Image Inference

Themis Papathemistocleous[1] and Rob Robinson [2]

[1] *35458*
[2] *11977*

December 7, 2018

Abstract

A noisy image defined as a grid of pixels y_i can be recovered to its underlying latent pixel representation x_i using a range of different techniques. This paper assesses the ability of a range of techniques to restore an underlying image.

Inference

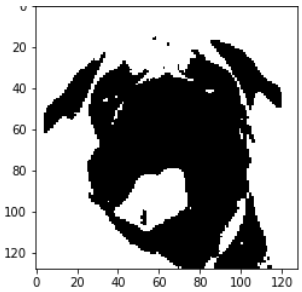

Figure 1: *Target Image*

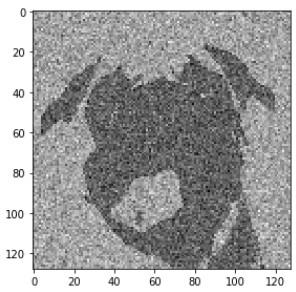

Figure 2: *Noisy Image*

A black and white image (**Figure 1**) is altered using Gaussian noise (**Figure 2**). The methods below aim to restore the original image using a range of techniques.

1 Approximate Inference

1.1 Iterative Conditional Modes (ICM)

Assuming the pixels to be conditionally independent given the latent variable x. The posterior distribution $p(x|y)$ can be obtained through Baye's rule (Equation 1). Even for a small binary image this calculation is computationally intractable and conjugacy can not be exploited, approximations of the integral must be made to infer the true value of each pixel.

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} \tag{1}$$

ICM determines whether a given pixel in an image should be altered by assigning an associated energy derived from E **Equation 2**. This is inversely proportional to the probability. y_i is the observed noisy image and x_i is the unknown noise free image. Assuming the level of noise to be low, there is a strong correlation between x_i and y_i. When recovering the image the strength of this correlation is approximated by η (positive constant). When $y_i = x_i$ the energy E is low, resulting in a high probability and so unlikely to change. When $y_i \neq x_i$ the reverse is the case and the associated energy is much higher. There is also a strong correlation between x_i and x_j (neighbouring pixels). This strength of the relationship is defined by β. Again when the pixel values are equivalent yields a low energy and when they differ yields a high energy. h can be adjusted to provide a bias towards a pixel being black or white. ICM uses a comparison of energies and so the ratio between β and η determines the performance of the algorithm, **Figure 3**. However, the optimal ratio varies for different levels of noise in an image, **Figure 4**. In general, when β is significantly larger than η the approximate outline of the image is recovered however, any finer detail is lost e.g. separation of right ear in **Figure 5**. When η is significantly larger than β the reverse is the case and the converged solution is obtained with fewer iterations however, there is still residual noise in the image (**Figure 6**).

$$E(x, y) = h \sum_i x_i - \beta \sum_{\{i,j\}} x_i x_j - \eta \sum_i x_i y_i \tag{2}$$

Gaussian Noise was applied to an entire image with increasingly larger variances of noise. ICM was applied

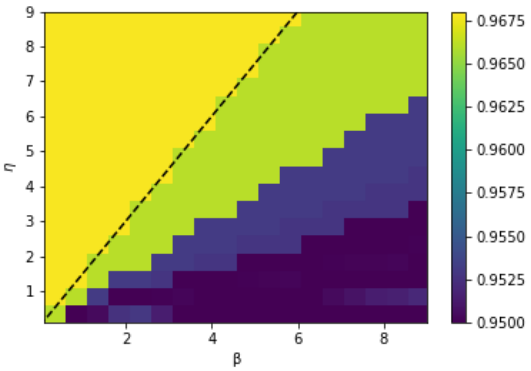

Figure 3: *% Image Restoration for varying levels of β and η*

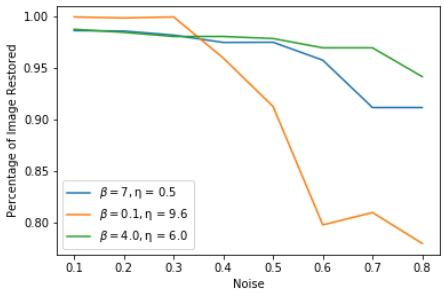

Figure 4: *% Image Restoration for varying Gaussian noise for ICM sampling*

to restore the image with a maximum of 25 iterations since after this point the number of pixel changes is negligible (**Figure 7**). **Table 1** summarises the results with $\beta = 4$ and $\eta = 6$. It is clear to see the number of neighbours is directly proportional to the computational cost since the larger the number of neighbours, the longer the solution takes to converge and the more calculations required during each iteration. However, for an image with Gaussian noise, the greater the number of neighbours the closer the resultant image is to the original. This becomes more significant as the amount of noise in the image increases. Additionally, using 8 neighbours over 4 neighbours results in an image with smoother edges and less noise, clearly visible when comparing the ears in **Figures 8** & **9**.

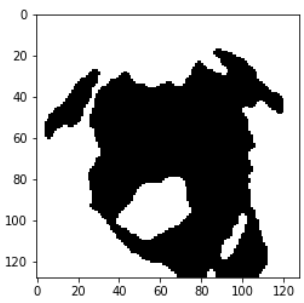

Figure 5: *ICM β : 10, η : 0.1*

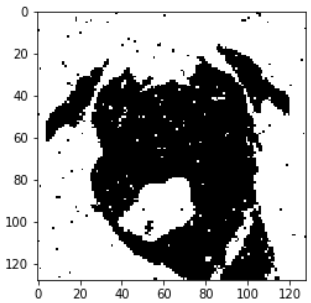

Figure 6: *$ICM\beta$: 0.1, η : 5*

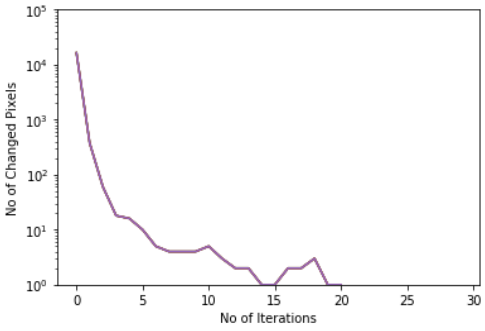

Figure 7: *Number of Pixels changing for each iteration of ICM algorithm*

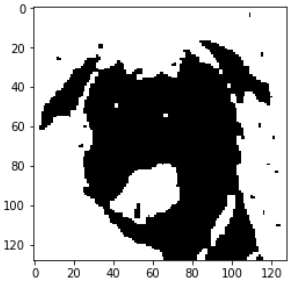

Figure 8: *ICM 4 Neighbours*

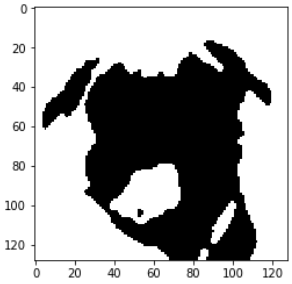

Figure 9: *ICM 8 Neighbours*

Table 1: *ICM Image Recovery Results*

	4 Neighbours		8 Neighbours	
Noise	% Pixels	Iter	% Pixels	Iter
0.1	98.72	4	99.34	5
0.3	98.35	5	98.41	11
0.5	97.29	5	97.00	15
0.7	92.61	5	94.15	20
0.9	81.16	5	92.56	20

Stochastic Inference

2 MCMC - Gibbs Sampler

The results from implementing a Gibbs sampling Markov Chain Monte Carlo (MCMC) algorithm are displayed in Table **2**. Generally speaking the % Pixel restoration remains high even with significant levels of noise as can be seen in **Figure 10** . Gibbs sampling can be applied in a sequential or random order. Using a random order requires more iterations in order to achieve suitably low levels of noise however the resultant image is a closer representation of the desired image Figures **11** & **12**. In general, as the number of iterations is increased the levels of noise in the image decreases. However as the number of iterations continues to increase, the more granular detail is smoothed out and the resultant image becomes less similar to the target. This can be seen on the right ear of **Figure 13** compared to **Figure 14**. This smoothing occurs more quickly for ordered sampling compared to ran-

dom. Note this subtly is not conveyed when solely examining the percentage of particles restored to the original and as such this must not be the sole metric when evaluating performance of image inference.

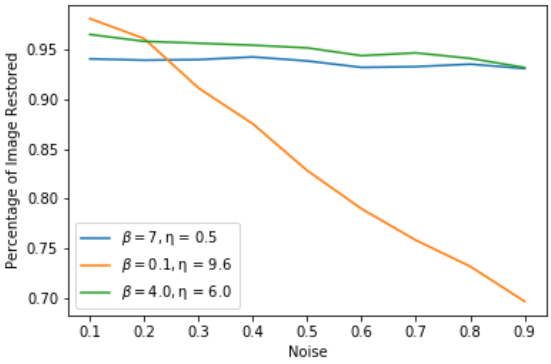

Figure 10: *% Image Restoration for varying Gaussian noise for Gibbs sampling*

Table 2: *MCMC Results using Sequential (S) and Random (R) Gibbs Sampling for varying numbers of iterations*

	% Particles Restored					
Iterations	1		5		20	
Noise	S	R	S	R	S	R
0.1	100	63	100	98	100	98
0.3	98	62	98	96	99	97
0.5	97	59	97	94	97	95
0.7	93	55	94	89	94	91

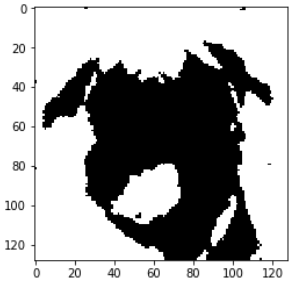

Figure 11: *Gibbs: Ordered*

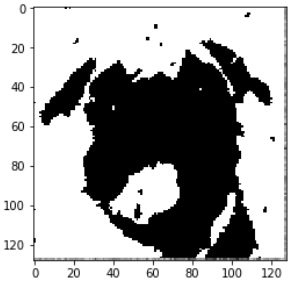

Figure 12: *Gibbs: Random*

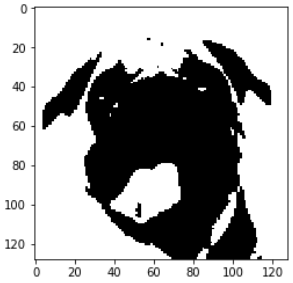

Figure 13: *Gibbs: 1 Iter*

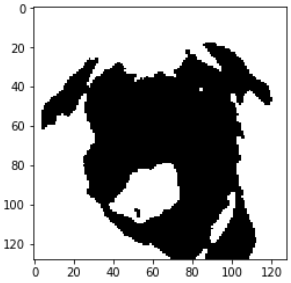

Figure 14: *Gibbs: 30 Iter*

3 Variational Bayes

3.1 Kullback–Leibler(KL) divergence

Variational Bayes aims to find a distribution $q(\mathbf{x})$ that maps as closely as possible to the original distribution $p(\mathbf{x})$, in doing so defines a bound to the previously intractable evidence. KL divergence can be used to compare how different probability distributions are from each other, but is not considered a statistical metric since it is not symmetric.

In Forward KL (Equation 3) a probability mass is placed on $q(\mathbf{x})$ at any point where $p(\mathbf{x})$ is non zero. When $p(\mathbf{x}) = 0$ KL returns 0 irrespective of $q(\mathbf{x})$. Assuming $p(\mathbf{x}) > 0$ when $|q(\mathbf{x})\text{-}p(\mathbf{x})|$ is large the returned KL divergence becomes large. These properties ensure that the distribution $q(\mathbf{x})$ is spread across the entire range of $p(\mathbf{x})$ even if the representation is poor.

$$KL(p(\mathbf{x})||q(\mathbf{x})) = \int p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})} dx \tag{3}$$

In contrast, Reverse KL 4 only places probability mass on $q(\mathbf{x})$ when its representation of $p(\mathbf{x})$ is accurate. When $q(\mathbf{x}) > 0$ the log term must be minimised to ensure minimum KL divergence. When $q(\mathbf{x}) = 0$ returns 0 KL divergence and so $p(\mathbf{x})$ can be ignored without incurring a large KL value. As a result Reverse KL will not necessarily place probability weighting across all non zero $p(\mathbf{x})$.

$$KL(q(\mathbf{x})||p(\mathbf{x})) = \int q(\mathbf{x}) \log \frac{q(\mathbf{x})}{p(\mathbf{x})} dx \tag{4}$$

3.2 Results

Figures 16, 28 & 18 show the results from using Variational Bayes on a range of noise types.

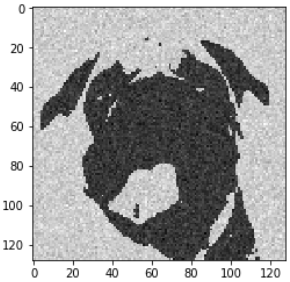

Figure 15: *Gaussian Noise* $\sigma^2 : 0.1$, $prop : 1$

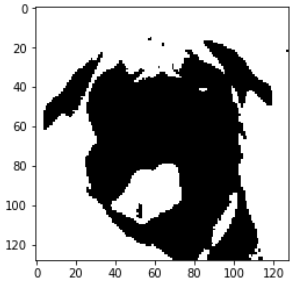

Figure 16: *Variational Bayes* $\beta : 1.2$, $\eta : 10$

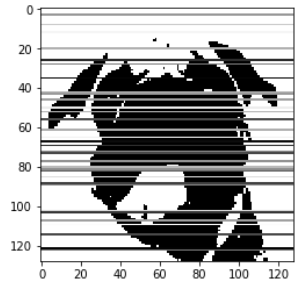

Figure 17: *Line noise prop* : 0.7

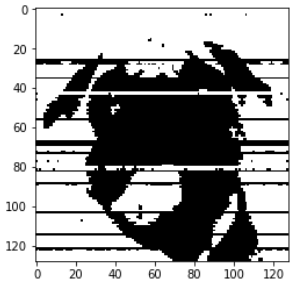

Figure 18: *Variational Bayes* $\beta : 1.2, \eta : 10$

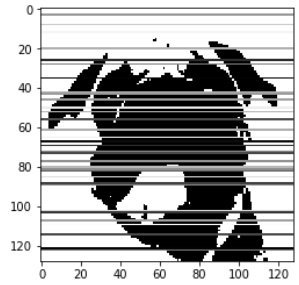

Figure 19: *Line noise prop* : 0.7

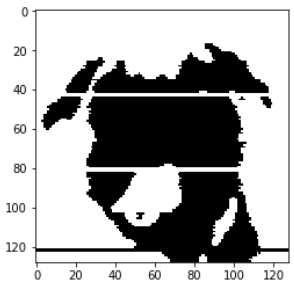

Figure 20: *ICM* $\beta : 4, \eta : 6$

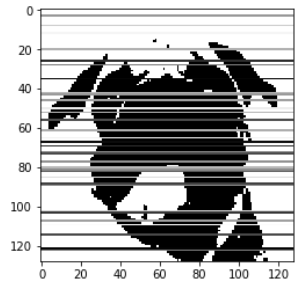

Figure 21: *Line noise prop* : 0.7

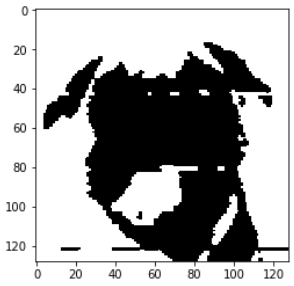

Figure 22: *Gibbs sampling* $\beta : 1.2, \eta : 10$

4 Comparison

Variational Bayes, is fast at estimating a distribution and able to cope with the images that have low variance and small disturbance in the image such as Gaussian noise shown in **Figures 15 and 16**. However an image that has black lines drawn on top (**Figure 17**) places a structure to the image that Variational Bayes might take into consideration as part of the underlying model since it is only trying to approximate and find a bound that is as close to the model as possible (**Figure 18**). Its benefit is relatively fast, requring minimal iterations even for images with large amounts of Gaussian noise.

In contrast, Gibbs sampling while more computationally intensive, is more exact in approximating the underlying model and in the case of adding lines on a model, it is able to remove most of them (**Figure 22**), showing that it is able to find a more accurate approximation of the original image. This is the main benefit but its drawback is that it needs a lot of time to compute, and therefore in large data samples, Variational Bayes is more effective.

The first method used, called Iterative Conditional Modes (ICM), is basically an energy optimisation method, initially introduced for Markov Random Field in the field of physics. However it has various applications in computer vision and image processing as well. The main benefits of using this, is the fact that it is simple and efficient, it can very quickly return a result. On the other hand its accuracy is worse than Gibbs Sampling methods for high noise/complex images as can be seen from comparing **Figures 4 & 10**. There are two main reasons for that. One is initialisation, ICM is heavily dependent on label initialisation compared to the other methods and therefore if the initialisation is bad, the error can spread through the iterations and provide very inaccurate results. The second reason is local convergence, compared to the other methods, ICM produces decent results only when the number of local optimum is minimal, but when there are a lot of local optimum, which is most of the cases, ICM return early because it gets stuck on one of the local optimum instead of finding the global optimum.

If an image has been made noisy through salt and pepper the correlation between y_i and x_i is significantly lower and therefore yields inaccurate results under the same parameter weightings tailored for Gaussian noise.

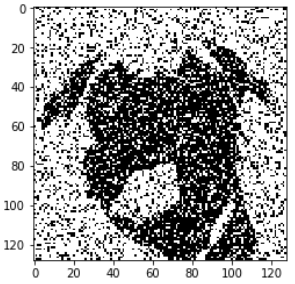

Figure 23: *Salt and Pepper* $varsigma : 0.3$ $prop : 0.3$

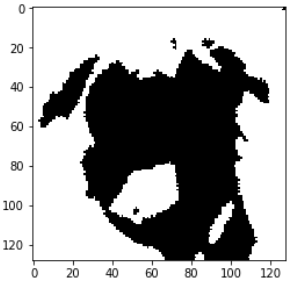

Figure 24: *ICM* $\beta : 4, \eta : 6$

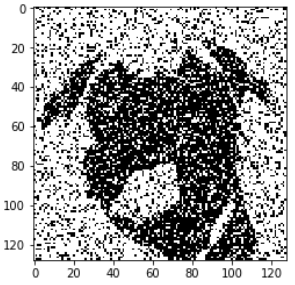

Figure 25: *Salt and Pepper* $varsigma : 0.3$ $prop : 0.3$

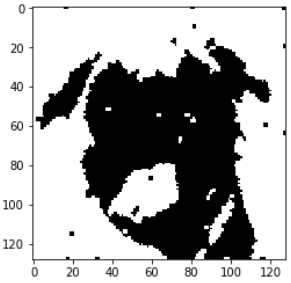

Figure 26: *Gibbs sampling* $\beta : 1.2, \eta : 10$

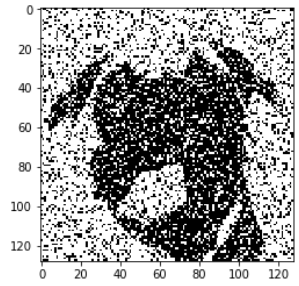

Figure 27: *Salt & Pepper,* $prop : 0.3$

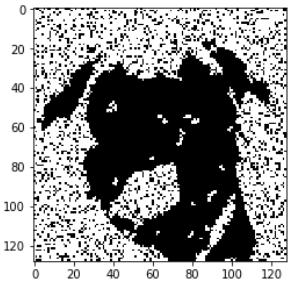

Figure 28: *Variational Bayes* $\beta : 1.2$*,* $\eta : 10$

5 Image Segmentation

Even though our implementation of image segmentation did not produced the desired results, we managed to understand the idea behind how using a few tricks we could change our previous implementations into image segmentation.

At the start we run a k-means algorithm on the coloured image for 20 clusters. The resulting cluster centroids show RGB colour values that can describe the image in a good way and therefore can be used as an approximation so that we don't use all the different colour values our image can have. Together with the k-means centroids, we have to provide the algorithm with two masks, one showing some of the foreground pixels, and another showing some of the background pixels. This centroids then are used in conjunction with the two masks to produce two distinct histograms, one for the foreground probability and one for the background probability. To produce the histograms we get every pixel value for our two masks and find the bin that matches as closest to that pixel colour by calculating the euclidean distance between the two RGB vectors. Then divide each histogram by the sum of its values to get a probability histogram.

After acquiring the two histograms, we combined it with variational Bayes by changing the likelihood function by again finding for each pixel, the bin that is closest to and accessing the histogram to get the probability that that pixel comes from the foreground and the background.

6 Variational Auto-encoder (VAE)

6.1 Difference in inference compared to Variational Bayes

The difference between Variational Bayes and variational auto-encoders (VAE), is the way in which they try to solve problem of optimising the KL divergence. VAE tries to encode the latent variable $\mathbf{z}$ in a lower dimensionality representation. It assumes that there is no simple interpretation of the dimensions of $\mathbf{z}$ and instead assumes that samples of $\mathbf{z}$ can be drawn from a simple distribution such as $\mathcal{N}(0, \mathcal{I})$. Then by providing a function that maps the independent, normally distributed $\mathbf{z}$ values to whatever latent variables might be required for the model using a decoder function for example mapping them to a Bernoulli distribution. It is worth noting that some information is lost since we map from the encoded lower dimensionality data back to a higher dimensionality data.

Basically the approach used is to get an inference model that learns to approximate the intractable posterior by optimising the variational lower bound. The better the representation of the data that it learns, then the better the result.