DL Programming Assignment 4

Sanchit Agrawal - CS13B061
Ishu Dharmendra Garg - CS13B060

April 2017

1 Introduction

RBMs are a special class of Undirected Graphical Models (also known as Markov Random Fields) that are used to model the probability distribution of a given dataset. The graphical structure of RBMs is restricted to be a complete bipartite graph, where the partition sets are the nodes corresponding to the visible & hidden random variables.
The parameters of an RBM are w_{ij}'s (the weight connecting hidden node h_i to the visible node v_j), b_j's (the bias of the visible node v_j), and c_i's (the bias of the hidden node h_i).
RBMs are trained using gradient ascent to maximize the log likelihood of the empirical data distribution. However, training RBMs using exact gradients is intractable, since the gradient computations require summing over an exponential number of state configurations. Thus, the gradient computation is approximated using Markov Chain Monte Carlo (MCMC) methods.
One of the popular MCMC methods used to train RBMs is the k-step Contrastive Divergence algorithm, which we have implemented for this assignment.

2 Task

For this assignment, we train an RBM with 500 hidden states on the train split of the MNIST dataset. The trained RBM is then used to obtain abstract representations of the test split. The abstract representations are simply the expected values of the hidden nodes. The 500 dimensional representations are projected to 2 dimensions using t-SNE. These embeddings are then visualized in order to see if different classes form different clusters in the lower dimensional space.

3 Training

The main hyperparameters for the task are:

- k : The number of sampling steps in Contrastive Divergence

- η : The learning rate
- n : Number of training epochs

We use stochastic & mini-batch gradient ascent as our learning algorithm.

4 Implementation

We implemented Contrastive Divergence using vectorized matrix operations that allow the execution to be parallelized across several cores (with the help of Numpy).

5 Plots

We show the plots of 2-dimensional t-SNE embeddings for the test set for various hyperparameter settings.

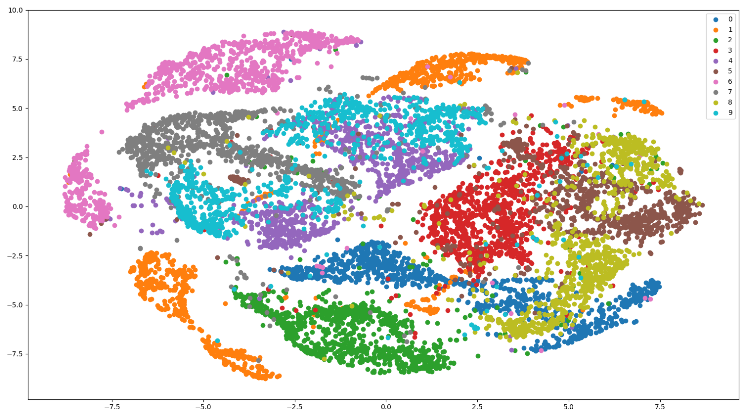

Figure 1: t-SNE embeddings for n = 10, $\eta = 10^{-3}$, k = 10, mini-batch size = 1

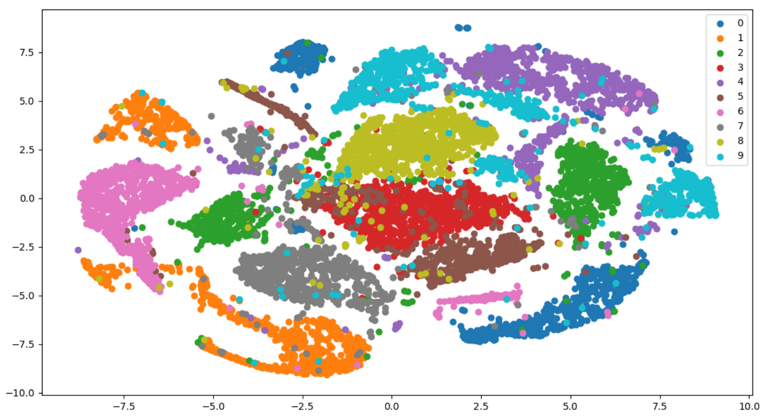

Figure 2: t-SNE embeddings for n = 10, $\eta = 10^{-2}$, k = 10, mini-batch size = 16

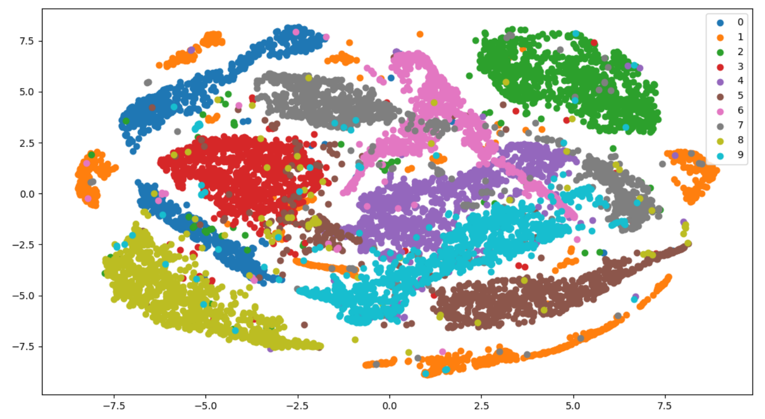

Figure 3: t-SNE embeddings for n = 5, $\eta = 10^{-2}$, k = 10, mini-batch size = 16

6 Results

We observe that examples belonging to a particular class are largely clustered together, with partial overlaps with other clusters.
Thus the t-SNE projections of the hidden representations learned by the RBM are well separated.