Experiment No. 2
Perform Edge detection on the MNIST using Markov Random Field (MRFs)
Date of Performance:
Date of Submission:

Aim: Perform Edge detection on the MNIST using Markov Random Field (MRFs)

Objective: Ablility to implement Markov Random Field (MRFs) for prediction Theory:

A Markov Random Field (MRF) is a graphical model of a joint probability distribution. It consist of an undi$\mathcal{E}$cted graph $G = (\mathcal{N}, \mathcal{E})$ in which the nodes $\mathcal{N}$ represent random variables. Let X_S be the set of random variables associated with the set of nodes S. Then, the edges encode conditional independence relationships via the following rule: given disjoint subsets of nodes A, B, and C, X_A is conditionally independent of X_B given X_C if there is no path from any node in A to any node in B that doesn't pass through a node of C. We write $X_A \perp\!\!\!\perp X_B \mid X_C$. If such a path does exist, the subsets are dependent.
The neighbour set N_n of a node n is defined to be the set of nodes that are connected to n via edges in the graph:

$$N_n = \{m \in \mathcal{N} \mid (n, m) \in \mathcal{E}\}$$

Given its neighbour set, a node n is independent of all other nodes in the graph. Therefore, we may write the following for the conditional probability of X_n:

$$P(X_n|X_{\mathcal{N}} - X_n) = P(X_n|X_{N_n})$$

This is the Markov property, and is where the model gets its name. The following diagram illustrates the concept:

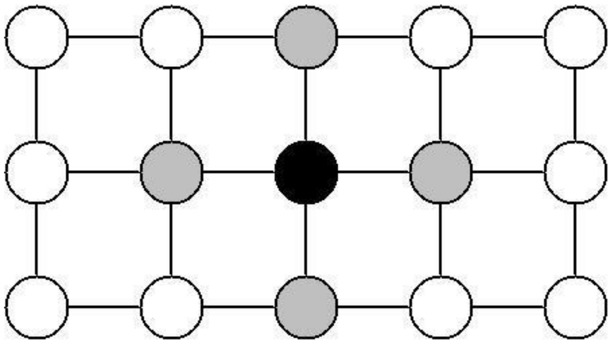


Figure 1: Given the grey nodes, the black node is conditionally independent of all other

nodes. From this point, "node" and "variable" shall be used interchangeably. x shall refer to a particular configuration of the set of the random variables. A subscript will denote a particular node or subset of nodes.
The Markov property tells us that the joint distribution of X is determined entirely by the local conditional distributions $P(X_n|X_{N_n})$. But it is not clear how to actually construct the global joint distribution from these local functions. In order to do this, we need to look at Gibbs distributions.

A Gibbs distribution on the graph G takes the form:

$$P(x) = \frac{1}{Z} \prod_{c \in C} \phi_c(x_c)$$

where the product is over all maximal cliques in the graph. A clique is a subset of nodes in which every node is connected to every other node. A maximal clique is a clique which cannot be extended by the addition of another node. Z is called the partition function, and takes the form:

$$Z = \sum_{x} \prod_{c \in C} \phi_c(x_c)$$

The $\phi_c(x_c)$ are usually written:

$$\phi_c(x_c) = e^{-\frac{1}{T} V_c(x_c)}$$

T is called the temperature, and is often taken to be 1. So $P(x)$ has the alternate form:

$$P(x) = \frac{1}{Z} e^{-\frac{1}{T} U(x)}$$

where

$$U(x) = \sum_{c \in C} V_c(x)$$

is called the energy. Either the $\phi_c(x_c)$ or the $V_c(x_c)$ may be referred to as clique potentials.
The Hammersley-Clifford theorem states that the joint probability distribution of any MRF can be written as a Gibbs distribution, and furthermore that for any Gibbs distribution there exists an MRF for which it is the joint. That is to say, Hammersley-Clifford establishes the equality of the MRF and Gibbs models. This solves the problem of how to specify the joint distribution of an MRF in terms of local functions: it can now be specified by defining the potential on every maximal clique.

Optimisation

An optimisation problem is one that involves finding the extremum of a quantity or function. Such problems often arise as a result of a source of uncertainty that precludes the possibility of an exact solution.

Optimisation in an MRF problem involves finding the maximum of the joint probability over the graph, usually with some of the variables given by some observed data. Equivalently, as can be seen from the equations above, this can be done by minimising the total energy, which in turn requires the simultaneous minimisation of all the clique potentials.

Techniques for minimisation of the MRF potentials are plentiful. Many of them are also applicable to optimisation problems other than MRF. For example, gradient descent methods are well-known techniques for finding local minima, while the closely-related method of simulated annealing attempts to find a global minimum.

An example of a technique invented specifically for MRF optimisation is Iterated Conditional Modes (ICM). This simple algorithm proceeds first by choosing an initial configuration for the variables. Then, it iterates over each node in the graph and calculates the value that minimises the energy given the current values for all the variables in its neighbourhood. At the end of an iteration, the new values for each variable become the current values, and the next iteration begins. The algorithm is guaranteed to converge, and may be terminated according to a chosen criterion of convergence. An example of this technique in action can be seen below.

MRF Applications To Vision

Problems in computer vision usually involve noise, and so exact solutions are most often impossible. Additionally, the latent variables of interest often have the Markov property. For example, the pixel values in an image usually depend most strongly on those in the immediate vicinity, and have only weak correlations with those further away. Therefore, vision problems are well suited to the MRF optimisation technique. Some examples of vision problems to which MRFs have been applied are:

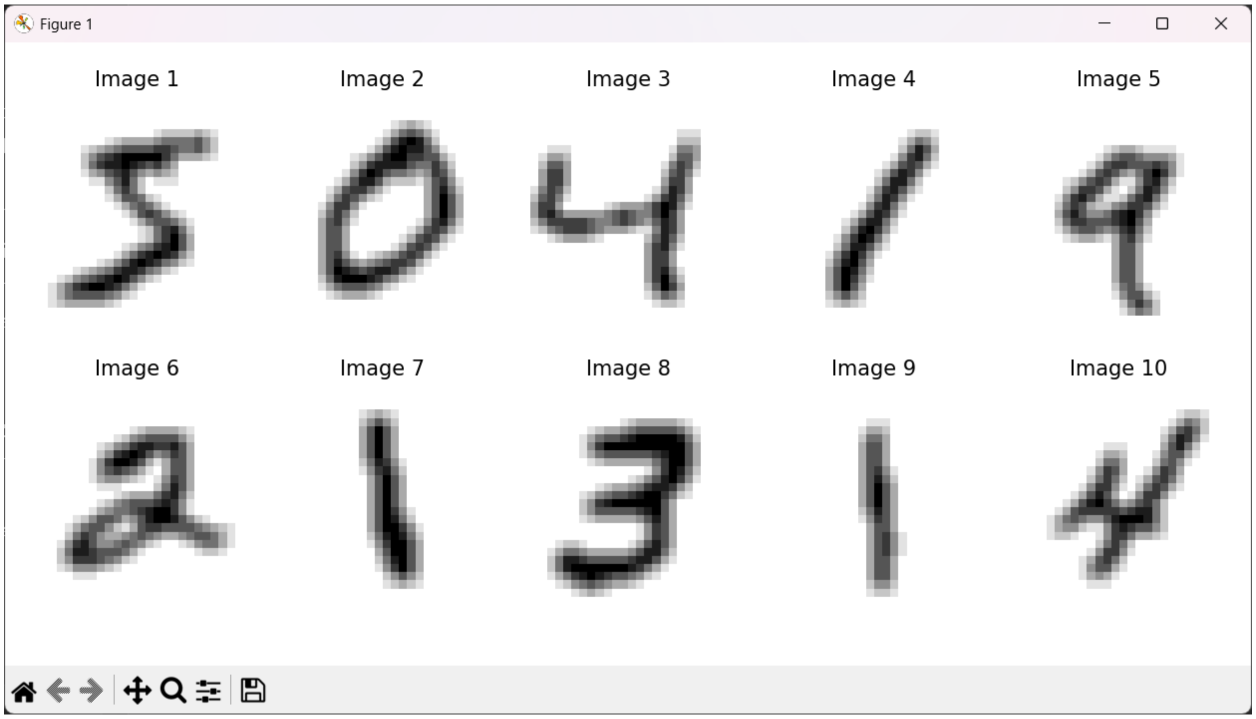
1. Image restoration
2. Image reconstruction
3. Segmentation
4. Edge detection

In the first 3 of these, there is a latent random variable for each pixel. The variables range over intensity values in 1 and 2, while in 3 the variables take on segment identifiers. In problem 4, the variables correspond to pairs of neighbouring pixels, and their values are binary, indicating the presence or absence of an edge. In all cases, the neighbourhood concept of the MRF maps to the geometric neighbourhood of the image. That is to say, the edges in the MRF graph will connect geometrically close pixels or edges. The appropriate size of the neighbourhood depends on the problem at hand: a larger neighbourhood has more modelling power at the expense of greater computational demand during optimisation.

Code:

```
import numpy as np
import cv2
import matplotlib.pyplot as plt
from sklearn.preprocessing import binarize
from sklearn.datasets import fetch_openml

def load_mnist():
    mnist = fetch_openml('mnist_784', version=1, parser='auto')
    X = mnist.data.values.reshape(-1, 28, 28)
    return X[:10]  # Load first 10 images for demonstration

def apply_mrf_edge_detection(image, beta=1.0, iterations=5):
    height, width = image.shape
    binary_image = binarize(image, threshold=127).astype(np.uint8)

    edge_map = np.zeros_like(binary_image, dtype=np.float32)

    for _ in range(iterations):
        for i in range(1, height - 1):
            for j in range(1, width - 1):
                neighbors = binary_image[i-1:i+2, j-1:j+2]
                energy = -beta * np.sum(neighbors)
                edge_map[i, j] = energy

    edge_map = (edge_map - edge_map.min()) / (edge_map.max() -
edge_map.min()) * 255
    return edge_map.astype(np.uint8)

# Load MNIST images
mnist_images = load_mnist()

# Perform edge detection using MRF on sample images
fig, axes = plt.subplots(2, 5, figsize=(10, 5))
for i in range(10):
    image = mnist_images[i]
    edge_detected = apply_mrf_edge_detection(image)
    ax = axes[i // 5, i % 5]
    ax.imshow(edge_detected, cmap='gray')
    ax.axis('off')
    ax.set_title(f'Image {i+1}')

plt.tight_layout()
plt.show()
```

Output:



Conclusion:

The Markov Random Field (MRF)-based edge detection successfully highlights the edges in MNIST images by leveraging the neighborhood relationships of pixels. The model assigns probabilities to pixel pairs, reinforcing edge structures while suppressing noise. The results show that MRF is an effective approach for edge detection, though computationally more expensive than traditional methods like Sobel or Canny filters. The choice of parameters like β and iteration count significantly impacts performance, allowing fine-tuning for better edge clarity.