Markov–Gibbs Texture Modelling with Learnt Freeform Filters

Ralph Versteegen(✉), Georgy Gimel'farb, and Patricia Riddle

Department of Computer Science, The University of Auckland, Auckland, New Zealand
rver017@aucklanduni.ac.nz, {g.gimelfarb,p.riddle}@auckland.ac.nz

Abstract. Energy-based Markov–Gibbs random field (MGRF) image models describe images by statistics of localised features; hence selecting statistics is crucial. This paper presently a procedure for searching much broader than typical families of linear-filter-based statistics, by alternately optimising in continuous parameter space and discrete graphical structure space. This unifies and extends the divergent models deriving from the well-known Fields of Experts (FoE), which learn parametrised features built on small linear filters, and the constrasting FRAME (exponential family) approach which iteratively selects large filters from a fixed set. While FoE is limited by computational cost to small filters, we use large sparse (non-contiguous) filters with arbitrary shapes which can capture long-range interactions directly. A filter pre-training step also improves speed and results. Synthesis of a variety of textures shows promising abilities of the proposed models to capture both fine details and larger-scale structure with a low number of small and efficient filters.

1 Introduction

An increasing variety of energy-based models have been proposed for image and texture modelling, motivated by their applicability to both generation (e.g. image synthesis and inpainting) and inference (e.g. classification and segmentation), and as building blocks of higher-level computer vision systems. These models estimate the density function of the data and depend on the selection of features whose statistics identify relevant data attributes. In particular, traditional statistical texture models are maximum-entropy Markov–Gibbs random fields (MGRFs), i.e. MRFs with Gibbs probability distributions, with parameters adjusting the strength of the Gibbs factors/potentials which are learnt by maximum likelihood estimation (MLE). Recently a number of non-max-entropy models have been proposed with much more complex parameterised potentials (usually interpretable as compositions of linear filters) and which often also incorporate implicit or explicit latent variables. Fields-of-Experts (FoE) [17] and restricted Boltzmann machines (RBMs) [9] are influential models of this class. Although not the max-entropy solutions, the MLEs are computed identically.

Learning of MGRFs by "model nesting" [2,21,23], also known as the minimax entropy principle [22,23], is the application of the max-entropy principle to iterative model selection by repeatedly adding features/potentials to a base model.

A. Robles-Kelly et al. (Eds.): S+SSPR 2016, LNCS 10029, pp. 379–389, 2016.
DOI: 10.1007/978-3-319-49055-7_34

Each iteration selects features estimated to provide the most additional information (about disagreements between the model and data), then learns parameters that encode that information. We generalise this procedure, moving beyond traditional MGRF models, which are parametrised only with "natural parameters" which select the model with maximum entropy among those with given sufficient statistics, to consider those with potentials additionally parametrised with "feature parameters", such as filter coefficients. Both are optimised simultaneously with MLE, as in FoE. This unifies FoE (which pre-specified the graphical structure, i.e. the shapes of the filters) with the well-known FRAME [23] texture models (which used pre-specified filters) and other MGRFs using iterative selection of arbitrary features, e.g. local binary patterns [21].

Below, we apply nesting to learn MGRF texture models which capture the marginal distributions of learnt filters. Taking advantage of feature learning, we allow non-contiguous (sparse) filters, proposed to efficiently capture large-scale texture-specific visual features in a much simpler way than a multi-scale or latent-variable model. We follow FRAME and other earlier MGRFs by describing filter outputs non-parametrically with histograms (nearly entirely abandoned since) because the marginals become increasingly complex and multimodal as one moves even a short way away from the simplest near-regular textures.

This paper extends our previous work [20] which introduced model nesting with square or non-contiguous filters selected by a pre-training procedure (see Sect. 3.3) rather than MLE filter optimisation (as in FoE and RBMs), but only picked filter shapes heuristically. We present below a better method for learning non-contiguous filters, even adjusting their shapes during gradient descent, by using regularisation. We retain the cheap pre-training step, finding that it improves results compared to initialising to noise. Texture synthesis experiments with varied and difficult types of textures qualitatively compare the efficacy of different modelling frameworks and show that our models can perform at least as well as others on many textures, while only using sparse filters, minimising the cost of sampling, and without latent variables.

2 Related Work

Model nesting was described independently in [2,22] and applied to text and texture modelling respectively. Della Pietra et al. [2] used features composed from previously selected ones, testing for the occurrence of growing patterns of characters. The FRAME models [22,23] use histograms of responses of filters of varying sizes, selected by model nesting from a fixed hand-selected bank. Nesting can learn heterogeneous models; as shown in [21], texture models containing both long-range grey-level differences (GLDs) $f_{\mathrm{GLD}}(x_1, x_2) := x_2 - x_1$ (which cheaply capture second-order interactions) and higher-order local binary patterns are much more capable than either alone. Feature selection for max-entropy models is also commonly performed by adding sparsity inducing regularisation [12], which is a variant of nesting that can also remove unnecessary features.

The FoE model [17] is a spatially homogeneous (convolutional) MGRF composed of nonlinear 'expert' potentials, fed linear filter responses as input. The responses of linear filters (even random ones) averaged over natural images are usually heavy-tailed unimodal distributions, so unimodal experts were used in [17]. However as Heess et al. [8] argued, for specific texture classes the marginal distributions are more complicated, and hence introduced BiFoE models with three-parameter uni- or bi-modal expert functions. These were much more capable than FoE of modelling textures. However, as pointed out by Schmidt et al. [18] (and further by [3]), the graph of the ideal expert function need not correspond to the shape of the filter marginal; this misidentification meaning that "the original FoE model does not capture the filter statistics".

Many works on image modelling with learnt filters have found that many of the filters are zero nearly everywhere, e.g. [3,17,18]. This suggests that fixing filter sizes and shapes is inefficient. (As an exception, for regular textures with very short tessellation distance periodic filters are likely to be learnt, e.g [4].) Many researchers have also encountered difficulties in learning FoE filters and other parameters using gradient descent e.g. [3,8,11,17]. This is one reason why these models have been limited to only small filters, from 7×7 in [4,8] up to 11×11 in [10]. Compare this to the fixed filters used in FRAME of up to 31×31 which were necessary to capture long range interactions, at a very high computational cost.

Many recent MGRF image models are hierarchical, generalising the FoE by adding latent variables, e.g. [16,18], including several works on texture modelling [4,7,10,13]. With the right design, these can be marginalised out easily or used for efficient block-Gibbs sampling; RBMs are very popular for the latter scheme. MGRFs with sophisticated higher-order structures including variables which locally modulate interactions, such as to represent edge discontinuities [16], pool features, or combine multiple texture models [10], are able to model complex textures. Building on [16], Luo et al. [13] stacked layers of latent variables to build convolutional deep belief networks (DBNs) for texture modelling, producing a quickly mixing sampler with leading synthesis results.

Portilla and Simoncelli [15] introduced a powerful texture synthesis algorithm which collects covariances of wavelet responses and uses iterated projections to produce images that match these statistics. However, these are not density models so are not as broadly applicable to different tasks. More recently Gatys et al. [5] introduced a similar algorithm using nearly a million covariances of features from a fixed 21 layer deep convolutional neural network, producing very high quality synthesis results. However these hugely complex summary statistics result in overfitting and computationally expensive texture synthesis.

3 Learning Markov–Gibbs Random Fields

Formulating as a general-form exponential family distribution, a generic MGRF (without explicit hidden variables) can be defined as

$$p(\mathbf{g}|\boldsymbol{\theta}, \boldsymbol{\lambda}) = \frac{1}{Z(\boldsymbol{\theta}, \boldsymbol{\lambda})} q(\mathbf{g}) \exp(-\boldsymbol{\theta} \cdot \boldsymbol{S}(\mathbf{g}|\boldsymbol{\lambda})) \tag{1}$$

where $q(\mathbf{g})$ is the base distribution, $\mathbf{g}$ is an image, parameters are $\boldsymbol{\Lambda} := (\boldsymbol{\theta}, \boldsymbol{\lambda})$ where $\boldsymbol{\theta}$ are the *natural parameters*, and the *feature parameters* $\boldsymbol{\lambda}$ parametrise the *sufficient statistics* $\boldsymbol{S}$. $E(\mathbf{g}|\boldsymbol{\theta}, \boldsymbol{\lambda}) := -\boldsymbol{\theta} \cdot \boldsymbol{S}(\mathbf{g}|\boldsymbol{\lambda})$ is called the *energy function* of the distribution and Z is the normalising constant. $\boldsymbol{S}$ is a vector of sums of *feature functions* f_i over subsets of the image called cliques: $\boldsymbol{S}_i(\mathbf{g}|\boldsymbol{\lambda}) = \sum_c f_i(\mathbf{g}_c|\boldsymbol{\lambda}_i)$ where $\mathbf{g}_c$ denotes the values of the pixels of $\mathbf{g}$ in the clique c of f_i.

The gradient of the log-likelihood $l(\boldsymbol{\Lambda}|\mathbf{g}_{\text{obs}}) = \log p(\mathbf{g}_{\text{obs}}|\boldsymbol{\Lambda})$ for a training image $\mathbf{g}_{\text{obs}}$ is given by [1]

$$\frac{\partial}{\partial \boldsymbol{\Lambda}_i} l(\boldsymbol{\Lambda}|\mathbf{g}_{\text{obs}}) = \mathbb{E}_{p(\cdot|\boldsymbol{\Lambda})}\left[\frac{\partial E(\mathbf{g}|\boldsymbol{\Lambda})}{\partial \boldsymbol{\Lambda}_i}\right] - \frac{\partial E(\mathbf{g}_{\text{obs}}|\boldsymbol{\Lambda})}{\partial \boldsymbol{\Lambda}_i} \tag{2}$$

The expectation is intractable so must be approximated, e.g. by MCMC.

If $\boldsymbol{\lambda}$ is empty or fixed, then the distribution is an exponential family distribution, and l is unimodal in $\boldsymbol{\Lambda}$ with gradient $\frac{\partial}{\partial \boldsymbol{\theta}_i} l(\boldsymbol{\theta}|\mathbf{g}_{\text{obs}}) = \mathbb{E}_{p_i}[\boldsymbol{S}_i(\mathbf{g})] - \boldsymbol{S}_i(\mathbf{g}_{\text{obs}})$. This shows that the MLE distribution p^* satisfies the constraint $\mathbb{E}_{p^*}[\boldsymbol{S}(\mathbf{g})] = \boldsymbol{S}(\mathbf{g}_{\text{obs}})$. Further, the MLE solution is the maximum entropy (ME) distribution, i.e. it is the distribution meeting this constraint which deviates least from $q(\mathbf{g})$.

3.1 Model Nesting and Sampling

For completeness, model nesting is outlined briefly here (see [21] for details). Nesting iteratively builds a model by greedily adding potentials/features f (and corresponding constraints $\mathbb{E}_{p^*}[\boldsymbol{S}_f(\mathbf{g})] = \boldsymbol{S}_f(\mathbf{g}_{\text{obs}})$) which will most rapidly move the model distribution closer to the training data, and then learning approximate MLE parameters to meet those constraints (i.e. repairing the difference in statistics). An arbitrary base distribution q can be used, as it does not even appear in the gradient (Eq. (2)). However, we do not need these constraints to be satisfied completely (by finding the MLE parameters), but only wish to improve the model by correcting some of the statistical difference on each iteration. Thus we can make the approximation of not drawing true samples from the model, but only finding images with which to approximate $\mathbb{E}_{p^*}[\boldsymbol{S}_f(\mathbf{g})]$.

Each iteration a set or space of candidate features is searched for one or more which might most increase the likelihood $p(\mathbf{g}_{\text{obs}})$. This is normally estimated with a norm of the gradient (2) w.r.t. to the parameters $\boldsymbol{\theta}_f$ of the new potential f, i.e. $||\mathbb{E}_{p_i}[\boldsymbol{S}_f(\mathbf{g})] - \boldsymbol{S}_f(\mathbf{g}_{\text{obs}})||_1$. This can be approximated using samples obtained from the model during the previous parameter learning step. Running time is quadratic in the number of nesting iterations.

Sampling. To rapidly learn parameters and obtain approximate 'samples' from the model we use a slight variant of persistent contrastive divergence (PCD) [19], earlier called CSA [6], starting from a small 'seed' image of random noise and interleaves Gibbs sampling steps with parameter gradient descent steps with a decaying learning rate, while slowly enlarging the seed. When feature parameters are fixed, the resulting image comes close to matching the desired statistics, which means this approximates Gibbs sampling from the model with optimal natural parameters [21]. This is sufficient to estimate the required expectations, even if still far from the MLE parameters. However a longer learning process with smaller steps is needed to get close to the MLE, or when the model has poorly initialised feature parameters in which case there are no real desired statistics to begin with. We used 200 PCD/Gibbs sampler steps for each nesting iteration, to produce a 100×100 image (after trimming boundaries), and 300 steps to synthesise the final images shown in the figures. Feature parameters were kept fixed while performing final synthesis, because there the intention is to match statistics, not tune the model. These hyperparameters were set to produce reasonable results quickly, rather than approach asymptotic performance.

3.2 Filter Learning

For filter $\boldsymbol{w}$ the marginal histogram is $\boldsymbol{S}_{\boldsymbol{w}}(\mathbf{g}) := [\sum_c \text{bin}_i(\boldsymbol{w} \cdot \mathbf{g}_c) : i \in \{1, \dots, k\}]$, where $\text{bin}_i : \mathbb{R} \rightarrow [0,1]$ is the non-negative band-pass function for the ith bin, with $\sum_i \text{bin}_i(x) = 1$. In experiments we used 32 bins stretched over the full possible range (usually less than half of the bins were nonempty). While performing Gibbs sampling we use binary-valued bins for speed but in order to have meaningful gradients or subgradients some smoothing is required. Hence for the purpose of computing the energy gradient we use a triangular bin function which linearly interpolates between the centres of each two adjacent bins.

By considering a sparse non-contiguous filter as a large square filter with most coefficients zero, filter shapes can be learnt by using sparsity-inducing regularisation. Calculating $\boldsymbol{S}_{\boldsymbol{w}}(\mathbf{g}_{\text{samp}})$ or $\frac{\partial}{\partial \boldsymbol{\lambda}_{\boldsymbol{w}}} \boldsymbol{S}_{\boldsymbol{w}}(\mathbf{g}_{\text{samp}})$, or performing one step of Gibbs sampling (with caching of filter responses) are all linear in the filter size, hence starting from a large square filter and waiting for it to become sparse would be slow. Instead, we start with a sparse filter (see below), and on every MLE gradient ascent iteration consider a subset of a larger set of available coefficients (we used an area of 25×25), looping through all possibilities every 10 iterations in a strided fashion. The majority of time is spent in Gibbs sampling, so considering extra coefficients only in the gradient calculation step has an insigificant cost. We apply l_1 regularisation to filter coefficients and limit the number of coefficients which are nonzero - when this constraint is exceeded, the regularisation amount is temporarily increased to force sufficient coefficients to zero. Once zero, a coefficient is removed from the filter until it randomly re-added to the active set. This directly imposed limit requires less finetuning than the indirect limit imposed by the l_1 penalty. Filters were constrained by projection to have zero mean and bounded coefficients.

3.3 Filter Pre-training

Optimising natural parameters and filters simultaneously according to (2) creates a non-convex objective. For example, since $\frac{\partial}{\partial \boldsymbol{\lambda}_i} l(\boldsymbol{\Lambda}|\mathbf{g}_{\text{obs}}) \propto \boldsymbol{\theta}_i$, if a potential has no influence then its filter will not be changed and it remains useless, the vanishing gradient problem. One way to simplify the learning procedure is to learn feature parameters and natural parameters separately, doing so in the feature selection step of nesting rather than the MLE step. That is, we attempt to find the coefficients of an ideal filter to add to the current model by maximising the expected gain in information by doing so. This objective function of for a new filter $\boldsymbol{w}$ is the error

$$e(\boldsymbol{w}) := ||\boldsymbol{S}_{\boldsymbol{w}}(\mathbf{g}_{\text{obs}}) - \boldsymbol{S}_{\boldsymbol{w}}(\mathbf{g}_{\text{samp}})||_1 \approx ||\frac{\partial}{\partial \boldsymbol{\theta}_{\boldsymbol{w}}} \ell(\boldsymbol{\Lambda}|\mathbf{g}_{\text{obs}})||_1 \qquad (3)$$

(presented for the simple case where we have a single sample $\mathbf{g}_{\text{samp}}$ synthesised from the current model, from the previous MLE step). The new filters are indirectly pushed away from existing filters, which would provide little information. Figure 1 compares filters found with and without pre-training and MLE.

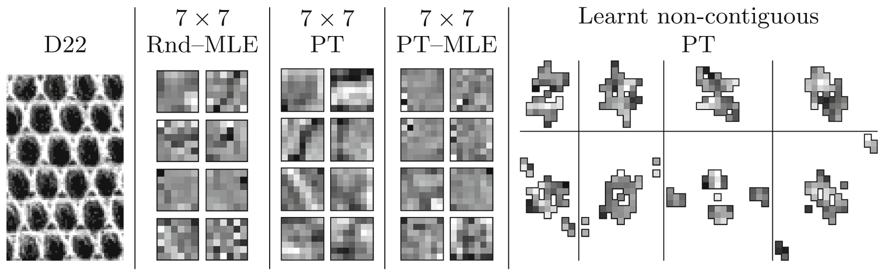

Fig. 1. Examples of learnt filters (GLD filters excluded) for texture D22. Rnd-MLE: starting from random noise then MLE-optimised. PT: pre-learnt filters kept fixed (only natural parameters adjusted during MLE). PT-MLE: pre-learnt then MLE-optimised.

We initialise filters to random noise, and follow the gradient while periodically doubling the number of empirical marginal bins, starting from 4. This allows the coefficients to be tuned to shift the mass of the empirical marginal around and gradually refine it, while overcoming the localised gradient associated with individual empirical marginals. It can be seen as a discrete version of a similar empirical marginal-smoothing technique used in [23].

4 Experimental Results

Models composed solely of GLD filters at appropriate offsets capture a majority of second order interactions and are relatively cheap to sample from. This frees other potentials to focus on more complex structure. The addition of GLD resulted in vastly improved models compared to those with only pairwise or only higher-order potentials. Hence in the majority of experiments we start with a 1st order potential to describe the grey-level histogram and then select 3 GLDs per nesting iteration, of offset up to 40 pixels, before adding filter potentials one at a time. For comparability with [21] and to avoid the randomness of a stopping rule we used the same fixed number of nesting iterations: 8 of GLDs and 8 of filters. Typical learning time for a nested model was 8–11 min (<1 min for nesting with only GLD potentials), mainly spent in the single-threaded Gibbs sampler.

Unfortunately quantitative evaluation of texture synthesis is very difficult, and most texture similarity measures implicitly make a choice of relevant statistics; We agree with Luo et al. [13] that the quantitative measures used in [8] are flawed, and since they are only applicable to highly regular textures as used in [8] and later, visual inspection of results on more challenging textures is necessary. Figure 2 provides comparison to all recent works which gave results for a set of 8 popular regular textures, following the same procedure of downscaling the original texture and using the top half for training; we quantised images to 16 grey levels so that Gibbs sampling could be used. For all other synthesis experiments we used 8 grey levels. Synthesis results comparing different methods of filter learning are shown in Fig. 3. We include a comparison (third column) to FoE-style unnested models by starting with 8 random noise

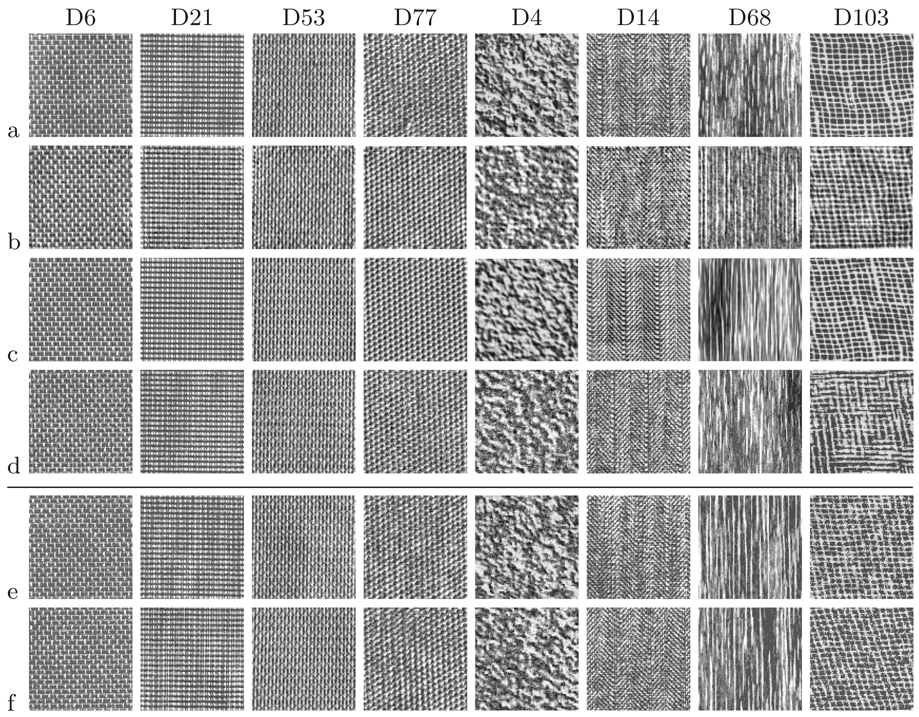

Fig. 2. Comparison of synthesis results against previously published works (images scaled, grey levels reversed and individually renormalised to allow comparison): (a) the eight original 98×98 Brodatz textures; results of (b) Multi-Tm [10] (a single model for all 8 textures); (c) a 2-layer TssDBN [13]; (d) nesting with jagstar-BP[13] (local binary pattern features) [21]; (e) our proposed nested models with 7×7 filters; (f) our proposed nested models with non-contiguous filters.

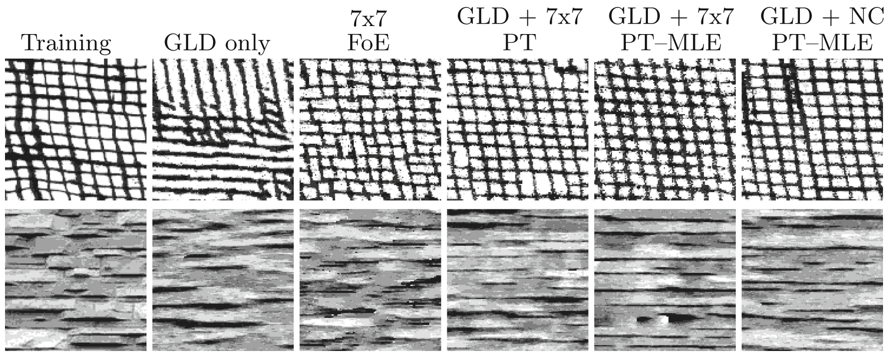

Fig. 3. Synthesis results comparing filter learning approaches. Columns are: *First*: Brodatz D103 and Simoncelli's stone-wall4 (http://www.cns.nyu.edu/~lcv/texture/). *Second*: nested models with 24 GLD potentials. *Third*: FoE-style model with eight 7×7 filter histogram potentials and no pre-training or nesting. *Following*: nested models with 24 GLDs and filters; see Fig. 1 for headings. NC: Non-contiguous filters.

7×7 filters and performing 800 PCD/Gibbs steps (restarting every 200 steps), so that the total computation was about the same as the others. These models without long-range GLD or filter potentials suffer badly when modelling any image detail or structure more than 7 pixels across, producing jumbled images. If the image is regular, adding GLD potentials works very well, but this fix does not work for irregular textures. Figure 4 shows further examples on a broader

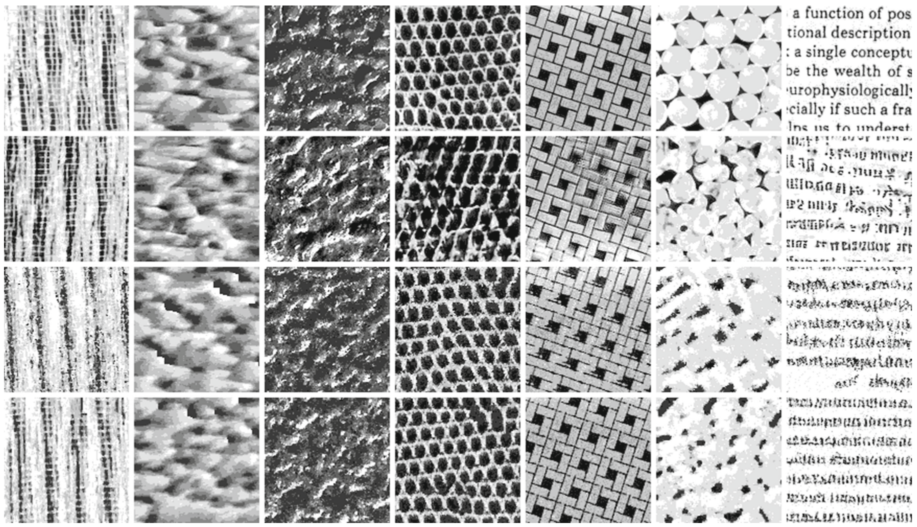

Fig. 4. Synthesis results. *First row*: training images from Brodatz, VisTex and Simoncelli. *Second*: Results by [15] as a baseline (quantised after synthesis). *Third row*: our synthesis results with nested MGRFs, 24 GLDs and square 7×7 filters. *Fourth row*: our results with 24 GLDs and non-contiguous filters.

range of challenging textures. Results for additional textures and models, and source code for the experiments are available on the accompanying website at http://www.ivs.auckland.ac.nz/texture_modelling/sparsefilt/.

5 Conclusions

Our synthesis results, using models with only at most 8 small filters plus simple GLD features and no latent variables, are comparable with those of other models (Fig. 2)—the best performing of which have up to hundreds of filters (in [13] and the multi-texture models of [10]) or up to 4096 parameters per potential [21]—and show ability on irregular textures which are seemingly too difficult for many previous works to attempt. The combination of learning the structure of a MGRF model (as in FRAME) and optimising the feature parameters (as in FoE) appears to allow complex textures to be reproduced more easily by considering a more general set of possible models, although suffering potentially longer learning times due to the iterated nesting procedure.

Filter pre-training instead of starting from random filters was essential for our potentials (results were poor without it), because the gradient can not push mass between histogram bins that are not adjacent, a major disadvantage of using histograms. This is likely a reason that learning sometimes failed to find sensible filters. An alternative parametrisation of the histogram with overlapping coarse and fine bins may provide a more navigable optimisation landscape. Performing MLE learning of filter coefficents gave mixed results, sometimes leading to worse or better results than using purely pre-trained filters, possibly due to the non-convexity of the optimisation problem and the use of histograms. Non-contiguous and contiguous filters also showed tradeoffs; in experiments we found that the noncontiguous ones are more able to handle larger-scale textures, and are more robust because models using square filters will produce bad results if the filters are too small. On the other hand for many textures with small localised details it appeared to be better to use traditional square filters instead. Ideally, such tradeoffs would be made by the nesting algorithm itself, by providing the right stopping rule and regularisation/prior. Unfortunately, this further increases the number of hyperparameters to be tuned, and there are already a relatively large number; there is a tradeoff between selecting model aspects manually (e.g. graphical structure), which may be easier but less robust, and having to tune hyperparameters to select them automatically.

More sophisticated image statistics in [5,14,15], notably filter response inter-dependencies/co-occurrences, have proven to be powerful texture descriptors when used for synthesis of complex and inhomogeneous textures. However, energy-based texture models with such sophisticated statistics are surprisingly yet to be investigated. This paper has focused on learning of linear filters as a case study, but future work should clearly attempt to bridge the gap between these fields by incorporating such kinds of powerful co-occurrence statistics.

References

1. Barndorff-Nielsen, O.: Information and Exponential Families in Statistical Theory. Wiley, Hoboken (1978)
2. Della Pietra, S., Della Pietra, V., Lafferty, J.: Inducing features of random fields. IEEE Trans. Pattern Anal. Mach. Intell. **19**(4), 380–393 (1997)
3. Gao, Q., Roth, S.: How well do filter-based MRFs model natural images? In: Pinz, A., Pock, T., Bischof, H., Leberl, F. (eds.) DAGM/OAGM 2012. LNCS, vol. 7476, pp. 62–72. Springer, Heidelberg (2012). doi:10.1007/978-3-642-32717-9_7
4. Gao, Q., Roth, S.: Texture synthesis: from convolutional RBMs to efficient deterministic algorithms. In: Fränti, P., Brown, G., Loog, M., Escolano, F., Pelillo, M. (eds.) S+SSPR 2014. LNCS, vol. 8621, pp. 434–443. Springer, Heidelberg (2014). doi:10.1007/978-3-662-44415-3_44
5. Gatys, L.A., Ecker, A.S., Bethge, M.: Texture synthesis using convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 262–270 (2015)
6. Gimel'farb, G.: Image Textures and Gibbs Random Fields. Kluwer Academic Publishers, Dordrecht (1999)
7. Hao, T., Raiko, T., Ilin, A., Karhunen, J.: Gated Boltzmann machine in texture modeling. In: Villa, A.E.P., Duch, W., Érdi, P., Masulli, F., Palm, G. (eds.) ICANN 2012. LNCS, vol. 7553, pp. 124–131. Springer, Heidelberg (2012). doi:10.1007/978-3-642-33266-1_16
8. Heess, N., Williams, C.K.I., Hinton, G.E.: Learning generative texture models with extended fields-of-experts. In: Proceedings of British Machine Vision Conference (BMVC 2009), pp. 1–11 (2009)
9. Hinton, G.E.: Training products of experts by minimizing contrastive divergence. Neural Comput. **14**(8), 1771–1800 (2002)
10. Kivinen, J.J., Williams, C.: Multiple texture Boltzmann machines. In: Proceedings of 15th International Conference on Artificial Intelligence and Statistics, vol. 2, pp. 638–646 (2012)
11. Köster, U., Lindgren, J.T., Hyvärinen, A.: Estimating Markov random field potentials for natural images. In: Adali, T., Jutten, C., Romano, J.M.T., Barros, A.K. (eds.) ICA 2009. LNCS, vol. 5441, pp. 515–522. Springer, Heidelberg (2009). doi:10.1007/978-3-642-00599-2_65
12. Lee, S., Ganapathi, V., Koller, D.: Efficient structure learning of Markov networks using L1-regularization. In: Advances in Neural Information Processing Systems 19 (NIPS 2006), pp. 817–824 (2007)
13. Luo, H., Carrier, P.L., Courville, A., Bengio, Y.: Texture modeling with convolutional spike-and-slab RBMs and deep extensions. J. Mach. Learn. Res. **31**, 415–423 (2013)
14. Peyré, G.: Sparse modeling of textures. J. Math. Imaging Vis. **34**(1), 17–31 (2009)
15. Portilla, J., Simoncelli, E.P.: A parametric texture model based on joint statistics of complex wavelet coefficients. Int. J. Comput. Vis. **40**(1), 49–70 (2000)
16. Ranzato, M., Mnih, V., Hinton, G.E.: Generating more realistic images using gated MRFs. In: Advances in Neural Information Processing Systems 23 (NIPS 2010), pp. 2002–2010. Curran Associates (2010)
17. Roth, S., Black, M.J.: Fields of experts. Int. J. Comput. Vis. **82**(2), 205–229 (2009)
18. Schmidt, U., Gao, Q., Roth, S.: A generative perspective on MRFs in low-level vision. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2010), pp. 1751–1758 (2010)

19. Tieleman, T.: Training restricted Boltzmann machines using approximations to the likelihood gradient. In: Proceedings of 25th International Conference on Machine Learning (ICML 2008), pp. 1064–1071. ACM (2008)
20. Versteegen, R., Gimel'farb, G., Riddle, P.: Texture modelling with non-contiguous filters. In: Proceedings of 30th International Conference on Image and Vision Computing New Zealand (IVCNZ 2015) (2015)
21. Versteegen, R., Gimel'farb, G., Riddle, P.: Texture modelling with nested high-order Markov-Gibbs random fields. Comput. Vis. Image Underst. **143**, 120–134 (2016)
22. Zhu, S.C., Wu, Y., Mumford, D.: Minimax entropy principle and its application to texture modeling. Neural Comput. **9**(8), 1627–1660 (1997)
23. Zhu, S.C., Wu, Y., Mumford, D.: Filters, random fields and maximum entropy (FRAME): towards a unified theory for texture modeling. Int. J. of Comput. Vis. **27**(2), 107–126 (1998)