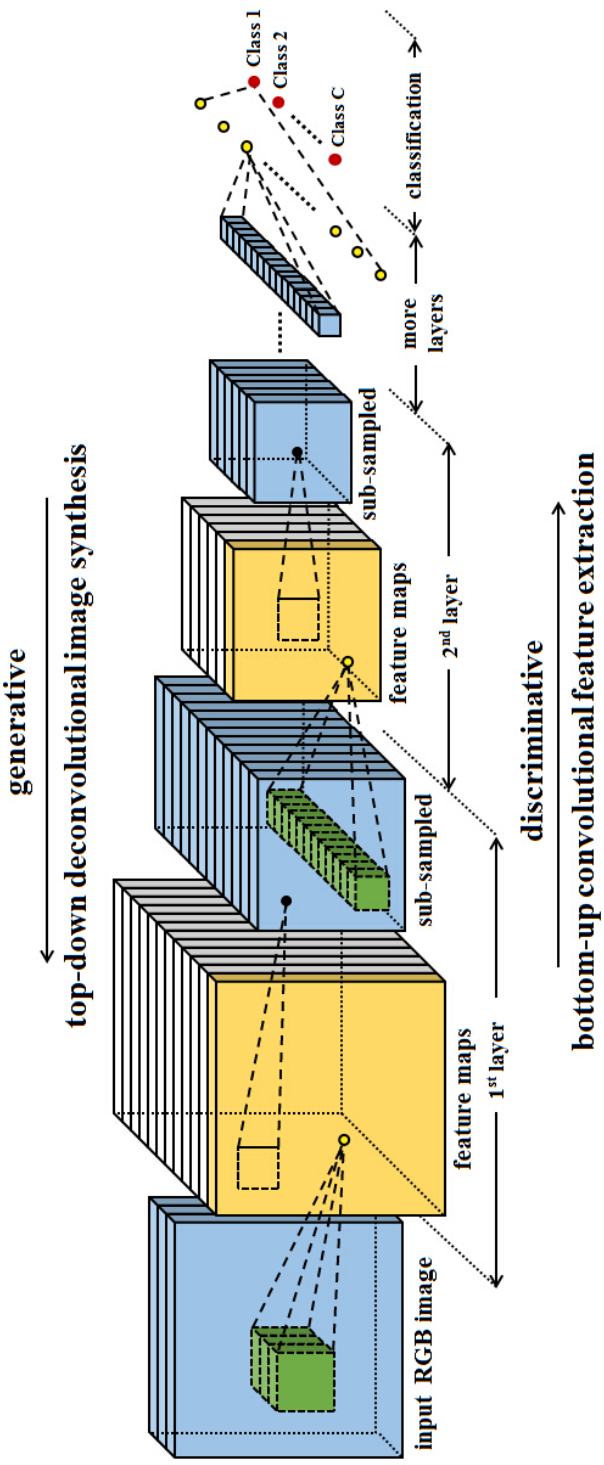

Fig 2: Convolutional neural networks consist of multiple layers of filtering and sub-sampling operations for bottom-up feature extraction, resulting in multiple layers of feature maps and their sub-sampled versions. The top layer features are used for classification via multinomial logistic regression. The discriminative direction is from image to category, whereas the generative direction is from category to image.

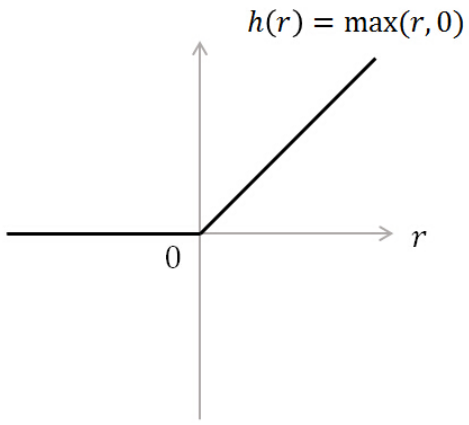

Fig 3: Rectified Linear Unit (ReLU).

map an image to an object category. The functions are parametrized by the multiple layers of weight and bias parameters, which correspond to the coefficient and intercept parameters of GLMs. The element-wise non-linear transformation corresponds to the link function of GLM. In the modern ConvNet, the non-linear transformation usually takes the form of Rectified Linear Unit (ReLU) as illustrated by Fig. 3. See Section 2 for a detailed technical account of the ConvNet.

1.3. *Discriminative vs generative.* Just like GLM or logistic regression, the ConvNet is a discriminative or predictive learning machine. The input to the ConvNet is an image. The output is an object category. The discriminative direction is from image to object category. Such a direction is often called the bottom-up direction.

The ConvNet tells us *how* to discriminate between, say, a bird and a cat. It does not tell us *what* a bird looks like or *what* a cat looks like. Such knowledge is most naturally represented by the generative direction, which is from object category to image. This direction is often called the top-down direction. The generative direction can be mathematically defined by a probability distribution on the image space, or a random field model. We can learn such a statistical model from training images. If we sample from the learned model, we can generate or synthesize new images. We may intuitively consider the sampling process as a matter of imagination, dreaming, or fantasizing, which is a gift that is obviously possessed by a human brain.

Despite the successes of the discriminative learning machines such as ConvNet, the progress on developing generative models is still lagging behind. In this article, we show that we can turn the discriminative ConvNet into a generative ConvNet

Fig 4: Generating texture patterns. The first image is the training image, and the rest are 2 of the images generated by the learning algorithm.

Fig 5: Generating object patterns. The first row displays 4 of the training images, and the second row displays 4 of the images generated by the learning algorithm.

model. We also show that the generative ConvNet can indeed generate surprisingly realistic image patterns. Figs. 4 and 5 display two examples. In Fig. 4, the first image is a single observed training image. The rest are 2 of the images sampled from the learned generative ConvNet model. In Fig. 5, the first row displays 4 of the 7 training images. The second row displays 4 of the images sampled from the learned model. Intuitively, the learned models tell us what ivy leaves look like and what an egret looks like.

While the discriminative ConvNet can be considered an unfolded version of GLM, the generative ConvNet is an elaborate Markov random field (MRF) or equivalently a Gibbs distribution [2], with its potential function defined by the ConvNet features.

1.4. *Motivation for generative ConvNet.* Statisticians sometimes complain that the ConvNet is a "black-box." Although such a criticism is not fair, it does convey a desire to make the ConvNet more transparent and interpretable. This goal can be partially achieved by turning the discriminative ConvNet into a generative model with an explicit representation of the image. The generative and representational

perspective is more appealing than the discriminative perspective to statisticians who care more about explaining the data than predicting the categories, especially because natural images contain such a rich variety of patterns. We can make the model more interpretable by sparsifying the parameters, which can be naturally accomplished within a generative and representational framework [36].

Developing generative models and representations is not only important for making the model understandable, it is also of fundamental importance for unsupervised learning, where we are only given unlabeled images without knowing their object categories, because the labeled images may be scarce and expensive to obtain. The generative models enable us to learn the parameters by explaining the image data instead of predicting the object categories.

With the success of the discriminative ConvNet, researchers in deep learning are still searching for deep generative models and unsupervised learning machines. Our endeavor of developing a generative version of the ConvNet is conceptually satisfying because it shows that in searching for deep generative models, we need to look no further beyond the ConvNet. Our work expands the scope of the CovnNet and connects the ConvNet to various important concepts and methods in machine learning. It leads to a unified framework of ConvNet that encompasses both the discriminative classifier and the generative model, and both supervised learning and unsupervised learning.

1.5. *Our results on generative ConvNet.* The probability distribution of the generative ConvNet model is in the form of an exponential tilting of a reference distribution, and the exponential tilting is defined by the ConvNet that involves multiple layers of liner filtering and non-linear transformation. The generative ConvNet model can be viewed as a hierarchical version of the FRAME (Filters, Random field, And Maximum Entropy) model [43], as well as the Product of Experts (PoE) [6] and Field of Experts (FoE) [27] models.

Being of the form of an exponential tilting model, the generative ConvNet may appear dull and opaque. The main purpose of this article is to show that the contrary is true. Assuming Gaussian white noise reference distribution and ReLU nonlinearity in Fig 3, we discover that the generative ConvNet contains a surprisingly explicit and exquisite representational structure. Specifically, it contains multiple layers of binary activation variables that indicate the presence or absence of the patterns modeled by the multiple layers of filters of the ConvNet. The generative ConvNet model is non-Gaussian, or more precisely, piecewise Gaussian, where each piece is determined by the binary activation variables. These binary variables are computed by a bottom-up process by the multiple layers of filters, and they reconstruct the mean of the Gaussian piece by a top-down process, where the multiple layers of filters serve as multiple layers of basis functions for image reconstruction.

The Langevin dynamics [20] can be employed to synthesize images by sampling from the generative ConvNet. Interestingly, the dynamics is driven by the reconstruction error, i.e., the difference between the current image and the reconstruction by the binary activation variables mentioned above. Thus image synthesis and image reconstruction are connected.

The deterministic gradient descent counterpart of the Langevin dynamics was employed by [41] for exploring the local energy minima of the FRAME model. They called it the Gibbs Reaction And Diffusion Equation (GRADE). It defines a dynamics that converges to a local energy minimum. The local energy minima are the means of the Gaussian pieces mentioned above, and they are auto-encoding via the aforementioned binary activation variables. Thus the generative ConvNet hides an auto-encoder at its energy minima. This observation establishes a connection between the Hopfield network for memory [11] and the auto-encoder.

The model can be learned by maximum likelihood or a simplified variation called the contrastive divergence [6]. For generative ConvNet, we show that the contrastive divergence tends to reconstruct the training images by the above mentioned auto-encoder.

Finally, we show that the maximum likelihood learning algorithm can generate a wide variety of realistic natural image patterns, such as those in Figs 4 and 5, thus validating the generative capacity of the generative ConvNet.

1.6. *Contributions and related work.* The following are the discoveries that we have made in this paper about the generative ConvNet. (1) It can be derived from the discriminative ConvNet. (2) It contains an explicit representational structure. (3) It is piecewise Gaussian. (4) It can be sampled by a reconstruction driven algorithm. (5) Its local energy minima are auto-encoding. (6) The contrastive divergence learning of it tends to reconstruct the observed images. (7) It is capable of generating realistic image patterns. Our work on the generative ConvNet may pave the way for unsupervised learning of ConvNet from large unlabeled datasets, which are essentially in unlimited supply without any cost for human annotation.

The model in the form of exponential tilting of a reference distribution where the exponential tilting is defined by ConvNet was first proposed by [4]. They learned the model by a non-parametric importance sampling scheme. [21] proposed to learn the FRAME models based on pre-learned filters of the existing ConvNets. They did not learn the models from scratch. The hierarchical energy-based models [17] were studied by the pioneering work of [9] and [24]. However, their models do not correspond directly to the modern ConvNet.

Compared to the above mentioned papers, we would like to emphasize the conceptual novelty of our work. Starting from a prototype model and then unfolding it, our work reveals a curious representational structure contained in the model that

involves multiple layers of activation variables. Such a representational structure is unexpected for the exponential family models, and was not studied by the papers cited above.

A main motivation for this paper is to reconcile the FRAME model [43], where the Gabor wavelets play the role of bottom-up filters, and the Olshausen-Field model [25], where the wavelets play the role of top-down basis functions, and unfold these models into a hierarchical sparse compositional model [10, 36]. The generative ConvNet may help solve this problem.

The representational structure in the generative ConvNet is similar to but subtly different from the deconvolution network of [40]. The top-down process of the generative ConvNet is controlled by multiple layers of binary activation variables computed by the bottom-up process. The generative ConvNet can synthesize new images in addition to reconstructing observed images.

Compared to the hierarchical models with explicit binary latent variables such as those based on the Boltzmann machine [7, 29, 18], the generative ConvNet is directly derived from the discriminative ConvNet. Our work seems to suggest that in searching for generative models and unsupervised learning machines, we need to look no further beyond the ConvNet.

There are two major classes of generative models. One consists of exponential family models such as the FRAME model, and the other consists of latent variable models such as the Olshausen-Field model. While the former class usually cannot reconstruct the observed data, the latter class typically needs to negotiate with the intractable inference. The generative ConvNet has both explicit bottom-up pass for computing binary variables and explicit top-down pass for reconstructing the image. It strikes a middle ground between the two classes of models.

One way to get around the intractable inference problem mentioned above is to use the wake-sleep algorithm [8] or the variational auto-encoder [14, 26, 22]. The parameters in the top-down generation model and the bottom-up recognition model of a variational auto-encoder are completely separate from each other. In generative ConvNet, there is a common set of parameters. More importantly, the generative ConvNet actually contains an auto-encoder at the local minima of its energy landscape, and the encoding and decoding of this auto-encoder share the same set of parameters.

1.7. *Long standing unsolved issues.* The following are the long standing issues that have not been resolved for ConvNet. (1) A ConvNet usually has a large number of parameters. The number of parameters is usually much larger than the number of observations. For such a large number of parameters, we can only afford the gradient-based learning algorithm, where the gradient is computed by back-propagation. (2) The objective function such as the log-likelihood function for

training the ConvNet is highly non-convex, with a huge number of local modes. The gradient-based learning algorithm can only be expected to get close to a local mode. It has been observed that different local modes often lead to comparable performances. It is also believed that getting close to a local mode instead of a global mode may actually prevent over-fitting [3]. However, we still do not understand the statistical properties of these local modes. (3) For generative models, we usually need to use Markov chain Monte Carlo (MCMC) [20, 19] to generate images from the models, but the convergence properties of the MCMC algorithms are unknown.

We do not pretend to solve the above issues in this paper. We believe statisticians have much to contribute to understanding these issues.

1.8. *Plan for the rest of the paper.* Section 2 reviews the discriminative ConvNet. Sections 3 and 4 derive and explain the generative ConvNet. Sections 5 and 6 elucidate the representational structure of the generative ConvNet. Section 7 studies methods for learning the generative ConvNet. Section 8 presents some experiments on image generation and reconstruction. Sections 9 and 10 conclude with a brief discussion.

2. Discriminative ConvNet as unfolded GLM. The generalized linear model (GLM) has two main components: (1) A weighted sum of input variables. (2) A non-linear link function. The discriminative ConvNet unfolds the GLM for analyzing image data (or video, speech data etc.)

To fix notation, let $\mathbf{I}(x)$ be an image defined on the square (or rectangular) image domain $\mathcal{D}$, where $x = (x_1, x_2)$ indexes the coordinates of pixels. We can treat $\mathbf{I}(x)$ as a two-dimensional function defined on $\mathcal{D}$. We can also treat $\mathbf{I}$ as a vector if we fix an ordering for the pixels. For a filter F, let $F * \mathbf{I}$ denote the filtered image or feature map, and let $[F * \mathbf{I}](x)$ denote the filter response or feature at position x.

A ConvNet is a composition of multiple layers of linear filtering and element-wise non-linear transformation as expressed by the following recursive formula:

$$[F_k^{(l)} * \mathbf{I}](x) = h\left(\sum_{i=1}^{N_{l-1}} \sum_{y \in \mathcal{S}_l} w_{i,y}^{(l,k)} [F_i^{(l-1)} * \mathbf{I}](x+y) + b_{l,k}\right), \tag{2.1}$$

where $l \in \{1, 2, ..., \mathcal{L}\}$ indexes the layer. $\{F_k^{(l)}, k = 1, ..., N_l\}$ are the filters at layer l, and $\{F_i^{(l-1)}, i = 1, ..., N_{l-1}\}$ are the filters at layer $l-1$. k and i are used to index filters at layers l and $l-1$ respectively, and N_l and N_{l-1} are the numbers of filters at layers l and $l-1$ respectively. The filters are locally supported, so the range of y is within a local support $\mathcal{S}_l$ (such as a 7×7 image patch). The weight parameters $(w_{i,y}^{(l,k)}, y \in \mathcal{S}_l, i = 1, ..., N_{l-1})$ defines a linear filter that operates on

$(F_i^{(l-1)} * \mathbf{I}, i = 1, ..., N_{l-1})$. The linear filtering operation is followed by a non-linear transformation $h()$. At the bottom layer, $[F_k^{(0)} * \mathbf{I}](x) = \mathbf{I}_k(x)$, where $k \in \{\mathrm{R}, \mathrm{G}, \mathrm{B}\}$ indexes the three color channels. Sub-sampling may be implemented so that in $[F_k^{(l)} * \mathbf{I}](x)$, $x \in \mathcal{D}_l \subset \mathcal{D}$. For notational simplicity, we do not make local max pooling explicit in (2.1).

See Fig. 2 for an illustration. The input image at the bottom layer has 3 color channels. The linear filtering operation at the bottom layer is illustrated by Fig. 1. At each subsequent layer, the yellow squares illustrate the filtered images or feature maps $\{F_k^{(l)} * \mathbf{I}, k = 1, ..., N_l\}$, and the blue squares illustrate their sub-sampled versions. There are multiple filtered images or feature maps $\{F_k^{(l)} * \mathbf{I}, k = 1, ..., N_l\}$ produced by the multiple filters $\{F_k^{(l)}, k = 1, ..., N_l\}$ at each layer l. Just as the input image has 3 color channels, the multiple filtered images at each layer are also called multiple channels, where each filter $F_k^{(l)}$ corresponds to a channel k at layer l. Each filter $F_k^{(l)}$ is illustrated by a set of green images. The filter response $[F_k^{(l)} * \mathbf{I}](x)$ is a local weighted sum of pixel values around each pixel x of the filtered images $F_i^{(l-1)} * \mathbf{I}$ at the layer below. The green squares illustrate the range of the local weighted sum, where the summation is also conducted across all the channels or all the filtered images. The filter response $[F_k^{(l)} * \mathbf{I}](x)$ is also called a feature, a node, or a unit at layer l.

We take $h(r) = \max(r, 0)$, the Rectified Linear Unit (ReLU) that is commonly adopted in the modern ConvNet [15]. See Fig. 3 for an illustration. This crisp piecewise linear transformation is the root of the binary activation variables and piecewise Gaussian form of the model. But some results in this paper can be extended to more general non-linearity.

Compared to the GLM, the weight parameters $(w_{i,y}^{(l,k)})$ and the bias term $b_{l,k}$ corresponds to the coefficient and intercept parameters of the GLM, and the non-linear transformation $h(r)$ corresponds to the link function. The weighted sum takes place around each location x and over multiple layers l, thus the ConvNet can be viewed as an unfolded GLM.

Let $(F_k^{(\mathcal{L})})$ be the top layer filters. The filtered images are usually 1×1 due to repeated sub-sampling. Suppose there are C categories. For category $c \in \{1, ..., C\}$, the scoring function for classification is

$$f_c(\mathbf{I}; w) = \sum_{k=1}^{N_{\mathcal{L}}} w_{c,k}[F_k^{(\mathcal{L})} * \mathbf{I}], \tag{2.2}$$

where $w_{c,k}$ are the category-specific weight parameters for classification.

DEFINITION 1. *Discriminative ConvNet: We define the following conditional*

distribution as the discriminative ConvNet:

$$p(c|\mathbf{I}; w) = \frac{\exp[f_c(\mathbf{I}; w) + b_c]}{\sum_{c=1}^{C} \exp[f_c(\mathbf{I}; w) + b_c]}. \tag{2.3}$$

where b_c is the bias term, and w collects all the weight and bias parameters at all the layers.

The discriminative ConvNet is a multinomial logistic regression (or soft-max) that is commonly used for classification [16, 15].

3. Deriving generative ConvNet from discriminative ConvNet. We shall first define the generative ConvNet and then show that it can be derived from the discriminative ConvNet.

DEFINITION 2. *Generative ConvNet (fully connected version): We define the following random field model as the fully connected version of the generative ConvNet:*

$$p(\mathbf{I}|c; w) = p_c(\mathbf{I}; w) = \frac{1}{Z_c(w)} \exp[f_c(\mathbf{I}; w)]q(\mathbf{I}), \tag{3.1}$$

where $q(\mathbf{I})$ is a reference distribution or the null model, assumed to be Gaussian white noise in this paper. $Z(w) = \mathrm{E}_q\{\exp[f_c(\mathbf{I}; w)]\}$ is the normalizing constant.

In (3.1), $p_c(\mathbf{I}; w)$ is obtained by the exponential tilting of q, and is the conditional distribution of image given category, $p(\mathbf{I}|c, w)$. The model was first proposed by [4].

PROPOSITION 1. *Generative and discriminative ConvNets can be derived from each other:*

(a) Let ρ_c be the prior probability of category c, if $p(\mathbf{I}|c; w) = p_c(\mathbf{I}; w)$ is defined according to model (3.1), then $p(c|\mathbf{I}; w)$ is given by model (2.3), with $b_c = \log \rho_c - \log Z_c(w) + \text{constant}$.

(b) Suppose a base category $c = 1$ is generated by $q(\mathbf{I})$, and suppose we fix the scoring function and the bias term of the base category $f_1(\mathbf{I}; w) = 0$, and $b_1 = 0$. If $p(c|\mathbf{I}; w)$ is given by model (2.3), then $p(\mathbf{I}|c; w) = p_c(\mathbf{I}; w)$ is of the form of model (3.1), with $b_c = \log \rho_c - \log \rho_1 + \log Z_c(w)$.

Proposition 1 can be proved by a simple exercise of the Bayes rule. Result (a) has already been explained in [4]. Result (b) is stronger and is new. First, it is entirely reasonable to include Gaussian white noise images as a base category and demand the discriminative ConvNet (2.3) not to misclassify the Gaussian white noise as an

object category. It is also reasonable to fix the scoring function and bias term of this base category at 0 in training for the sake of identifiability. In fact, in the binary (two-category) logistic regression, the scoring function and the bias term for the negative category are always fixed at 0. Then

$$\frac{p(c|\mathbf{I};w)}{p(c=1|\mathbf{I};w)} = \exp[f_c(\mathbf{I};w) + b_c]. \tag{3.2}$$

Meanwhile

$$\frac{p(c|\mathbf{I};w)}{p(c=1|\mathbf{I};w)} = \frac{p(c,\mathbf{I}|w)}{p(c=1,\mathbf{I}|w)} = \frac{\rho_c p_c(\mathbf{I};w)}{\rho_1 q(\mathbf{I})}, \tag{3.3}$$

because $p(c,\mathbf{I}|w) = p(c|\mathbf{I};w)P(\mathbf{I};w)$, where the marginal distribution $P(\mathbf{I};w) = \sum_{c=1}^{C} \rho_c p(\mathbf{I}|c;w)$ is the mixture of all the categories. Thus $p_c(\mathbf{I};w)$ is of the form of model (3.1).

As to learning, we may use the discriminative log-likelihood based on $\log p(c|\mathbf{I};w)$, or we may use the generative log-likelihood based on $\log p(c,\mathbf{I}|w) = \log p_c(\mathbf{I};w) + \log \rho_c$. Because $\log p(c,\mathbf{I}|w) = \log p(c|\mathbf{I};w) + \log P(\mathbf{I};w)$, the discriminative log-likelihood $\log p(c|\mathbf{I};w)$ is without the marginal log-likelihood $\log P(\mathbf{I};w)$, resulting in the loss of statistical efficiency.

If we only observe unlabeled data $\{\mathbf{I}_m, m = 1,...,M\}$, we may still use the exponential tilting form to model and learn from them. A possible model is to learn filters at a certain convolutional layer $L \in \{1,...,\mathcal{L}\}$ of a ConvNet.

DEFINITION 3. *Generative ConvNet (convolutional version or FRAME version): we define the following Markov random field model as the convolutional version or the FRAME version of generative ConvNet:*

$$p(\mathbf{I};w) = \frac{1}{Z(w)} \exp\left[\sum_{k=1}^{K} \sum_{x \in \mathcal{D}_L} [F_k^{(L)} * \mathbf{I}](x)\right] q(\mathbf{I}), \tag{3.4}$$

where w consists of all the weight and bias terms that define the filters $(F_k^{(L)}, k = 1,...,K = N_L)$, and q is the Gaussian white noise model.

Model (3.4) corresponds to the exponential tilting model (3.1) with scoring function

$$f(\mathbf{I};w) = \sum_{k=1}^{K} \sum_{x \in \mathcal{D}_L} [F_k^{(L)} * \mathbf{I}](x). \tag{3.5}$$

Essentially the above model treats the images $\{\mathbf{I}_m\}$ as coming from a single meta-category, which is to be discriminated from the base category q by the filters $(F_k^{(L)})$

to be learned from the data. However, it may be too easy to discriminate $\{\mathbf{I}_m\}$ from q, so that we cannot learn anything meaningful by the discriminative log-likelihood. In this case, we can learn w based on the generative log-likelihood $L(w) = \sum_{m=1}^{M} \log p(\mathbf{I}_m; w)/M$, with $p(\mathbf{I}; w)$ defined by (3.4). The learning of filters $\{F_k^{(L)}\}$ by the generative log-likelihood is considered to be unsupervised because the observed images are unlabeled, i.e., their categories are unknown.

For the rest of the paper, we shall focus on the model (3.4), but all the results can be easily extended to model (3.1).

4. Generative ConvNet as a Markov random field and hierarchical FRAME model. Model (3.4) is a Markov random field model [2], where the clique functions are $[F_k^{(L)} * \mathbf{I}](x), \forall k, x$. According to the Hammersley-Clifford theorem [2], a Markov random filed model can be written as

$$p(\mathbf{I}) = \frac{1}{Z} \exp\left[\sum_{\mathcal{C}} \lambda_{\mathcal{C}}(\mathbf{I}(\mathcal{C}))\right], \tag{4.1}$$

where $\mathcal{C} \subset \mathcal{D}$ are the cliques, and each clique $\mathcal{C}$ consists of pixels that are neighbors of each other according to a pre-defined neighborhood system. $\mathbf{I}(\mathcal{C})$ are the intensities of pixels in clique $\mathcal{C}$, and $\lambda_{\mathcal{C}}$ is the potential function. The challenge in developing a Markov random field model is to specify the clique functions $\lambda_{\mathcal{C}}$ and estimate them from the data. Model (3.4) solves this problem by assuming $\lambda_{\mathcal{C}}(\mathbf{I}(\mathcal{C})) = [F_k^{(L)} * \mathbf{I}](x)$ using the ConvNet filters. The Gaussian white noise $q(\mathbf{I})$ contributes to cliques that consist of single pixels.

At the first glance, defining clique functions by the ConvNet filters may appear to be arbitrary and ad hoc, but it is actually based on a rich tradition in generative modeling. The first model in the literature that represents the clique functions by non-linear transformations of linear filter responses is the FRAME (Filters, Random field, And Maximum Entropy) model [43]. The generative ConvNet (3.4) can be considered a hierarchical FRAME model, with alternating layers of linear filtering and non-linearity.

More importantly, the recursive form of equation (2.1) has an interesting justification by generative modeling. Based on filters $\{F_i^{(l-1)}, \forall i\}$ at layer $l-1$, each filter $F_k^{(l)}$ at layer l corresponds to a non-stationary FRAME model [36, 21] of an image patch defined on the support of the filter, $\mathcal{S}_l$, and centered at x:

$$p_k^{(l)}(\mathbf{I}; w, x) = \frac{1}{Z_k^{(l)}(w, x)} \exp\left[\sum_{i=1}^{N_{l-1}} \sum_{y \in \mathcal{S}_l} w_{i,y}^{(l,k)} [F_i^{(l-1)} * \mathbf{I}](x + y)\right] q(\mathbf{I}), \tag{4.2}$$

where $(w_{i,y}^{(l,k)}, \forall i, y)$ are the parameters of the above exponential family model. The model is also a generative ConvNet model, and it can generate vivid object patterns

[21]. The bias term $b_{l,k}$ and the ReLU non-linearity $h()$ in equation (2.1) can be justified by a mixture model $P_k^{(l)}(\mathbf{I}; w, x) = \alpha p_k^{(l)}(\mathbf{I}; w, x) + (1 - \alpha) q(\mathbf{I})$, which is a mixture of presence and absence of the object pattern modeled by the non-stationary FRAME model (4.2). Writing $P_k^{(l)}(\mathbf{I}; w, x) = \exp\left[f_k^{(l)}(\mathbf{I}; w, x)\right] q(\mathbf{I})$ gives rise to the soft-max non-linearity $\log(1 + e^r)$ that can be approximated by ReLU $\max(0, r)$, and the bias term $b_{l,k}$ that is determined by α and $Z_k^{(l)}(w, x)$. Finally, taking the product of $P_k^{(l)}(\mathbf{I}; w, x)$ over k and x gives rise to a Product of Experts (PoE) model [6], which is the generative ConvNet (3.4) using filters at layer l. See [21] for details.

5. A prototype model. We shall explain the key properties of the generative ConvNet by the simplest prototype model, which makes crystal clear most of the key elements of the generative ConvNet model (3.4). A similar model was studied by [37]. The generative ConvNet can be obtained from the prototype model by unfolding the latter both convolutionally and hierarchically, but with much more involved notation that is in danger of obscuring the key ideas. Hence it is helpful to start from the prototype model.

In our prototype model, we assume that the image domain $\mathcal{D}$ is small (e.g., 10×10). Suppose we want to learn a dictionary of filters or basis functions from a set of observed image patches $\{\mathbf{I}_m, m = 1, ..., M\}$ defined on $\mathcal{D}$. We denote these filters or basis functions by $(\mathbf{w}_k, k = 1, ..., K)$, where each $\mathbf{w}_k$ itself is an image patch defined on $\mathcal{D}$. Let $\langle \mathbf{I}, \mathbf{w}_k \rangle = \sum_{x \in \mathcal{D}} \mathbf{w}_k(x) \mathbf{I}(x)$ be the inner product between image patches $\mathbf{I}$ and $\mathbf{w}_k$. It is also the response of $\mathbf{I}$ to the linear filter $\mathbf{w}_k$.

DEFINITION 4. *Prototype model: We define the following random field model as the prototype model:*

$$p(\mathbf{I}; w) = \frac{1}{Z(w)} \exp\left[\sum_{k=1}^{K} h(\langle \mathbf{I}, \mathbf{w}_k \rangle + b_k)\right] q(\mathbf{I}), \tag{5.1}$$

where b_k is the bias term, $w = (\mathbf{w}_k, b_k, k = 1, ..., K)$, and $h(r) = \max(r, 0)$. $q(\mathbf{I})$ is the Gaussian white noise model,

$$q(\mathbf{I}) = \frac{1}{(2\pi\sigma^2)^{|\mathcal{D}|/2}} \exp\left[-\frac{1}{2\sigma^2} ||\mathbf{I}||^2\right], \tag{5.2}$$

where $|\mathcal{D}|$ counts the number of pixels in the domain $\mathcal{D}$.

The following are our findings about the prototype model.

(1) *Piecewise Gaussian and binary activation variables:* The prototype model (5.1) is a piecewise Gaussian distribution. Without loss of generality, let us assume

$\sigma^2 = 1$ in $q(\mathbf{I})$. Define the binary activation variable $\delta_k(\mathbf{I}; w) = 1$ if $\langle \mathbf{I}, \mathbf{w}_k \rangle + b_k > 0$ and $\delta_k(\mathbf{I}; w) = 0$ otherwise, i.e.,

$$\delta_k(\mathbf{I}; w) = 1(\langle \mathbf{I}, \mathbf{w}_k \rangle + b_k > 0), \tag{5.3}$$

where $1()$ is the indicator function. Then

$$h(\langle \mathbf{I}, \mathbf{w}_k \rangle + b_k) = \delta_k(\mathbf{I}; w)(\langle \mathbf{I}, \mathbf{w}_k \rangle + b_k). \tag{5.4}$$

The image space is divided into 2^K pieces by the K hyper-planes, $\langle \mathbf{I}, \mathbf{w}_k \rangle + b_k = 0$, $k = 1, ..., K$, according to the values of the binary activation variables $(\delta_k(\mathbf{I}; w), k = 1, ..., K)$. Consider the piece where $\delta_k(\mathbf{I}; w) = \delta_k$ for $k = 1, ..., K$. Here we abuse the notation slightly where $\delta_k \in \{0, 1\}$ on the right hand side denotes the value of $\delta_k(\mathbf{I}; w)$. Write $\delta(\mathbf{I}; w) = (\delta_k(\mathbf{I}; w), k = 1, ..., K)$, and $\delta = (\delta_k, k = 1, ..., K)$ as an instantiation of $\delta(\mathbf{I}; w)$. We call $\delta(\mathbf{I}; w)$ the activation pattern of $\mathbf{I}$. Let $A(\delta; w) = \{\mathbf{I} : \delta(\mathbf{I}; w) = \delta\}$ be the piece of image space that consists of images sharing the same activation pattern δ, then the probability density on this piece

$$\begin{aligned} p(\mathbf{I}; w, \delta) &\propto \exp\left[\sum_{k=1}^{K} \delta_k b_k + \langle \mathbf{I}, \sum_{k=1}^{K} \delta_k \mathbf{w}_k \rangle - \frac{\|\mathbf{I}\|^2}{2}\right] \\ &\propto \exp\left[-\frac{1}{2}\|\mathbf{I} - \sum_{k=1}^{K} \delta_k \mathbf{w}_k\|^2\right], \end{aligned} \tag{5.5}$$

which is $\mathrm{N}(\sum_k \delta_k \mathbf{w}_k, \mathbf{1})$ restricted to the piece $A(\delta; w)$, where the bold font $\mathbf{1}$ is the identity matrix (recall we assume $\sigma^2 = 1$). $\delta = (\delta_k)$ are the binary activation variables that reconstruct the mean of this Gaussian piece, $\sum_k \delta_k \mathbf{w}_k$, which can be considered an approximated reconstruction of the images in $A(\delta; w)$.

(2) *Synthesis via reconstruction:* One can sample from $p(\mathbf{I}; w)$ in (5.1) by the Langevin dynamics:

$$\mathbf{I}_{\tau+1} = \mathbf{I}_\tau - \frac{\epsilon^2}{2}\left[\mathbf{I}_\tau - \sum_{k=1}^{K} \delta_k(\mathbf{I}_\tau; w)\mathbf{w}_k\right] + \epsilon Z_\tau, \tag{5.6}$$

where τ denotes the time step, ϵ denotes the step size, assumed to be sufficiently small throughout this paper, and $Z_\tau \sim \mathrm{N}(0, \mathbf{1})$. The dynamics is driven by the reconstruction error $\mathbf{I} - \sum_k \delta_k \mathbf{w}_k$, where the reconstruction is based on the binary activation variables (δ_k). This links synthesis to reconstruction.

(3) *Auto-encoding local modes:* The deterministic part of the dynamics $\mathbf{I}_{\tau+1} = \mathbf{I}_\tau - \frac{\epsilon^2}{2}\left[\mathbf{I}_\tau - \sum_{k=1}^{K} \delta_k(\mathbf{I}_\tau; w)\mathbf{w}_k\right]$ will converge to a local energy minimum $\hat{\mathbf{I}}$,

where

$$\hat{\mathbf{I}} = \sum_{k=1}^{K} \delta_k(\hat{\mathbf{I}}; w)\mathbf{w}_k. \tag{5.7}$$

That is, $\hat{\mathbf{I}}$ is auto-encoding. The encoding process is bottom-up and infers $\delta_k = \delta_k(\hat{\mathbf{I}}; w) = 1(\langle \hat{\mathbf{I}}, \mathbf{w}_k \rangle + b_k > 0)$. The decoding process is top-down and reconstructs $\hat{\mathbf{I}} = \sum_k \delta_k \mathbf{w}_k$. In the encoding process, $\mathbf{w}_k$ plays the role of filter. In the decoding process, $\mathbf{w}_k$ plays the role of basis function. The local modes are the means of the Gaussian pieces mentioned above, but the converse if not true, since $\hat{\mathbf{I}}$ may not belong to $A(\delta; w)$.

The learning of w from training images $\{\mathbf{I}_m, m = 1, ..., M\}$ can be accomplished by maximum likelihood. Define $L(w) = \sum_{m=1}^{M} \log p(\mathbf{I}; w)/M$, with $p(\mathbf{I}; w)$ defined in (5.1), then

$$\begin{aligned} \frac{\partial L(w)}{\partial \mathbf{w}_k} &= \frac{1}{M} \sum_{m=1}^{M} \delta_k(\mathbf{I}_m; w)\mathbf{I} - \mathrm{E}_w[\delta_k(\mathbf{I}; w)\mathbf{I}], \\ \frac{\partial L(w)}{\partial b_k} &= \frac{1}{M} \sum_{m=1}^{M} \delta_k(\mathbf{I}_m; w) - \mathrm{E}_w[\delta_k(\mathbf{I}; w)], \end{aligned} \tag{5.8}$$

where E_w is the expectation with respect to $p(\mathbf{I}; w)$, and can be approximated by Monte Carlo samples produced by the Langevin dynamics. At the maximum likelihood estimate of w, the model matches the observed images in terms of (1) the frequency that δ_k is on, and (2) the average of images on which δ_k is on, for every k.

The reference distribution is usually not emphasized in previous treatments of exponential family models. It plays a crucial role in our work. The Gaussian white noise reference distribution makes the density $p(\mathbf{I}; w)$ in (5.1) integrable, and leads to the reconstruction error interpretation in the Langevin dynamics. It is also crucial for the auto-encoder form of the local modes.

The reason we choose Gaussian white noise model as the reference distribution is that it is the maximum entropy distribution with given marginal mean and variance. Thus it is the most featureless distribution. The exponential tilting seeks to explain the departure from the Gaussian distribution by learning non-Gaussian features.

Another justification for the Gaussian white noise distribution is that it is the limiting distribution if we zoom out a texture image due to the central limit theorem, a phenomenon called information scaling by [34]. The exponential tilting is to recover the non-Gaussian distribution before the central limit theorem takes effect.

A key point is that the ReLU $h(r) = \max(0, r)$ can be written as $h(r) = \delta r$, where $\delta = 1(r > 0)$. Thus given the binary indicators, the scoring function $f(\mathbf{I}; w) = \sum_{k=1}^{K} h(\langle \mathbf{I}, \mathbf{w}_k \rangle + b_k)$ is piecewise linear in $\mathbf{I}$. Together with the Gaussian white noise term $\|\mathbf{I}\|^2/2$, the probability distribution is piecewise Gaussian. The ReLU non-linarity is a perfect match to the Gaussian white noise q. The combination of the two gives us a simple and rich structure.

We would also like to compare the prototype model with two strategies for developing generative models. Strategy 1: Treat δ_k as latent random variables. We put a prior distribution on $(\delta_k, k = 1, ..., K)$, usually assumed to be independent of each other, and then let (δ_k) generate $\mathbf{I}$, e.g., $\mathbf{I} \sim \mathrm{N}(\sum_k \delta_k \mathbf{w}_k, \sigma^2)$. The model has an explicit representation of $\mathbf{I}$ in terms of $\sum_k \delta_k \mathbf{w}_k$, where $(\mathbf{w}_k)$ play the role of basis functions for representing $\mathbf{I}$. The problem with such a model is that the posterior distribution of (δ_k) given $\mathbf{I}$ is usually not in closed form. In fact, inferring (δ_k) from $\mathbf{I}$ given the basis functions $(\mathbf{w}_k)$ is a variable selection problem, where $\mathbf{I}$ is the response variable, and $(\mathbf{w}_k)$ are the predictor variables. The difficulty with the inference of (δ_k) makes it difficult to learn w. Strategy 2: Assume an exponential family model based on the sufficient statistics or feature statistics $h(\langle \mathbf{I}, \mathbf{w}_k \rangle)$, where $(\mathbf{w}_k)$ play the role of linear filters or projections. The problem with such a model is that in general we do not have an explicit representation or reconstruction of the image $\mathbf{I}$. However, with ReLU h and Gaussian white noise q, we can elucidate an explicit representation or reconstruction as explained above. Thus our method strikes a middle ground between the above two strategies. Such a representation is not only conceptually appealing, it can also be important for learning, where we may seek to minimize the reconstruction error instead of maximizing the log-likelihood, which requires MCMC sampling. In Section 7, we shall see that a popular variation of the maximum likelihood learning algorithm called contrastive divergence actually seeks to minimize the reconstruction error.

In Strategy 1, it is possible to make the posterior distribution of (δ_k) given $\mathbf{I}$ explicit at the expense of an implicit prior distribution on (δ_k), such as in the restricted Boltzmann machine [7, 29, 18]. However, in the hierarchical generalizations such as the deep Boltzmann machine, the posterior of the binary variables becomes intractable [29]. In contrast, the prototype model can be easily generalized to a hierarchical model without such a difficulty as we shall see in the next section.

The binary activation variables (δ_k) indicate the selection of the basis functions from $(\mathbf{w}_k)$ for representing $\mathbf{I}$. In the prototype model, the selection is based on $\delta_k = 1(\langle \mathbf{I}, \mathbf{w}_k \rangle + b_k > 0)$. It is possible to generalize the definition of δ_k to incorporate explaining-away competition, e.g., among highly correlated basis functions, only the one with the biggest $|\langle \mathbf{I}, w_k \rangle|$ should be selected. Thus writing the model in terms of the binary activation variables may lead to interesting generalizations,

which is another advantage of our model.

6. Representational structure of generative ConvNet. Just as the discriminative ConvNet can be considered an unfolded version of the GLM, the generative ConvNet can be considered an unfolded version of the prototype model. In order to generalize the prototype model (5.1) to the generative ConvNet (3.4), we only need to add two elements: (1) Horizontal unfolding: make the filters $(\mathbf{w}_k)$ convolutional. (2) Vertical unfolding: make the filters $(\mathbf{w}_k)$ multi-layer or hierarchical. The results we have obtained for the prototype model can be unfolded accordingly.

Following the analysis of the prototype model, our plan is to write the scoring function $f(\mathbf{I}; w) = \sum_{k=1}^{K} \sum_{x \in \mathcal{D}_L} [F_k^{(L)} * \mathbf{I}](x)$ of the generative ConvNet (3.4) as a piecewise linear function of $\mathbf{I}$. We shall use the vector notation for the ConvNet in order to minimize the use of indices (although there are still plenty remaining). For filters at level l, the N_l filters are denoted by the compact notation $\mathbf{F}^{(l)} = (F_k^{(l)}, k = 1, ..., N_l)$. The N_l filtered images or feature maps are denoted by the compact notation $\mathbf{F}^{(l)} * \mathbf{I} = (F_k^{(l)} * \mathbf{I}, k = 1, ..., N_l)$. $\mathbf{F}^{(l)} * \mathbf{I}$ is a 3D image, containing all the N_l filtered images at layer l. See Fig. 2 for an illustration, where the filtered images at each layer are illustrated as a 3D image. In the vector notation, the recursive formula (2.1) of ConvNet filters can be rewritten as

$$[F_k^{(l)} * \mathbf{I}](x) = h\left(\langle \mathbf{w}_{k,x}^{(l)}, \mathbf{F}^{(l-1)} * \mathbf{I} \rangle + b_{l,k}\right), \tag{6.1}$$

where $\mathbf{w}_{k,x}^{(l)}$ matches the dimension of $\mathbf{F}^{(l-1)} * \mathbf{I}$, which is a 3D image containing all the N_{l-1} filtered images at layer $l-1$. Specifically,

$$\langle \mathbf{w}_{k,x}^{(l)}, \mathbf{F}^{(l-1)} * \mathbf{I} \rangle = \sum_{i=1}^{N_{l-1}} \sum_{y \in \mathcal{S}_l} w_{i,y}^{(l,k)} [F_i^{(l-1)} * \mathbf{I}](x+y). \tag{6.2}$$

The 3D basis function $\mathbf{w}_{k,x}^{(l)}$ is locally supported (on $x+\mathcal{S}_l$), and $(\mathbf{w}_{k,x}^{(l)})$ are spatially translated copies for different positions x, i.e.,

$$\mathbf{w}_{k,x,i}^{(l)}(x+y) = w_{i,y}^{(l,k)}, \tag{6.3}$$

for $i \in \{1, ..., N_{l-1}\}$, $x \in \mathcal{D}_l$ and $y \in \mathcal{S}_l$. For instance, at layer $l = 1$, $\mathbf{w}_{k,x}^{(1)}$ is a Gabor-like wavelet of type k centered at position x. In Fig. 2, the 3D filtered images at layer l, $\mathbf{F}^{(l)} * \mathbf{I}$, is illustrated by a yellow cuboid. The sub-sampled version is illustrated by a blue cuboid. The locally supported 3D basis function, $\mathbf{w}_{k,x}^{(l)}$, is illustrated by a green cuboid.

$\mathbf{w}_{k,x}^{(l)}$ is the unfolded version of $\mathbf{w}_k$ in the prototype model, where x indexes the position for convolutional unfolding, and l indexes the layer for hierarchical unfolding.

Define the binary activation variable

$$\delta^{(l)}_{k,x}(\mathbf{I}; w) = 1\left(\langle \mathbf{w}^{(l)}_{k,x}, \mathbf{F}^{(l-1)} * \mathbf{I}\rangle + b_{l,k} > 0\right). \tag{6.4}$$

Since $F^{(l)}_k$ corresponds to a non-stationary FRAME model (4.2), $\delta^{(l)}_{k,x}(\mathbf{I}; w)$ is a decision maker based on the likelihood ratio test of $H_1 : p^{(l)}_k(\mathbf{I}; w, x)$ vs $H_0 : q(\mathbf{I})$ for detecting the pattern modeled by $F^{(l)}_k$.

According to (2.1), we have the following bottom-up process:

$$[F^{(l)}_k * \mathbf{I}](x) = \delta^{(l)}_{k,x}(\mathbf{I}; w)\left(\langle \mathbf{w}^{(l)}_{k,x}, \mathbf{F}^{(l-1)} * \mathbf{I}\rangle + b_{l,k}\right). \tag{6.5}$$

Let $\delta(\mathbf{I}; w) = (\delta^{(l)}_{k,x}(\mathbf{I}; w), \forall k, x, l)$ be the activation pattern at all the layers. The activation pattern $\delta(\mathbf{I}; w)$ can be computed by the bottom-up process (6.4) and (6.5) of the ConvNet.

For the scoring function $f(\mathbf{I}; w) = \sum_{k=1}^{K} \sum_{x\in\mathcal{D}_L} [F^{(L)}_k * \mathbf{I}](x)$ defined in (3.5) for the generative ConvNet, we can write it in terms of lower layers $(l \le L)$ of filter responses:

$$\begin{aligned} f(\mathbf{I}; w) &= \alpha_l + \langle \mathbf{B}^{(l)}, \mathbf{F}^{(l)} * \mathbf{I}\rangle \\ &= \alpha_l + \sum_{k=1}^{N_l} \sum_{x\in\mathcal{D}_l} \mathbf{B}^{(l)}_k(x)[F^{(l)}_k * \mathbf{I}](x), \end{aligned} \tag{6.6}$$

where $\mathbf{B}^{(l)} = (\mathbf{B}^{(l)}_k(x), k = 1, ..., N_l, x \in \mathcal{D}_l)$ consists of the maps of the coefficients at layer l. $\mathbf{B}^{(l)}$ matches the dimension of $\mathbf{F}^{(l)} * \mathbf{I}$. When $l = L$, $\mathbf{B}^{(L)}$ consists of maps of 1's, i.e., $\mathbf{B}^{(L)}_k(x) = 1$ for $k = 1, ..., K = N_L$ and $x \in \mathcal{D}_L$. According to equations (6.5) and (6.6), we have the following top-down process:

$$\mathbf{B}^{(l-1)} = \sum_{k=1}^{N_l} \sum_{x\in\mathcal{D}_l} \mathbf{B}^{(l)}_k(x)\delta^{(l)}_{k,x}(\mathbf{I}; w)\mathbf{w}^{(l)}_{k,x}, \tag{6.7}$$

where both $\mathbf{B}^{(l-1)}$ and $\mathbf{w}^{(l)}_{k,x}$ match the dimension of $\mathbf{F}^{(l-1)} * \mathbf{I}$. Equation (6.7) is a top-down deconvolution process, where $\mathbf{B}^{(l)}_k(x)\delta^{(l)}_{k,x}$ serves as the coefficient of the basis function $\mathbf{w}^{(l)}_{k,x}$. The top-down deconvolution process (6.7) is similar to but subtly different from that in [39], because equation (6.7) is controlled by the multiple layers of activation variables $\delta^{(l)}_{k,x}$ computed in the bottom-up process of the ConvNet. Specifically, $\delta^{(l)}_{k,x}$ turns on or off the basis function $\mathbf{w}^{(l)}_{k,x}$, while $\delta^{(l)}_{k,x}$ is determined by $F^{(l)}_k$. The recursive relationship for α_l can be similarly derived.

In the bottom-up convolution process (6.5), $(\mathbf{w}_{k,x}^{(l)})$ serve as filters. In the top-down deconvolution process (6.7), $(\mathbf{w}_{k,x}^{(l)})$ serve as basis functions.

Let $\mathbf{B} = \mathbf{B}^{(0)}$, $\alpha = \alpha_0$. Since $\mathbf{F}^{(0)} * \mathbf{I} = \mathbf{I}$, we have $f(\mathbf{I}; w) = \alpha + \langle \mathbf{I}, \mathbf{B} \rangle$. Note that $\mathbf{B}$ depends on the activation pattern $\delta(\mathbf{I}; w) = (\delta_{k,x}^{(l)}(\mathbf{I}; w), \forall k, x, l)$, as well as w that collects the weight and bias parameters at all the layers.

On the piece of image space $A(\delta; w) = \{\mathbf{I} : \delta(\mathbf{I}; w) = \delta\}$ of a fixed activation pattern (again we slightly abuse the notation where $\delta = (\delta_{k,x}^{(l)} \in \{0, 1\}, \forall k, x, l)$ denotes an instantiation of the activation pattern), $\mathbf{B}$ and α depend on δ and w. To make this dependency explicit, we denote $\mathbf{B} = \mathbf{B}_{w,\delta}$ and $\alpha = \alpha_{w,\delta}$, thus

$$f(\mathbf{I}; w) = \alpha_{w,\delta} + \langle \mathbf{I}, \mathbf{B}_{w,\delta} \rangle. \tag{6.8}$$

See [23] for an analysis of the number of linear pieces.

THEOREM 1. *Generative ConvNet is piecewise Gaussian: With ReLU* $h(r) = \max(0, r)$ *and Gaussian white noise* $q(\mathbf{I})$, $p(\mathbf{I}; w)$ *of model (3.4) is piecewise Gaussian. On each piece* $A(\delta; w)$, *the density is* $\mathrm{N}(\mathbf{B}_{w,\delta}, \mathbf{1})$ *truncated to* $A(\delta; w)$, *i.e.,* $\mathbf{B}_{w,\delta}$ *is an approximated reconstruction of images in* $A(\delta; w)$.

Theorem 1 follows from the fact that on $A(\delta; w)$,

$$\begin{aligned} p(\mathbf{I}; w, \delta) &\propto \exp\left[\alpha_{w,\delta} + \langle \mathbf{I}, \mathbf{B}_{w,\delta} \rangle - \frac{\|\mathbf{I}\|^2}{2}\right] \\ &\propto \exp\left[-\frac{1}{2}\|\mathbf{I} - \mathbf{B}_{w,\delta}\|^2\right], \end{aligned} \tag{6.9}$$

which is $\mathrm{N}(\mathbf{B}_{w,\delta}, \mathbf{1})$ restricted to $A(\delta; w)$.

For each $\mathbf{I}$, the binary activation variables in $\delta = \delta(\mathbf{I}; w)$ are computed by the bottom-up detection process (6.4) and (6.5), and $\mathbf{B}_{w,\delta}$ is computed by the top-down deconvolution process (6.7).

One can sample from $p(\mathbf{I}; w)$ of model (3.4) by the Langevin dynamics:

$$\mathbf{I}_{\tau+1} = \mathbf{I}_\tau - \frac{\epsilon^2}{2}\left[\mathbf{I}_\tau - \mathbf{B}_{w,\delta(\mathbf{I}_\tau; w)}\right] + \epsilon Z_\tau, \tag{6.10}$$

where $Z_\tau \sim \mathrm{N}(0, \mathbf{1})$. Again, the dynamics is driven by the reconstruction error $\mathbf{I} - \mathbf{B}_{w,\delta(\mathbf{I}; w)}$.

The deterministic part of the Langevin equation (6.10) was employed by [41] for exploring the local modes of the FRAME model. They called it the Gibbs Reaction and Diffusion Equation (GRADE). The GRADE attractor dynamics $\mathbf{I}_{\tau+1} = \mathbf{I}_\tau - \frac{\epsilon^2}{2}\left[\mathbf{I}_\tau - \mathbf{B}_{w,\delta(\mathbf{I}_\tau; w)}\right]$ converges to a local energy minimum that is auto-encoding.

PROPOSITION 2. *The local modes are auto-encoding: Let $\hat{\mathbf{I}}$ be a local maximum of $p(\mathbf{I}; w)$ of model (3.4), then $\hat{\mathbf{I}}$ is auto-encoding with the following bottom-up and top-down passes:*

$$\begin{aligned} &\textit{Bottom-up encoding: } \delta = \delta(\hat{\mathbf{I}}; w); \\ &\textit{Top-down decoding: } \hat{\mathbf{I}} = \mathbf{B}_{w,\delta}. \end{aligned} \tag{6.11}$$

The local energy minima are the means of the Gaussian pieces in Theorem 1, but the reverse is not necessarily true because $\mathbf{B}_{w,\delta}$ does not necessarily belong to $A(\delta; w)$. But if $\mathbf{B}_{w,\delta} \in A(\delta; w)$, then $\mathbf{B}_{w,\delta}$ must be a local mode.

Proposition 2 can be generalized to general non-linear $h()$, whereas Theorem 1 is true only for piecewise linear $h()$ such as ReLU.

Proposition 2 is interestingly related to the Hopfield network [11] and attractor network [30]. The main idea of the Hopfield network is to memorize the observations by the local energy minima. Such a memory is content addressable in the sense that if we are given part of an observed image, we may still be able to recall the whole image by running a gradient descent algorithm towards the local mode. Such a gradient descent algorithm is called an attractor dynamics. Proposition 2 shows that the Hopfied minima can be represented by a hierarchical auto-encoder.

To gain a more comprehensive understanding of the deconvolution process, we would like to treat it from a slightly different perspective. For each filter $F_k^{(l)}$ defined in the recursive formulas (2.1) and (6.1), let $[\bar{F}_k^{(l)} * \mathbf{I}](x) = \langle \mathbf{w}_{k,x}^{(l)}, F^{(l-1)} * \mathbf{I} \rangle + b_{l,k}$ be the linear combination before ReLU non-linearity. On $A(\delta; w) = \{\mathbf{I} : \delta(\mathbf{I}; w) = \delta\}$, i.e., given the activation pattern δ, $[\bar{F}_k^{(l)} * \mathbf{I}](x)$ becomes a linear filter, and we can write

$$[\bar{F}_k^{(l)} * \mathbf{I}](x) = a_{k,x}^{(l)} + \langle \mathbf{I}, \mathbf{b}_{k,x}^{(l)} \rangle, \tag{6.12}$$

where $\mathbf{b}_{k,x}^{(l)}$ is the basis function or a basis image defined on $\mathcal{D}$. According to (2.1),

$$\begin{aligned} \mathbf{b}_{k,x}^{(l)} &= \sum_{i=1}^{N_{l-1}} \sum_{y \in \mathcal{S}_l} w_{i,y}^{(l,k)} \delta_{i,x+y}^{(l-1)} \mathbf{b}_{i,x+y}^{(l-1)}, \\ a_{k,x}^{(l)} &= \sum_{i=1}^{N_{l-1}} \sum_{y \in \mathcal{S}_l} w_{i,y}^{(l,k)} \delta_{i,x+y}^{(l-1)} a_{i,x+y}^{(l-1)} + b_{l,k}. \end{aligned} \tag{6.13}$$

At the bottom layer, for color channel $k \in \{\mathrm{R}, \mathrm{G}, \mathrm{B}\}$, we have $\mathbf{b}_{k,x}^{(0)}(y) = 1$ if $y = x$ and $\mathbf{b}_{k,x}^{(0)}(y) = 0$ otherwise, i.e., the delta-function. Also, $b_{0,k} = 0$, and $\delta_{k,x}^{(0)} = 1$.

Equation (6.13) corresponds to back-propagation calculation in the discriminative ConvNet. But it takes a novel representational role in the generative ConvNet. There are two complementary views of equation (6.13).

(1) *Bottom-up composition:* In view of the basis functions, (6.13) defines a composition process, where the higher layer basis function $\mathbf{b}_{k,x}^{(l)}$ is a composition of lower layer basis functions $\{\mathbf{b}_{i,x+y}^{(l-1)}, i = 1, ..., N_{l-1}, y \in \mathcal{S}_l\}$ with coefficients $\{w_{i,y}^{(l,k)}\delta_{i,x+y}^{(l-1)}\}$. Note that the weight parameters $w_{i,y}^{(l,k)}$ can be turned on or off by the activation variable $\delta_{i,x+y}^{(l-1)}$, so that the composition is reconfigurable and $\mathbf{b}_{k,x}^{(l)}$ is a reconfigurable basis function that follows an And-Or logic [42]. Sparsifying the connections will make the compositions more explicit and meaningful [37, 35].

(2) *Top-down decomposition:* In view of the maps of coefficients, (6.13) defines a top-down deconvolution process, where the coefficients of the lower layer basis functions $\mathbf{b}_{i,x+y}^{(l-1)}$ are obtained by expanding the coefficients of higher layer basis functions $\mathbf{b}_{k,x}^{(l)}$. This view is consistent with the deconvolution process about $\mathbf{B}^{(l)}$ in equation (6.7).

On the piece $A(\delta; w) = \{\mathbf{I} : \delta(\mathbf{I}; w) = \delta\}$, $f(\mathbf{I}; w) = \sum_{k=1}^{K} \sum_{x \in \mathcal{D}_L} [F_k^{(L)} * \mathbf{I}](x)$ in (3.5) is linear, i.e., $f(\mathbf{I}; w) = \alpha_{w,\delta} + \langle \mathbf{I}, \mathbf{B}_{w,\delta} \rangle$, where

$$\begin{aligned} \mathbf{B}_{w,\delta} &= \sum_{k=1}^{K} \sum_{x \in \mathcal{D}_L} \delta_{k,x}^{(L)} \mathbf{b}_{k,x}^{(L)}, \\ \alpha_{w,\delta} &= \sum_{k=1}^{K} \sum_{x \in \mathcal{D}_L} \delta_{k,x}^{(L)} a_{k,x}^{(L)}. \end{aligned} \tag{6.14}$$

In the prototype model, $\mathbf{B}_{w,\delta} = \sum_k \delta_k \mathbf{w}_k$ and $\alpha_{w,\delta} = \sum_k \delta_k b_k$.

7. Learning generative ConvNet. The learning of w from training images $\{\mathbf{I}_m, m = 1, ..., M\}$ can be accomplished by maximum likelihood. Let

$$L(w) = \frac{1}{M} \sum_{m=1}^{M} \log p(\mathbf{I}; w), \tag{7.1}$$

with $p(\mathbf{I}; w)$ defined in (3.4), then

$$\frac{\partial L(w)}{\partial w} = \frac{1}{M} \sum_{m=1}^{M} \frac{\partial}{\partial w} f(\mathbf{I}_m; w) - \mathrm{E}_w \left[\frac{\partial}{\partial w} f(\mathbf{I}; w) \right]. \tag{7.2}$$

The expectation can be approximated by Monte Carlo samples [38] from the Langevin dynamics (6.10). See Algorithm 1 for a description of the learning and sampling algorithm.

We can build up the model layer by layer. Given the filters at layers below, the top layer weight and bias parameters can be learned according to

$$\begin{aligned}\frac{\partial L(w)}{\partial w_{i,y}^{(L,k)}} =& \frac{1}{M}\sum_{m=1}^{M}\sum_{x\in\mathcal{D}_L}\delta_{k,x}^{(L)}(\mathbf{I}_m;w)[F_i^{(L-1)}*\mathbf{I}_m](x+y) \\ &- \mathrm{E}_w\left[\sum_{x\in\mathcal{D}_L}\delta_{k,x}^{(L)}(\mathbf{I};w)[F_i^{(L-1)}*\mathbf{I}](x+y)\right],\end{aligned} \tag{7.3}$$

and

$$\frac{\partial L(w)}{\partial b_{L,k}} = \frac{1}{M}\sum_{m=1}^{M}\sum_{x\in\mathcal{D}_L}\delta_{k,x}^{(L)}(\mathbf{I}_m;w) - \mathrm{E}_w\left[\sum_{x\in\mathcal{D}_L}\delta_{k,x}^{(L)}(\mathbf{I};w)\right]. \tag{7.4}$$

The above equations are unfolded versions of (5.8). At the maximum likelihood estimate of w, the model matches the observed images in terms of (1) the average frequency that $\delta_{k,x}^{(L)}$ is on, and (2) the average of patches of the filtered image $F_i^{(L-1)}*\mathbf{I}$ on which $\delta_{k,x}^{(L)}$ is on, where the average is pooled over x, and the matching happens for every k.

Algorithm 1 Learning and sampling algorithm

Input:

(1) training images $\{\mathbf{I}_m, m = 1, ..., M\}$
(2) number of synthesized images $\tilde{M}$
(3) number of Langevin steps L
(4) number of learning iterations T

Output:

(1) estimated parameters w
(2) synthesized images $\{\tilde{\mathbf{I}}_m, m = 1, ..., \tilde{M}\}$

1: Let $t \leftarrow 0$, initialize $w^{(0)} \leftarrow 0$.
2: Initialize $\tilde{\mathbf{I}}_m \leftarrow 0$, for $m = 1, ..., \tilde{M}$.
3: **repeat**
4: For each m, run L steps of Langevin dynamics to update $\tilde{\mathbf{I}}_m$, i.e., starting from the current $\tilde{\mathbf{I}}_m$, each step follows equation (6.10).
5: Calculate $H^{\mathrm{obs}} = \sum_{m=1}^{M}\frac{\partial}{\partial w}f(\mathbf{I}_m;w^{(t)})/M$, and $H^{\mathrm{syn}} = \sum_{m=1}^{\tilde{M}}\frac{\partial}{\partial w}f(\tilde{\mathbf{I}}_m;w^{(t)})/\tilde{M}$.
6: Update $w^{(t+1)} \leftarrow w^{(t)} + \eta(H^{\mathrm{obs}} - H^{\mathrm{syn}})$, with step size η.
7: Let $t \leftarrow t + 1$
8: **until** $t = T$

If we want to learn from big data, we may use the contrastive divergence method [6] by starting the Langevin dynamics from the observed images, and run one or a small number of iterations. Then we use the sampled images to approximate the expectation in equation (7.2) for updating the parameters. The contrastive divergence

algorithm has become very popular for learning generative models and often lead to reasonable performances, although its theoretical properties are still not well understood. We shall show that the behavior of the contrastive divergence learning of the generative ConvNet is interestingly connected to reconstruction.

Suppose we start from an observed image $\mathbf{I}^{\rm obs}$, and run a small number of iterations of Langevin dynamics (6.10) to get a synthesized image $\mathbf{I}^{\rm syn}$. If both $\mathbf{I}^{\rm obs}$ and $\mathbf{I}^{\rm syn}$ share the same activation pattern $\delta(\mathbf{I}^{\rm obs}; w) = \delta(\mathbf{I}^{\rm syn}; w) = \delta$, then $f(\mathbf{I}; w) = a_{w,\delta} + \langle \mathbf{I}, \mathbf{B}_{w,\delta} \rangle$ for both $\mathbf{I}^{\rm obs}$ and $\mathbf{I}^{\rm syn}$. Then the contribution of $\mathbf{I}^{\rm obs}$ to the learning gradient is

$$\frac{\partial}{\partial w} f(\mathbf{I}^{\rm obs}; w) - \frac{\partial}{\partial w} f(\mathbf{I}^{\rm syn}; w) = \langle \mathbf{I}^{\rm obs} - \mathbf{I}^{\rm syn}, \frac{\partial}{\partial w} \mathbf{B}_{w,\delta} \rangle. \tag{7.5}$$

If $\mathbf{I}^{\rm syn}$ is close to the mean $\mathbf{B}_{w,\delta}$ and if $\mathbf{B}_{w,\delta}$ is a local mode, then the contrastive divergence tends to reconstruct $\mathbf{I}^{\rm obs}$ by the local mode $\mathbf{B}_{w,\delta}$, because the gradient

$$\frac{\partial}{\partial w} \|\mathbf{I}^{\rm obs} - \mathbf{B}_{w,\delta}\|^2/2 = -\langle \mathbf{I}^{\rm obs} - \mathbf{B}_{w,\delta}, \frac{\partial}{\partial w} \mathbf{B}_{w,\delta} \rangle. \tag{7.6}$$

Hence the contrastive divergence learns the Hopfield network which memorizes the observations by the local modes.

We can establish a precise connection for one-step contrastive divergence.

PROPOSITION 3. *Contrastive divergence learns to reconstruct: If the one-step Langevin dynamics does not change the activation pattern, i.e., $\delta(\mathbf{I}^{\rm obs}; w) = \delta(\mathbf{I}^{\rm syn}; w) = \delta$, then the one-step contrastive divergence has an expected gradient that is proportional to the reconstruction gradient:*

$$\mathrm{E}\left[\frac{\partial}{\partial w} f(\mathbf{I}^{\rm obs}; w) - \frac{\partial}{\partial w} f(\mathbf{I}^{\rm syn}; w)\right] \propto \frac{\partial}{\partial w} \|\mathbf{I}^{\rm obs} - \mathbf{B}_{w,\delta}\|^2. \tag{7.7}$$

This is because

$$\mathbf{I}^{\rm syn} = \mathbf{I}^{\rm obs} - \frac{\epsilon^2}{2}\left[\mathbf{I}^{\rm obs} - \mathbf{B}_{w,\delta}\right] + \epsilon Z, \tag{7.8}$$

hence

$$\mathrm{E}_Z\left[\mathbf{I}^{\rm obs} - \mathbf{I}^{\rm syn}\right] \propto \mathbf{I}^{\rm obs} - \mathbf{B}_{w,\delta}, \tag{7.9}$$

and Proposition 3 follows from (7.5) and (7.6).

Proposition 3 is related to score matching estimator of [12], whose connection with the contrastive divergence based on one-step Langevin dynamics was studied by [13]. The relationship between score matching and auto-encoder was discovered by [33] and [31]. Our work can be considered a sharpened specialization of

the above mentioned connection and relationship, where the piecewise linear structure of the ConvNet greatly simplifies the matter by getting rid of the complicated second derivative terms, so that the contrastive divergence gradient becomes exactly proportional to the gradient of the reconstruction error, which is not the case in general score matching estimator. Also, our work gives a novel hierarchical realization of the relationship between probability model and auto-encoder, as well as an explicit hierarchical realization of auto-encoder based sampling of [1]. The connection with the Hopfied network also appears new.

Proposition 3 suggests that we may learn the weight parameters by directly minimizing the reconstruction error, without having to deal with Monte Caro fluctuations. As to the bias parameters, which threshold the likelihood ratio tests for pattern detection, we can simply set their values to constrain the sparsity [25] of activations. Such an unsupervised learning method is as fast as training a discriminative ConvNet in supervised learning.

In general, it is possible for multi-step Langevin dynamics to move out of the Gaussian piece of $\mathbf{I}^{\text{obs}}$ and into another piece of a different activation pattern. But it is unlikely for the activation patterns of $\mathbf{I}^{\text{obs}}$ and $\mathbf{I}^{\text{syn}}$ to be very different. It may be particularly difficult to flip the activation variables at higher layers. In this case, the contrastive divergence may be maximizing the pseudo-likelihood [2] conditioning on the higher layer binary variables. In any case, equation (7.4) makes it clear that we do not have much information to update the bias terms associated with these variables, but we can set their values by sparsity constraints as mentioned above.

8. Synthesis and reconstruction by generative ConvNet. We show that the generative ConvNet is capable of learning and generating realistic natural image patterns. Such an empirical proof of concept validates the generative capacity of the model. We also show that contrastive divergence learning can indeed reconstruct the observed images, thus empirically validating Proposition 3.

The experiments in this section are qualitative and illustrative in nature. They are not intended for quantitative performance study. The code in our experiments is based on the MatConvNet package of [32].

Unlike [21], the generative ConvNets in our experiments are learned from scratch without relying on the pre-learned filters of the existing ConvNets.

When learning the generative ConvNet, we grow the layers sequentially. Starting from the first layer, we sequentially add the layers one by one. Each time we learn the model and generate the synthesized images using Algorithm 1. While learning the new layer of filters, we can either fix lower layers of filters while updating the top layer weight and bias parameters according to equations (7.3) and (7.4), or we can additionally refine the lower layer filters by back-propagation at the same time. Both strategies work well for image synthesis. We adopt the latter strategy in our

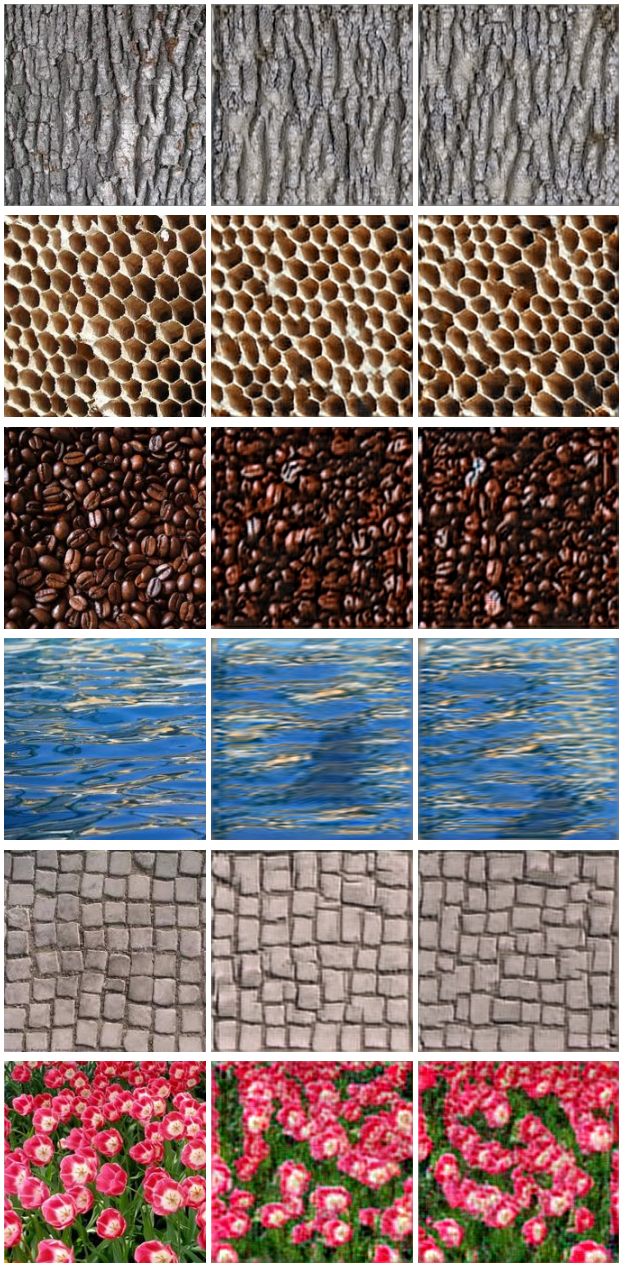

Fig 6: Generating texture patterns. For each category, the first image is the training image, and the rest are 2 of the images generated by the learning algorithm.

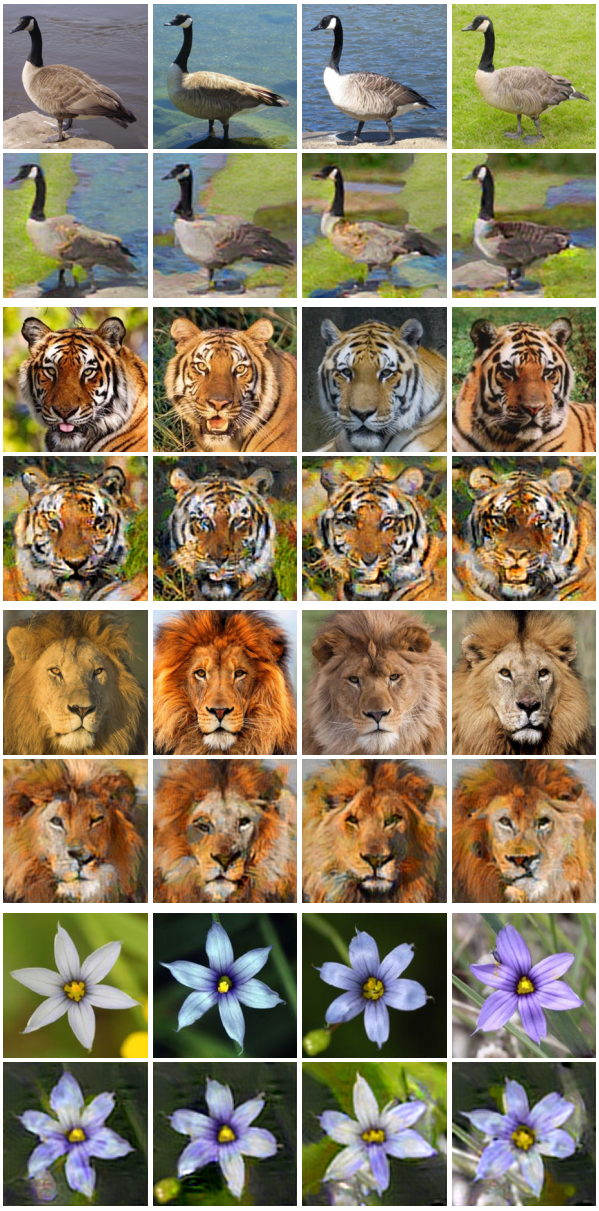

Fig 7: Generating object patterns. For each category, the first row displays 4 of the training images, and the second row displays 4 of the images generated by the learning algorithm.

Fig 8: Reconstruction by the one-step contrastive divergence. The first row displays 4 of the training images, and the second row displays the corresponding reconstructed images.

experiments.

We use $\tilde{M} = 16$ parallel chains for Langevin sampling. The number of Langevin iterations between every two consecutive updates of parameters is $L = 10$. With each new added layer, the number of learning iterations is $T = 700$. We follow the standard procedure to prepare the training images of size 224×224, whose intensities range from $[0, 255]$, and we subtract the mean image. We fix $\sigma^2 = 1$ in the reference distribution $q(\mathbf{I})$.

For each of the 3 experiments, we use the same set of parameters for all the categories without tuning.

8.1. *Experiment 1: Generating texture patterns.* We learn a 3-layer generative ConvNet. The first layer has 100 15×15 filters with sub-sampling size of 3. The second layer has 64 5×5 filters with sub-sampling size of 1. The third layer has 30 3×3 filters with sub-sampling size of 1. We learn a generative ConvNet for each category from a single training image. Fig. 6 displays the results. For each category, the first image is the training image, and the rest are 2 of the images generated by the learning algorithm.

8.2. *Experiment 2: Special case: generating aligned object patterns.* Experiment 1 shows clearly that the generative ConvNet can learn from images without alignment. We can also specialize it to learning aligned object patterns by using a single top-layer filter that covers the whole image. It is actually a non-stationary FRAME model of the form (4.2), i.e., a convolutional filter at a fixed position before ReLU non-linearity.

We learn a 4-layer generative ConvNet from images of aligned objects. The first layer has 100 7×7 filters with sub-sampling size of 2. The second layer has 64

5×5 filters with sub-sampling size of 1. The third layer has 20 3×3 filters with sub-sampling size of 1. The fourth layer is a fully connected layer with a single filter that covers the whole image. When growing the layers, we always keep the top-layer single filter, and train it together with the existing layers. We learn a generative ConvNet for each category, where the number of training images for each category is around 10, and they are collected from the Internet. Fig. 7 shows the results. For each category, the first row displays 4 of the training images, and the second row shows 4 of the images generated by the learning algorithm.

8.3. *Experiment 3: Contrastive divergence learns to reconstruct.* We evaluate the one-step contrastive divergence learning on a small training set of 10 images collected from the Internet. The ConvNet structure is the same as in experiment 1. For computational efficiency, we learn all the layers of filters simultaneously. The number of learning iterations is $T = 1200$. Starting from the observed images, the number of Langevin iterations is $L = 1$. Fig. 8 shows the results. The first row displays 4 of the training images, and the second row displays the corresponding auto-encoding reconstructions with the learned parameters.

9. General non-linearity. For general non-linearity, i.e., $h(r)$ is not necessarily ReLU or piecewise linear, some of the above results still hold. Let $h'(r)$ be the derivative of $h(r)$, then the activation variables become

$$\delta^{(l)}_{k,x}(\mathbf{I}; w) = h'\left(\langle \mathbf{w}^{(l)}_{k,x}, F^{(l-1)} * \mathbf{I}\rangle + b_{l,k}\right), \tag{9.1}$$

which is not binary in general, and the auto-encoding reconstruction

$$\mathbf{B}_{w,\delta(\mathbf{I};w)} = \frac{\partial}{\partial \mathbf{I}} f(\mathbf{I}; w), \tag{9.2}$$

where $f(\mathbf{I}; w)$ is defined by (3.5). The Langevin dynamics (6.10) and the Hopfield auto-encoder (6.11) in Proposition 2 still hold, but the piecewise Gaussian form (6.9) in Theorem 1 is lost. The exact proportionality in Proposition 3 is also lost. The crisp ReLU non-linearity is crucial for the simplicity of modeling and learning, and should be kept as much as possible.

Equations (9.1) and (9.2) show that the top-down generation is actually the back-propagation derivative (with respect to $\mathbf{I}$ instead of w) of the bottom-up recognition, i.e., representation = back-propagation.

We can also incorporate max pooling in the bottom-up computation, whose derivative is arg-max retrieval in the top-down reconstruction.

10. Conclusion. The development of the generative ConvNet in this paper is very natural, almost axiomatic, with minimal extra assumptions. Assuming the

commonly used ReLU non-linearity and a Gaussian white noise base category, the generative ConvNet is naturally derived from the discriminative ConvNet, and we show that the model has a representational structure based on multiple layers of binary activation variables. We also empirically show that the model can reconstruct the observed images and synthesize new images. It is possible to generalize the definition of the binary activation variables to take into account explaining-away competitions for learning sharp dictionaries of filters and basis functions at multiple layers.

Some of the results in this paper can be mapped to back-propagation in the discriminative ConvNet, but our reinterpretation of them in terms of representation is novel and is richly expansive. Our paper unifies, reconciles or connects the following antagonizing or disparate pairs: (1) discriminative ConvNet and generative CongNet, (2) supervised learning and unsupervised learning, (3) exponential family models and latent variable models, (4) bottom-up filters (operation) and top-down basis functions (representation), (5) synthesis (dream) and reconstruction (memory), (6) hierarchal probability model and hierarchical auto-encoder, (7) Hopfield attractor network and auto-encoder, (8) contrastive divergence (learning) and reconstruction (memory).

The generative ConvNet has the potential to learn from unlabeled data. In our future work, we shall scale up the unsupervised learning from big unlabeled data using reconstruction based methods. Our preliminary results suggest that it is possible to learn meaningful dictionaries of filters and that the learning is as fast as the discriminative training of the ConvNet. We shall report our empirical findings elsewhere.

APPENDIX A: CODE AND DATA

The code and training images can be downloaded from the project page: http://www.stat.ucla.edu/~ywu/GenerativeConvNet/main.html

ACKNOWLEDGEMENTS

The code in our work is based on the Matlab code of MatConvNet of [32]. We thank the authors for making their code public.

We thank Jifeng Dai for earlier collaboration on the generative ConvNet. We thank Wenze Hu for earlier collaboration on the inhomogeneous FRAME model.

The work is supported by NSF DMS 1310391, ONR MURI N00014-10-1-0933 and DARPA SIMPLEX N66001-15-C-4035.

REFERENCES

[1] ALAIN, G. and BENGIO, Y. (2014). What regularized auto-encoders learn from the data-generating distribution. *The Journal of Machine Learning Research* **15** 3563–3593.

[2] BESAG, J. (1974). Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society. Series B (Methodological)* 192–236.
[3] CHOROMANSKA, A., HENAFF, M., MATHIEU, M., AROUS, G. B. and LECUN, Y. (2014). The loss surface of multilayer networks. *arXiv preprint arXiv:1412.0233*.
[4] DAI, J., LU, Y. and WU, Y. N. (2015). Generative Modeling of Convolutional Neural Networks. In *ICLR*.
[5] DENG, J., DONG, W., SOCHER, R., LI, L.-J., LI, K. and FEI-FEI, L. (2009). Imagenet: A large-scale hierarchical image database. In *CVPR* 248–255. IEEE.
[6] HINTON, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Computation* **14** 1771–1800.
[7] HINTON, G. E., OSINDERO, S. and TEH, Y. W. (2006). A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation* **18** 1527–1554.
[8] HINTON, G. E., DAYAN, P., FREY, B. J. and NEAL, R. M. (1995). The" wake-sleep" algorithm for unsupervised neural networks. *Science* **268** 1158–1161.
[9] HINTON, G., OSINDERO, S., WELLING, M. and TEH, Y.-W. (2006). Unsupervised discovery of nonlinear structure using contrastive backpropagation. *Cognitive science* **30** 725–731.
[10] HONG, Y., SI, Z., HU, W., ZHU, S.-C. and WU, Y. N. (2013). Unsupervised learning of compositional sparse code for natural image representation. *Quarterly of Applied Mathematics* **72** 373–406.
[11] HOPFIELD, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences* **79** 2554–2558.
[12] HYVÄRINEN, A. (2005). Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research* **6** 695–709.
[13] HYVÄRINEN, A. (2007). Connections between score matching, contrastive divergence, and pseudolikelihood for continuous-valued variables. *Neural Networks, IEEE Transactions on* **18** 1529–1531.
[14] KINGMA, D. P. and WELLING, M. (2014). Auto-Encoding Variational Bayes. *ICLR*.
[15] KRIZHEVSKY, A., SUTSKEVER, I. and HINTON, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *NIPS* 1097–1105.
[16] LECUN, Y., BOTTOU, L., BENGIO, Y. and HAFFNER, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE* **86** 2278–2324.
[17] LECUN, Y., CHOPRA, S., HADSELL, R., RANZATO, M. and HUANG, F. J. (2006). A tutorial on energy-based learning. In *Predicting Structured Data*. MIT Press.
[18] LEE, H., GROSSE, R., RANGANATH, R. and NG, A. Y. (2009). Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th Annual International Conference on Machine Learning* 609–616. ACM.
[19] LIANG, F., LIU, C. and CARROLL, R. (2011). *Advanced Markov chain Monte Carlo methods: learning from past samples* **714**. John Wiley & Sons.
[20] LIU, J. S. (2008). *Monte Carlo strategies in scientific computing*. Springer Science & Business Media.
[21] LU, Y., ZHU, S.-C. and WU, Y. N. (2016). Learning FRAME Models Using CNN Filters. In *Thirtieth AAAI Conference on Artificial Intelligence*.
[22] MNIH, A. and GREGOR, K. (2014). Neural Variational Inference and Learning in Belief Networks. In *ICML*.
[23] MONTUFAR, G. F., PASCANU, R., CHO, K. and BENGIO, Y. (2014). On the number of linear regions of deep neural networks. In *NIPS* 2924–2932.
[24] NGIAM, J., CHEN, Z., KOH, P. W. and NG, A. Y. (2011). Learning Deep Energy Models. In *ICML*.
[25] OLSHAUSEN, B. A. and FIELD, D. J. (1997). Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research* **37** 3311–3325.

[26] REZENDE, D. J., MOHAMED, S. and WIERSTRA, D. (2014). Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *ICML* (T. JEBARA and E. P. XING, eds.) 1278-1286. JMLR Workshop and Conference Proceedings.
[27] ROTH, S. and BLACK, M. J. (2005). Fields of experts: A framework for learning image priors. In *CVPR* **2** 860–867. IEEE.
[28] RUSSAKOVSKY, O., DENG, J., SU, H., KRAUSE, J., SATHEESH, S., MA, S., HUANG, Z., KARPATHY, A., KHOSLA, A., BERNSTEIN, M., BERG, A. C. and FEI-FEI, L. (2014). ImageNet Large Scale Visual Recognition Challenge. *arXiv preprint arXiv:1409.0575*.
[29] SALAKHUTDINOV, R. and HINTON, G. E. (2009). Deep boltzmann machines. In *AISTATS*.
[30] SEUNG, H. S. (1998). Learning continuous attractors in recurrent networks. In *NIPS* 654–660. MIT Press.
[31] SWERSKY, K., RANZATO, M., BUCHMAN, D., MARLIN, B. and FREITAS, N. (2011). On Autoencoders and Score Matching for Energy Based Models. In *ICML* (L. GETOOR and T. SCHEFFER, eds.). *ICML '11* 1201–1208. ACM, New York, NY, USA.
[32] VEDALDI, A. and LENC, K. (2014). MatConvNet – Convolutional Neural Networks for MATLAB. *CoRR* **abs/1412.4564**.
[33] VINCENT, P. (2011). A connection between score matching and denoising autoencoders. *Neural computation* **23** 1661–1674.
[34] WU, Y. N., ZHU, S.-C. and GUO, C.-E. (2008). From information scaling of natural images to regimes of statistical models. *Quarterly of Applied Mathematics* **66** 81-122.
[35] WU, Y. N., SI, Z., GONG, H. and ZHU, S.-C. (2010). Learning active basis model for object detection and recognitio. *International Journal of Computer Vision* **90** 198-235.
[36] XIE, J., HU, W., ZHU, S.-C. and WU, Y. N. (2014). Learning Sparse FRAME Models for Natural Image Patterns. *International Journal of Computer Vision* 1–22.
[37] XIE, J., LU, Y., ZHU, S.-C. and WU, Y. N. (2015). Inducing wavelets into random fields via generative boosting. *Journal of Applied and Computational Harmonic Analysis*.
[38] YOUNES, L. (1999). On the convergence of Markovian stochastic algorithms with rapidly decreasing ergodicity rates. *Stochastics: An International Journal of Probability and Stochastic Processes* **65** 177–228.
[39] ZEILER, M. D. and FERGUS, R. (2014). Visualizing and understanding convolutional neural networks. *ECCV*.
[40] ZEILER, M. D., TAYLOR, G. W. and FERGUS, R. (2011). Adaptive deconvolutional networks for mid and high level feature learning. In *Computer Vision (ICCV), 2011 IEEE International Conference on* 2018–2025. IEEE.
[41] ZHU, S. C. and MUMFORD, D. B. (1998). GRADE: Gibbs Reaction And Diffusion Equations. In *ICCV*.
[42] ZHU, S. C. and MUMFORD, D. (2006). A Stochastic Grammar of Images. *Foundations and Trends in Computer Graphics and Vision* **2** 259-362.
[43] ZHU, S. C., WU, Y. N. and MUMFORD, D. (1997). Minimax entropy principle and its application to texture modeling. *Neural Computation* **9** 1627–1660.

JIANWEN XIE
DEPARTMENT OF STATISTICS
UNIVERSITY OF CALIFORNIA, LOS ANGELES
LOS ANGELES, CALIFORNIA 90095
USA
E-MAIL:JIANWEN@UCLA.EDU

YANG LU
DEPARTMENT OF STATISTICS
UNIVERSITY OF CALIFORNIA, LOS ANGELES
LOS ANGELES, CALIFORNIA 90095
USA
E-MAIL:YANGLV@UCLA.EDU

SONG-CHUN ZHU
DEPARTMENT OF STATISTICS
UNIVERSITY OF CALIFORNIA, LOS ANGELES
LOS ANGELES, CALIFORNIA 90095
USA
E-MAIL:SCZHU@STAT.UCLA.EDU

YING NIAN WU
DEPARTMENT OF STATISTICS
UNIVERSITY OF CALIFORNIA, LOS ANGELES
LOS ANGELES, CALIFORNIA 90095
USA
E-MAIL:YWU@STAT.UCLA.EDU