CS5340 Project

Wu Wenqi
A0124278A

1 Data denoising

For both denoising algorithms, the image is represented as an Ising model as shown in Figure 1, where every pixel and every observation of a pixel are represented by nodes X_i (unshaded) and Y_i (shaded) respectively. Every pixel X_i depends only on $nbr(i)$ and Y_i, where $nbr(i)$ are the neighboring pixels of X_i and Y_i is the observation of X_i. Every observation of a pixel Y_i only depends on the pixel X_i.

Both algorithms accept the image to be denoised as a text file, where each row contains the coordinates and color of a pixel observation Y_i. Since the image is black and white, every pixel has a single color channel with value either 0 or 255. We represent 0 and 255 to be -1 and 1 respectively in the Ising model.

Let X and Y be the sets containing the pixel nodes and pixel observation nodes respectively. Y is fully observed. The aim of our denoising algorithms is to find a good approximate configuration for X, given Y.

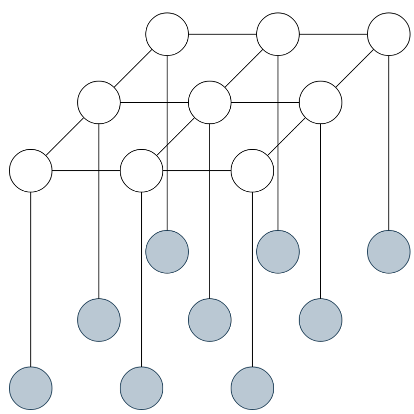


Figure 1: Ising model representation of binary images

1.1 Gibbs sampling algorithm

Every X_i node is initialized to y_i, which is the observed value of Y_i. The pairwise potential term ψ_{st} between nodes X_s and X_t is given by:

$$\psi(x_s, x_t) = \exp(J x_s x_t) \tag{1}$$

where J is the coupling strength. The local evidence term is given by:

$$\psi_t(x_t) = \mathcal{N}(y_t | x_t, \sigma^2) \tag{2}$$

Finally, the full conditional probability of a node X_t is given by:

$$p(x_t|x_{-t}, y, \theta) = p(x_t|nbr(t), y_t, \theta) \tag{3}$$
$$= \frac{\psi_t(x_t)\prod_{s\in nbr(t)}\psi_{st}(x_s, x_t)}{\sum_{x_t}\psi_t(x_t)\prod_{s\in nbr(t)}\psi_{st}(x_s, x_t)} \tag{4}$$

The pseudocode is as follows:

for all $X_i \in X$ **do**
  $X_i \leftarrow Y_i$
end for
$t \leftarrow 0$
while $t < T$ **do**
  for all $X_i \in X$ **do**
    $X_i \leftarrow$ sampled value from $p(X_t|nbr(t), y_t, \theta)$
  end for
  $t \leftarrow t + 1$
end while

Here, T is the number of iterations. For our case, T is set to 20. The coupling strength J and local evidence term variance σ^2 are hyperparameters, and are both set to 1.

1.2 Variational inference algorithm

The prior has the form:

$$p(x) = \frac{1}{Z_0}\exp(-E_0(x)) \tag{5}$$
$$\text{where } E_0 = -\sum_{i=1}^{|X|}\sum_{j\in nbr(i)} W_{ij}x_ix_j \tag{6}$$

where W_{ij} is the pairwise smoothness term between nodes X_i and X_j. The likelihood has the form:

$$p(y|x) = \prod_i p(y_i|x_i) \tag{7}$$
$$= \exp\left(\sum_i -L_i(x_i)\right) \tag{8}$$
$$\text{where } L_i(x_i) = -\log\left(\mathcal{N}\left(y_i|x_i, c^2\right)\right) \tag{9}$$

Hence, the joint probability has the form:

$$p(x, y) = p(y|x)p(x) \tag{10}$$
$$= \frac{1}{Z_0}\exp(-E_0(x) - \sum_i L_i(x_i)) \tag{11}$$

We approximate $p(x, y)$ with $q(x)$. Under mean field theory, we can factorize $q(x)$ to get the following form:

$$q(x) = \prod_{i}^{|X|} q_i(x_i) \tag{12}$$

The optimized factor $q_i(x_i)$ can be found to be proportional to $\exp\left(x_i \sum_{j\in nbr(i)} W_{ij}\mu_j + L_i(x_i)\right)$, where $\mu_j = \sum_j x_j q(x_j)$ is the mean value of node X_j.

Let $m_i = \sum_{j\in nbr(i)} W_{ij}\mu_j$ to be the mean field influence on node X_i. The approximate marginal posterior is given by:

$$q_i(x_i = 1) = \text{sigmoid}(2a_i) \tag{13}$$

$$q_i(x_i = -1) = \text{sigmoid}(-2a_i) \tag{14}$$

Here, $a_i = m_i + 0.5(L_i^+ - L_i^-)$, where $L_i^+ = L_i(+1)$ and $L_i^- = L_i(-1)$. Finally, the mean value μ_i is given by:

$$\begin{aligned} \mu_i &= \mathbb{E}[x_i] && (15)\\ &= q_i(x_i = +1)\cdot(+1) + q_i(x_i = -1)\cdot(-1) && (16)\\ &= \tanh(a_i) && (17) \end{aligned}$$

Similar to Gibbs sampling, we initialize μ_i with the value y_i for all $X_i \in X$. Then, we iteratively update the value of μ_i using the equation above. The pseudocode is as follows:

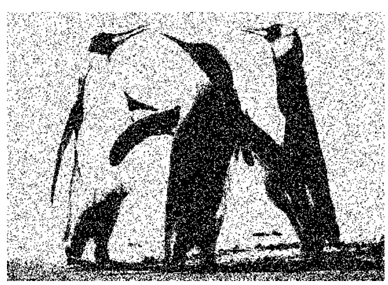
for all $\mu_i \in M$ **do**
    $\mu_i \leftarrow Y_i$
end for
$t \leftarrow 0$
while $t < T$ **do**
    for all $\mu_i \in M$ **do**
        $\mu_i \leftarrow \tanh(a_i)$
    end for
    $t \leftarrow t + 1$
end while
for all $X_i \in X$ **do**
    if $\mu_i \leq 0$ **then**
        $X_i \leftarrow 0$
    else
        $X_i \leftarrow 255$
    end if
end for

T is the number of iterations. We set $T = 20$, and the coupling strength $W_i j$ to be 1 when X_i and X_j are adjacent and 0 otherwise.

1.3 Results

(a) Original (b) Denoised using Gibbs (c) Denoised using VI

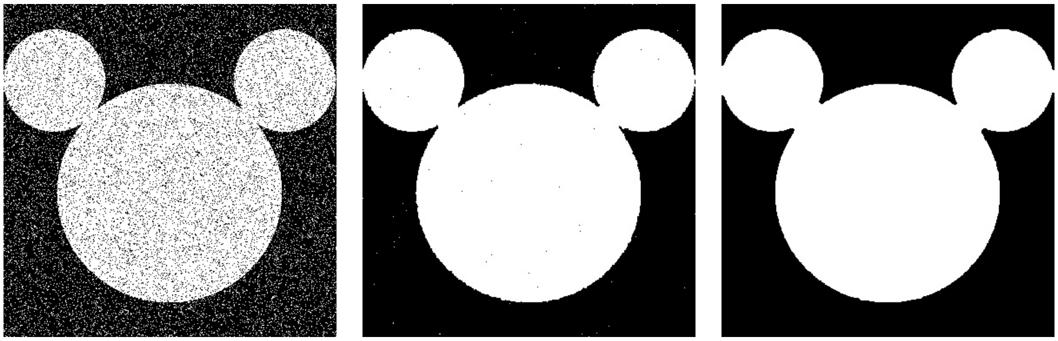
Figure 2: Image denoising of cow from `1_noise.png`



(a) Original

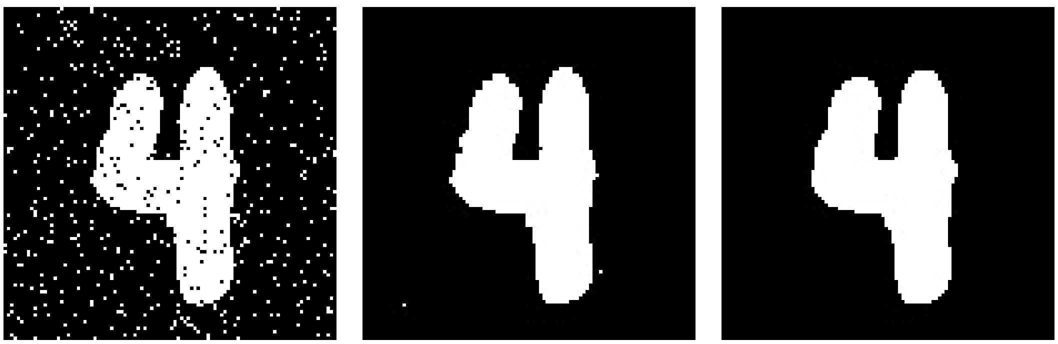
(b) Denoised using Gibbs (c) Denoised using VI

Figure 3: Image denoising of penguins from `2_noise.png`



(a) Original (b) Denoised using Gibbs (c) Denoised using VI

Figure 4: Image denoising of Mickey from `3_noise.png`



(a) Original (b) Denoised using Gibbs (c) Denoised using VI

Figure 5: Image denoising of number 4 from `4_noise.png`

2 Expectation-maximization segmentation

We use the EM algorithm to segment an image into its foreground and background segments. Each segment k is modeled as a Gaussian distribution with $\theta_k = (\mu_k, \Sigma_k)$. Together they form a Gaussian mixture model, with each segment weighted by a mixing weight α_k. The Gaussian mixture model has the form:

$$p(x|\Theta) = \sum_{k=1}^{K} \alpha_k p_k(x|\theta_k) \tag{18}$$

$$\text{where } p_k(x|\theta_k) = \mathcal{N}(x|\mu_k, \theta_k) \tag{19}$$

where Θ is the parameter vector $(\alpha_1, ..., \alpha_K, \theta_1, ..., \theta_K)$. For our case, $K = 2$ since we only want to segment the image into the foreground and background. Given the above Gaussian mixture model, the aim of the EM algorithm is to maximize the likelihood function with respect to Θ.

Our input is an image text file containing the color of each pixel in the CIE-Lab color space, hence the observed data $x_i \in \mathbb{R}^3$.

2.1 Initialization

Our EM algorithm initializes by performing k-means++ for a small number of iterations (10 in our case) with $K = 2$. This allows us to find the approximate cluster centroids as well as assign each image pixel to a cluster (either foreground or background).

μ_k is initialized with the value of cluster k's centroid. Σ_k is initialized with the variance of all the image pixels that are assigned to cluster k. α_k is initialized with the proportion of image pixels that are assigned to cluster k.

2.2 E step

Let Z_i be a latent variable. Its realization z_i is a one-of-K vector. z_{ik} refers to the z_i vector with its k^{th} bit set to 1 and all other bits set to 0. This indicates that X_i belongs to the k^{th} component. Since $K = 2$, z_i is a 2-dimensional vector. We want to evaluate $p(Z|X, \Theta)$. This is given the form:

$$p(z_{ik}|X, \Theta) = \frac{\alpha_k \mathcal{N}(x_i|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \alpha_j \mathcal{N}(x_i|\mu_j, \Sigma_j)} \tag{20}$$

$p(z_{ik}|X, \Theta)$ is also the responsibility value of the k^{th} component for the observation X_i, and is denoted by $\gamma(z_{ik})$.

2.3 M step

In the M step, we want to find Θ^{new}, such that:

$$\Theta^{\text{new}} = \arg\max_{\Theta} \sum_{Z} p(Z|X, \Theta^{\text{old}}) \ln p(X, Z|\Theta) \tag{21}$$

where Θ^{old} is fixed to the current parameter values. The $\arg\max$ with respect to each of the parameters are given by:

$$\mu_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) x_n \tag{22}$$

$$\Sigma_k^{\text{new}} = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(x_n - \mu_k^{\text{new}})(x_n - \mu_k^{\text{new}})^T \tag{23}$$

$$\alpha_k^{\text{new}} = \frac{N_k}{N} \tag{24}$$

where N is the number of image pixels, and $N_k = \sum_{n=1}^{N} \gamma(z_{nk})$.

2.4 Algorithm

We first initialize the parameter values for Θ. Then, we repeatedly perform the E step followed by the M step until the convergence criterion is satisfied or the number of iterations have exceeded the maximum allowed number of iterations. One possible convergence criterion would be checking for convergence of the log likelihood $\ln p(X|\Theta) = \sum_{n=1}^{N} \ln \left(\sum_{k=1}^{K} \alpha_k \mathcal{N}(x_n|\mu_k, \Sigma_k) \right)$. The pseudocode is as follows:

Initialize Θ
while convergence criterion not satisfied **do**
    Do E step
    Do M step
end while

2.5 Results

(a) Original

(b) Mask

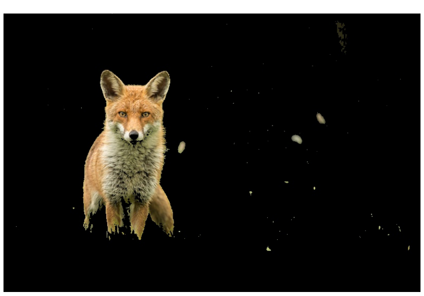
(c) Foreground

(d) Background

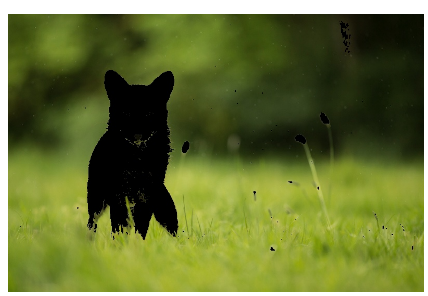
Figure 6: Image segmentation of `cow.jpg`

(a) Original

(b) Mask

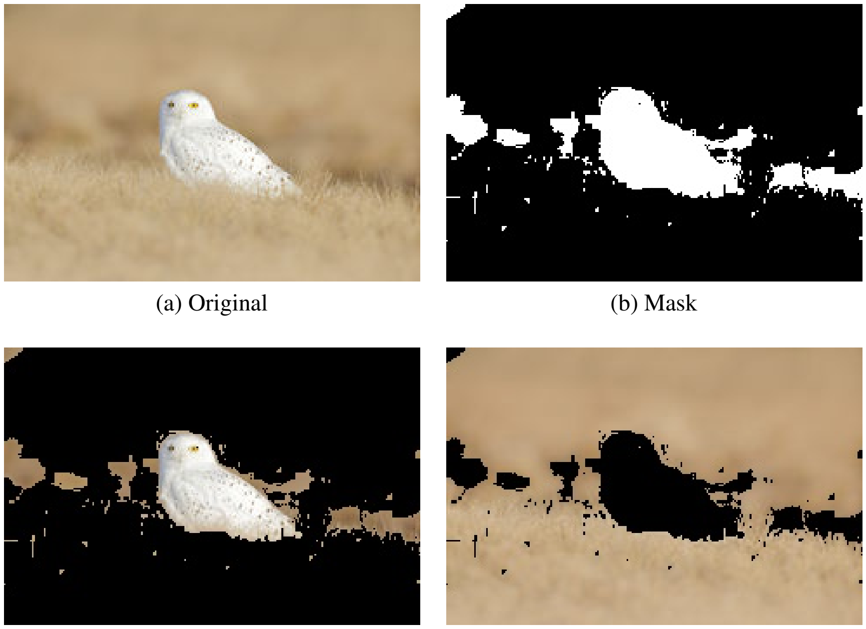


(c) Foreground



(d) Background

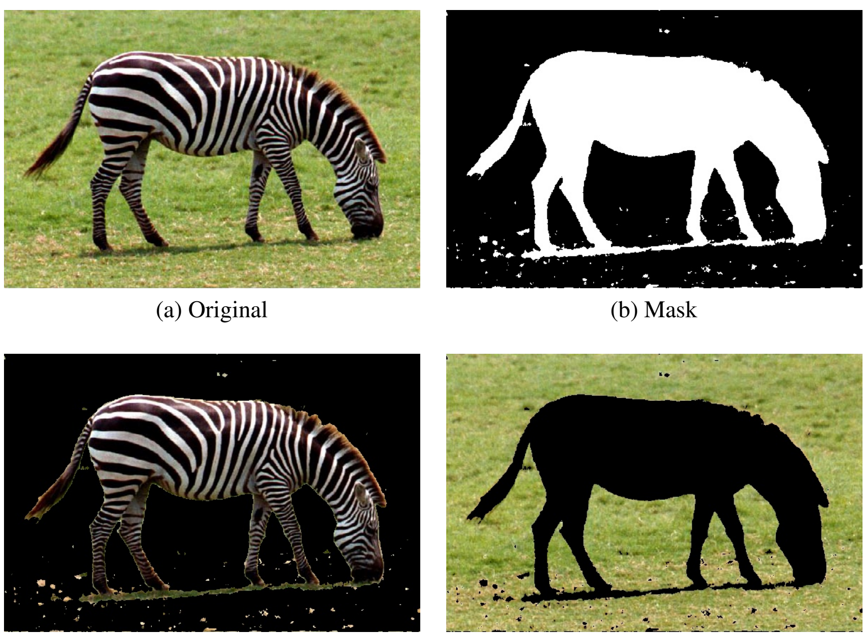
Figure 7: Image segmentation of `fox.jpg`



(a) Original (b) Mask

(c) Foreground (d) Background

Figure 8: Image segmentation of `owl.jpg`



(a) Original (b) Mask

(c) Foreground (d) Background

Figure 9: Image segmentation of `zebra.jpg`

3 Instructions

The source files are written in Python 3 and located in the `src/` directory.

`src/a1.py`
This file contains the source code for image denoising. To run this, navigate to `src/` directory in a terminal and run `python3 a1.py`. Both Gibbs sampling and variational inference algorithms are run on the noisy images, and the denoised images are saved to `output/` directory.

`src/a2.py`
This file contains the source code for image segmentation. To run this, navigate to `src/` directory in a terminal and run `python3 a2.py`. EM algorithm is run on the images, and the masks and segmented images are saved to `output/` directory.

4 Third-party libraries

The following third-party libraries are used:

- NumPy
- OpenCV-Python
- tqdm