Inference
Assignment 2 in COMS30007 Machine Learning

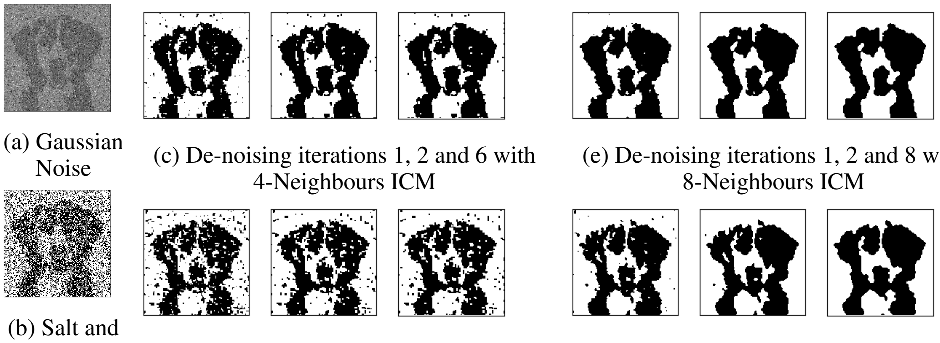
Luke Storry LS14172 **Louis Wyborn** LW15771 **Ahmer Butt** AB15015

Q1 The number of passes through all the nodes in the graphical model (Ising Model) is dependent on our prior assumptions which we encode into certain parameters. A discussion follows. We model pixels in the latent image $\mathbf{x}$ and observed image $\mathbf{y}$ as discrete binary variates that take values of either +1 or -1. Next, we formulate our prior beliefs which we will encode into a model. Firstly, we know that, unless the noise is too large, there is a strong correlation between x_i and corresponding y_i. Secondly, we can assume spatial correlation between neighbouring pixels in the latent space. We can capture the possible probability distributions we are now describing by using a Markov Random Field. This is displayed in Figure 1 of the assignment text. In particular, the Markov blanket of a node x_i is its neighbours and corresponding y_i.

Now, we formulate the joint distribution $p(\mathbf{x}, \mathbf{y})$. By the Hammersley-Clifford Theorem, we know that we can factorise the joint as a product of potential functions of the maximal cliques in the graph, and normalised by the partition function $\mathbf{Z}$. In the Ising Model we define clique potentials of the form $\exp\{-E(C)\}$ for some clique C and energy function E.. There are two types of cliques in the model. Firstly, those of the form x_i, x_j, for any two neighbours. To encode our prior belief of spatial correlation, we choose $E(x_i, x_j) = -\beta x_i x_j$. This gives a low energy (higher probability) when x_i and x_j are the same sign, and vice-versa. Similarly, for cliques of the form x_i, y_i, we choose $E(x_i, y_i) = -\eta x_i y_i$. Thus, computing the energy function takes the form:

$$E(\mathbf{x}, \mathbf{y}) = -\beta \Sigma_{\{i,j\}} x_i x_j - \eta \Sigma_i x_i y_i.$$

This defines a joint distribution $p(\mathbf{x}, \mathbf{y}) = \frac{1}{\mathbf{Z}} \exp\{-E(\mathbf{x}, \mathbf{y})\}$. We now fix the elements of $\mathbf{y}$ to the observed noisy image. We now use the ICM algorithm to find a setting for $\mathbf{x}$ that finds a local maximum for $p(\mathbf{x}, \mathbf{y})$. The number of iterations before convergence depends also on the choice of parameters β and η. Increasing the value of β or η will mean that $p(\mathbf{x}, \mathbf{y})$ has the potential to rise rapidly and reach a local optimum. Therefore, the number of iterations will be less. If β is very high, convergence will be in a location with lots of clustering of same coloured pixel values. If β is very high, convergence will be in a location very similar to the observed image.



(a) Gaussian Noise

(b) Salt and Pepper Noise

(c) De-noising iterations 1, 2 and 6 with 4-Neighbours ICM

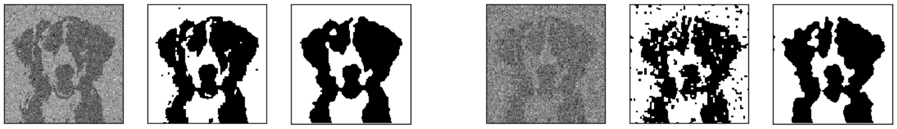
(d) De-noising iterations 1, 2 and 10 with 4-Neighbours ICM

(e) De-noising iterations 1, 2 and 8 with 8-Neighbours ICM

(f) De-noising iterations 1, 2 and 21 with 8-Neighbours ICM

Figure 1: De-noising with ICM after 1 and 2 iterations, and converged.

We begin to reach good results very quickly, especially with 8 neighbours. As demonstrated in Figure 1, after a single iteration the majority of the Gaussian noise has been eliminated, and after 2 iterations there is little left with 4 neighbours, and none with 8 neighbours. We displayed the final converged iteration after this to highlight how similar the de-noised image is after 2 iterations and after it converges.



(a) 70% of pixels modified, sigma=0.5, total iterations=4 and 8 respectively

(b) 100% of pixels modified, sigma=1.0, total iterations=8 and 16 respectively

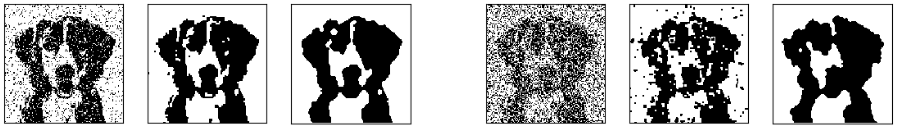
Figure 2: Various levels of Gaussian noise de-noised with 4- and 8-neighbour ICM respectively.

As shown in Figure 2 and 3, as we increase the amount of noise in the images, the number of iterations before convergence increases. This is because, whilst there are lots of speckles, it is likely flipping a speck to become the same as its neighbours will result in an increased probability.

We found the best results with the following values:

- h = 0, ie. no bias in our prior assumption about the value of the latent pixel.
- η = 2.1
- β = 1.5 for 4 neighbours, ie. fewer neighbours, so we weight the sum of their values higher.
- β = 1.1 for 8 neighbours, ie. more neighbours, so we weight the sum of their values lower.

With 8 neighbours the noise is eliminated much more effectively than with 4 neighbours, however fine detail in the original image is lost as it is smoothed out, as large areas of pixels that are the same colour are 'clumped' together.

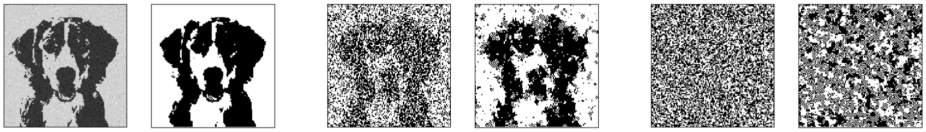


(a) 10% of pixels flipped, total iterations=5 and 13 respectively

(b) 30% of pixels flipped, total iterations=10 and 16 respectively

Figure 3: Various levels of Salt and Pepper noise added, then de-noised with 4-neighbour and 8-neighbour ICM respectively.

Q2



(a) Gaussian noise with 30% proportion and $\sigma = 0.1$.

(b) Salt-n-Pepper noise with 33% proportion.

(c) Salt-n-Pepper noise with 50% proportion.

Figure 4: Running the Gibbs Sampler for on observed images of increasing amounts of noise. Results shown after running for 100 burn-in iterations plus another 30 iterations.

Figure 4a shows an image with 20% proportion and $= 0.1$. The Gibbs sampler clearly removes most of the noise. Nearly all of the noise 'specks' are removed and the structure of the black area given by the dog remains intact. This is because of our prior assumptions of correlation between adjacent nodes and of correlation between corresponding nodes in the observed image. However, if we parameterised our priors poorly, e.g. if we had chosen our weight parameters w_{i_j}) poorly, then the posterior which we sample from will give a probability distribution which does not encode this well, or even encodes an inverse correlation.

Figure 4b shows the results after applying significantly high salt-n-pepper noise, at 33%. We see that most of the noise is still removed, due to encoding spatial correlation in latent space. We also observe that the structure of the dog is somewhat recovered, even though details are not able to be recovered. This is recovered from the faintly darker areas in the noisy image. It occurs because x_i values correlate with y_i values in the noisy image. Therefore, in these areas, more latent pixels will be sampled as black. Then, due to spatial correlation, these areas will cluster. Clustering will then tend to stop at the edges of the darker regions in the noisy image because the correlation with y_i will no longer more likely imply black.

The sampler cannot recover images where there is too much noise. In the worst case, there would be 50% salt-n-pepper applied and hence the observed image would be pure noise, so it becomes impossible to recover any latent image. This is shown in Figure 4c. One interesting observation is that the recovered image forms clusters of noise. This is again because of the prior assumption of correlation between adjacent nodes.

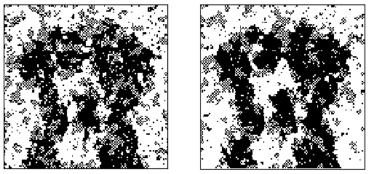


Figure 5: Gibbs sampler iterating in sequential order through the image (left) vs random order (right).

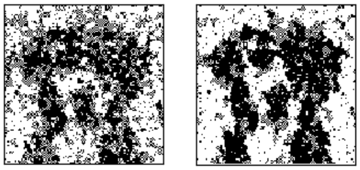


Figure 6: Gibbs sampler for $N = 15$ iterations without burn-in (left), and running $K = 40$ burn-in iterations first (right)

Q3 Cycling through the nodes in random order increases the stochasticity of the samples. Therefore, it is more likely for the samples drawn to better reflect the true posterior distribution. As seen in Figure 5, the image recovered using sequential sampling includes more noise. More iterations are needed, i.e. more samples drawn, for the image to reach the same location. Iterating in sequential order means we will always draw samples for pixels towards the end of the image at a later time step than those at the start. This results in an inherent bias in the approximation of the posterior distribution. Hence, the optimum point we reach will be somewhat shifted from the optimum point of the unbiased approximation. Therefore, the recovered latent image will probably be more noisy.

Q4 After many iterations, the Markov Chain over our image space will enter its stationary distribution. Therefore, after this point the image should converge and so continuing to sample from this point will just give us an image representation more accurate to the true posterior. If the initial representation is in a particularly low probability location, then samples from these early time steps in the chain may remain in the recovered image. To obtain a better representation, we wait a certain amount of iterations before collecting samples. This is known as the burn-in time. Including any artifacts from the initial state is now avoided. This is illustrated in Figure 6. Both images are run from an initialisation of $\mathbf{x}$ to random values of +1 or -1. The results shown are after 15 iterations. However, on the right, we first run a number of burn-in iterations. As can be seen, there is far less noise corresponding to no noisy artifacts remaining from the beginning samples.

Q5 Kullback-Leibler divergence is not symmetric. Therefore, using it as a metric when finding a distribution $q(x)$ that optimally fits $p(x)$ can have different results depending on whether we decide to minimise the forward-KL divergence $KL(p(x)||q(x))$ or reverse-KL divergence $KL(q(x)||p(x))$.

First with forward-KL: $KL(p||q) = \sum_x p(x) ln \frac{p(x)}{q(x)}$. We can see that this tends to infinity as $q(x)$ tends to zero whilst $p(x) > 0$. Thus to minimise forward-KL, $q(x)$ must spread its probability mass out to be positive in all places where $p(x)$ is positive. This is known as zero-avoiding for $q(x)$, where $q(x)$ attempts to 'cover' $p(x)$.

Alternatively, with reverse-KL: $KL(q||p) = \sum_x q(x) ln \frac{q(x)}{p(x)}$. This tends to infinity when $p(x) = 0$ and $q(x) > 0$, so to minimise this we must ensure that $q(x) = 0$ in all places where $p(x) = 0$, also known as zero-forcing.

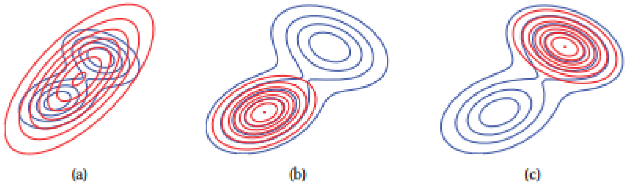
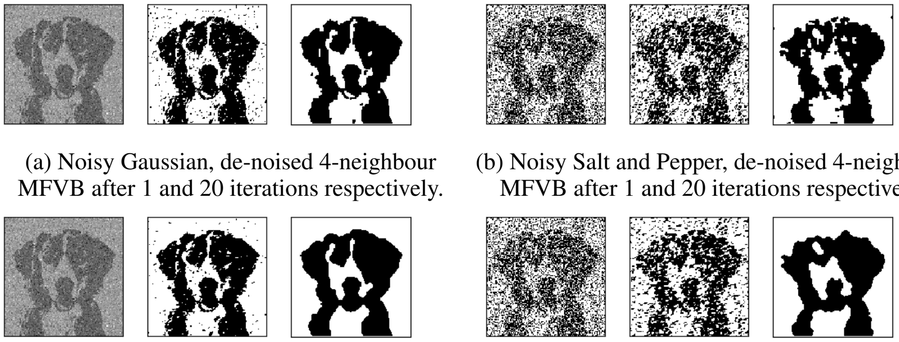


Figure 7: Illustrating forwards vs reverse KL on a bimodal distribution, Figure 21.1 from [11]

This difference between minimising forwards- and backwards-KL when finding an 'optimal' approximation is nicely demonstrated in Figure 7. The red curves are the contours of three different unimodal distributions $q(x)$. These are approximations of the bimodal distribution $p(x)$, which is displayed in blue. The first, (a), is the result of minimising forwards-KL, and shows what is meant by 'zero-avoiding': the red $q(x)$ covers $p(x)$, avoiding any $p(x) > 0 \wedge q(x) = 0$. The other two, (b) and (c), show two different outcomes of minimising reverse-KL, where $q(x)$ 'locks onto' one of the two peaks in $p(x)$, ensuring that $q(x) = 0 \ \forall p(x) = 0$, i.e. zero-forcing.

Q6 Algorithm 4 was used to build a Variational Bayes de-noiser for Ising Models, see `6.ipynb` for code and Figure 8 for sample results.



(a) Noisy Gaussian, de-noised 4-neighbour MFVB after 1 and 20 iterations respectively.

(b) Noisy Salt and Pepper, de-noised 4-neighbour MFVB after 1 and 20 iterations respectively.

(c) Noisy Gaussian, de-noised 8-neighbour MFVB after 1 and 20 iterations respectively.

(d) Noisy Salt and Pepper, de-noised 8-neighbour MFVB after 1 and 20 iterations respectively.

Figure 8: Mean Field Variational Bayes de-noising.

Q7 The resulting images from MFVB have fewer 'speckles' than the resulting images from ICM, highlighted in the difference between Figure 8b and Figure 3b for 4 neighbours. Here the MFVB results in a clean white background and a clean black foreground, whereas the ICM converges to an image with 'speckles' breaking up the large areas of the same colour.

The results from the Gibbs Sampler are 'blurrier' than the results from the MFVB, with less defined edges to shapes within the images. This is because the mean field tends to be overconfident, resulting in less uncertainty around shape boundaries [11]. This is highlighted in the difference between Figure 4b and Figure 8b.

ICM is also very prone to converging to a local optima of the probability distribution. However, the Gibbs sampling method is not. As seen in Q4, the algorithm will converge to a good image even when initialising $\mathbf{x}$ to random values.

Stochastic inference (Gibbs sampling) is faster than deterministic inference (MFVB) for large models such as high resolution images. Therefore, in the same amount of time, Gibbs sampling could give a better result. However, for the small images considered here, MFVB is faster. The deterministic property of MFVB may also be considered advantageous because it allows the use of back-propogation to determine the gradient of the variational estimate with respect to the model parameters[13].

Q8 We have implemented a image-segmentation system that uses Iterative Conditional Modes to separate an image into a foreground and background, find the average colour of each segment, and export a two-tone image. (See the attached `8.ipynb`).

Whether a pixel is labelled as foreground or background is the Latent Variables, encoded as masks over the image. Lots of experimentation was done to determine the best approach for initialising the masks. Things such as random assignment did not work for obvious reasons and methods such as labelling inner and edge pixels as foreground and background had limited success with our variety of test images. Our final chosen prior of thresholding intensity values to encodes our assumption that the foreground object will usually by slightly darker or lighter than the background.

Then, as with ICM, we fix all the pixels as final, apart from one. Using the foreground and background masks, the image is split into two sets of pixels, and the red green and blue histograms are calculated for each. The testing pixel's RGB values are then compared to these histograms by finding the bin that pixel would have been sorted into, and dividing the area of that bin by the total area of all the bins in the histogram. (This effectively uses the histograms as probability mass distribution look-up tables.) These RGB probabilities are then averaged to give the probabilities of the pixel belonging to either the foreground or the background. The pixel is then reassigned to the mask with the highest probability, and the histograms all updated before moving onto the next pixel. Once every pixel has been processed in this way, then

The intensity thresholding can initially form a very noisy image but, after around 10 iterations. The results will usually stabilise, as can be seen in Figure 9. However, sometimes under-segmentation will occur and the detected image will all be one grey blob.

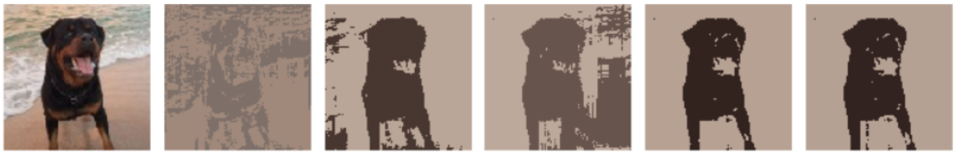


Figure 9: Image Segmentation, after 0, 1, 2, 3, 5, and 10 iterations

On some images with diverging gradients (such as a blue sky turning white), under-segmentation was often caused by the linear ordering with which we looped through the pixels. As detailed above, every time a new pixel is added to one of the masks, that mask's histogram is updated to include that pixel. Over time, as each successive neighboring pixel is added, this can skew the histogram to accept a wider variety of pixels than it would have if the pixels were tested and added in a different order.

For some things, such as skies that have these gradients, this is a good effect, however it created problems for some images, such as a brown dog sat in a field of dark green grass. In an attempt to solve this issue, an option was added to our system to shuffle the order in which the pixels are tested.

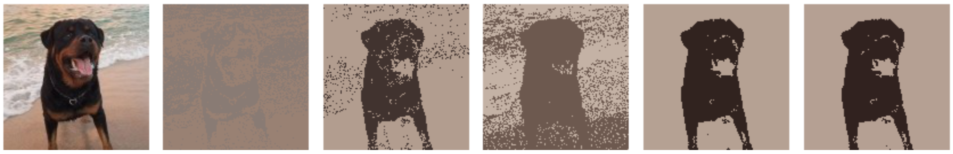


Figure 10: Image Segmentation with Random ordering, after 0, 1, 2, 3, 5, and 10 iterations

As can be seen from Figure 10, for some images the results are initially different but usually converge to the same output after 10 iterations. The random ordering of pixels helps for some images, and hinders for others, we although we left it as an option in our system, we did not make it the default.

References

[1] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006. ISBN: 0387310738.

[2] Claudia Blaiotta, Manuel Jorge Cardoso, and John Ashburner. "Variational inference for medical image segmentation". In: *Computer Vision and Image Understanding* 151 (2016), pp. 14–28. DOI: 10.1016/j.cviu.2016.04.004. URL: https://doi.org/10.1016/j.cviu.2016.04.004.

[3] Andrew Blake, Pushmeet Kohli, and Carsten Rother. *Markov Random Fields for Vision and Image Processing*. The MIT Press, 2011.

[4] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. "Variational Inference: A Review for Statisticians". In: *Journal of the American Statistical Association* 112.518 (2017), pp. 859–877. DOI: 10.1080/01621459.2017.1285773. eprint: https://doi.org/10.1080/01621459.2017.1285773. URL: https://doi.org/10.1080/01621459.2017.1285773.

[5] H. Derin and H. Elliott. "Modeling and Segmentation of Noisy and Textured Images Using Gibbs Random Fields". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 9.1 (Jan. 1987), pp. 39–55. ISSN: 0162-8828. DOI: 10.1109/TPAMI.1987.4767871. URL: http://dx.doi.org/10.1109/TPAMI.1987.4767871.

[6] Stuart Geman and Donald Geman. "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 6.6 (Nov. 1984), pp. 721–741. ISSN: 0162-8828. DOI: 10.1109/TPAMI.1984.4767596. URL: http://dx.doi.org/10.1109/TPAMI.1984.4767596.

[7] Stuart Geman and Christine Graffigne. "Markov random field image models and their applications to computer vision". In: *Proceedings of the International congress of mathematicians 1986 Ed.* American Mathematical Society. Berkeley, California, 1987, pp. 1496–1517.

[8] Dandan Hu, Peter Ronhovde, and Zohar Nussinov. "A Replica Inference Approach to Unsupervised Multi-Scale Image Segmentation". In: *CoRR* abs/1106.5793 (2011). arXiv: 1106.5793. URL: http://arxiv.org/abs/1106.5793.

[9] Michael I. Jordan et al. "Learning in Graphical Models". In: ed. by Michael I. Jordan. Cambridge, MA, USA: MIT Press, 1999. Chap. An Introduction to Variational Methods for Graphical Models, pp. 105–161. ISBN: 0-262-60032-3. URL: http://dl.acm.org/citation.cfm?id=308574.308660.

[10] Z. Li et al. "A variational inference based approach for image segmentation". In: *2008 19th International Conference on Pattern Recognition*. Dec. 2008, pp. 1–4. DOI: 10.1109/ICPR.2008.4761226.

[11] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[12] Anand Rangarajan, Rama Chellappa, and Anand Rangarajan. *Markov Random Field Models in Image Processing*. 1995.

[13] Veselin Stoyanov, Alexander Ropson, and Jason Eisner. "Empirical Risk Minimization of Graphical Model Parameters Given Approximate Inference, Decoding, and Model Structure". In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*. Ed. by Geoffrey J. Gordon, David B. Dunson, and Miroslav Dudík. Vol. 15. JMLR Proceedings. JMLR.org, 2011, pp. 725–733. URL: http://www.jmlr.org/proceedings/papers/v15/stoyanov11a/stoyanov11a.pdf.

[14] Jozef Strecka and Michal Jascur. *A brief account of the Ising and Ising-like models: Mean-field, effective-field and exact results*. Nov. 2015. arXiv: 1511.03031. URL: http://arxiv.org/abs/1511.03031.

[15] Michael J. Swain and Dana H. Ballard. "Color indexing". In: *International Journal of Computer Vision* 7.1 (Nov. 1991), pp. 11–32. ISSN: 1573-1405. DOI: 10.1007/BF00130487. URL: https://doi.org/10.1007/BF00130487.

[16] Michael J. Swain and Dana H. Ballard. "Indexing via Color Histograms". In: *Active Perception and Robot Vision*. Ed. by Arun K. Sood and Harry Wechsler. Berlin, Heidelberg: Springer Berlin Heidelberg, 1992, pp. 261–273. ISBN: 978-3-642-77225-2. DOI: 10.1007/978-3-642-77225-2_13. URL: https://doi.org/10.1007/978-3-642-77225-2_13.

[17] Martin J. Wainwright and Michael I. Jordan. “Graphical Models, Exponential Families, and Variational Inference”. In: *Foundations and Trends in Machine Learning* 1.1fffdfffdfffd2 (2008), pp. 1–305. ISSN: 1935-8237. DOI: 10.1561/2200000001. URL: http://dx.doi.org/10.1561/2200000001.

[18] B. Walsh. *Markov Chain Monte Carlo and Gibbs Sampling*. 2004.