Pranav Mishra pmishr23@uic.edu
Aneesh Potnis apotni2@uic.edu
Aditya Pimpley apimp@uic.edu

**CS 512: Advanced Machine Learning**

# Lab 1: Graphical Models

*Student:* *Email:*

**Deadline: 23:59 PM, Mar 12, 2024**

**This lab is for group work. Unless otherwise specified you cannot use any library beyond the standard ones provided with Python, Numpy, and Scipy**. That is, the use of machine learning libraries such as sklearn is prohibited. We provided some utility code.

**How to submit.** Only one member of each team needs to submit a zip file on Blackboard under Assessment/Lab 1. The filename should be `Firstname_Lastname.zip`, where both names correspond to the member who submits it.

Inside the zip file, the following contents should be included:

1. A PDF report named `Report_Firstname_Lastname.pdf` with answers to the questions detailed below. **Your report should include the name and NetID of *all* team members.** The LaTeX source code of this document is provided with the package, and you may write up your report based on it.
2. A folder named `result` containing **<u>four</u> output result files** (underlined below in this document).
3. A folder named `code` that contains your source code, along with a short `readme.txt` file (placed inside `code/`) that explains how to run it. Your code should be well commented.

You are allowed to resubmit as often as you like and your grade will be based on the last version submitted. Late submissions will not be accepted in any case, unless there is a documented personal emergency. Arrangements must be made with the instructor as soon as possible after the emergency arises, preferably well before the deadline.

## Q1 Solution

(1a) We know that

$$\log p(\mathbf{y}|X) = \log \frac{1}{Z_X} \exp\left(\sum_{s=1}^{m} \langle \mathbf{w}_{y_s}, \mathbf{x}_s \rangle + \sum_{s=1}^{m-1} T_{y_s, y_{s+1}}\right) \tag{1}$$

This simplifies to:

$$\log p(\mathbf{y}|X) = \sum_{s=1}^{m} \langle \mathbf{w}_{y_s}, \mathbf{x}_s \rangle + \sum_{s=1}^{m-1} T_{y_s, y_{s+1}} - \log Z_x \tag{2}$$

We shall now take the derivative of the first part of equation 2 with respect to $\boldsymbol{w}_y$.

$$\nabla_{\mathbf{w}_y} \sum_{s=1}^{m} \langle \mathbf{w}_{y_s}, \mathbf{x}_s \rangle + \sum_{s=1}^{m-1} T_{y_s, y_{s+1}} = \sum_{s=1}^{m} [\![ y_s = y ]\!] x_s^t \tag{3}$$

Notice that the summation of transitions vanishes as it was independent of $\mathbf{w}_y$ and $x_s^t$ only appears when $y_s = y$.

We now calculate the derivative of $\log Z_X$:

$$\nabla_{\mathbf{w}_c} \log Z_X = \frac{1}{Z_X} \nabla_{\mathbf{w}_c} Z_X \quad = \frac{1}{Z_X} * \sum_{\hat{\mathbf{y}} \in \mathcal{Y}^m} \exp\left(\sum_{s=1}^{m} \langle \mathbf{w}_{\hat{y}_s}, \mathbf{x}_s \rangle + \sum_{s=1}^{m-1} T_{\hat{y}_s, \hat{y}_{s+1}}\right) * \sum_{s=1}^{m} [\![ y_s = y ]\!] x_s^t \tag{4}$$

We can simplify the above equation by substituting the first part as p($\boldsymbol{y}|X$) :

$$\sum_{\hat{\mathbf{y}} \in \mathcal{Y}^m} p(y|X) \sum_{s=1}^{m} [\![ y_s = y ]\!] x_s^t = \sum_{s=1}^{m} \sum_{\hat{\mathbf{y}} \in \mathcal{Y}^m} p(y|X) [\![ y_s = y ]\!] x_s^t \tag{5}$$

The second summation is a marginalization over y except the label we are derivating against. As a result we can write the equation as follows:

$$\nabla_{\mathbf{w}_c} \log Z_X = \sum_{s=1}^{m} p(y_s = y|X^t)) x_s^t \tag{6}$$

Now we combine equation 3 and 6 to get the desired result:

$$\nabla_{\mathbf{w}_y} \log p(\mathbf{y}^t|X^t) = \sum_{s=1}^{m} ([\![ y_s^t = y ]\!] - p(y_s = y|X^t)) \mathbf{x}_s^t \tag{7}$$

We shall now differentiate with respect to T
We can start out by writing:

$$\nabla_{Tij} \sum_{s=1}^{m} \langle \mathbf{w}_{y_s}, \mathbf{x}_s \rangle + \sum_{s=1}^{m-1} T_{y_s,y_{s+1}} = \sum_{s=1}^{m-1} [\![ y_s = i, y_{s+1} = j ]\!] \tag{8}$$

The first part vanishes since it is independent of T. Only the terms having i,j will remain.

We now calculate the derivative of $\log Z_X$ similar to how we did in equation 4:

$$\nabla_{T_{ij}} \log Z_X = \frac{1}{Z_X} * \sum_{\hat{\mathbf{y}} \in \mathcal{Y}^m} \exp\left( \sum_{s=1}^{m} \langle \mathbf{w}_{\hat{y}_s}, \mathbf{x}_s \rangle + \sum_{s=1}^{m-1} T_{\hat{y}_s,\hat{y}_{s+1}} \right) * \sum_{s=1}^{m-1} [\![ y_s = i, y_{s+1} = j ]\!] \tag{9}$$

We will substitute $p(\mathbf{y}|X)$ as we did in equation 5:

$$\sum_{\hat{\mathbf{y}} \in \mathcal{Y}^m} p(y|X) \sum_{s=1}^{m-1} [\![ y_s = i, y_{s+1} = j ]\!] = \sum_{s=1}^{m-1} \sum_{\hat{\mathbf{y}} \in \mathcal{Y}^m} p(y|X) [\![ y_s = i, y_{s+1} = j ]\!] \tag{10}$$

After marginalization we end up with

$$\nabla_{T_{ij}} \log Z_X = \sum_{s=1}^{m-1} p(y_s = i, y_{s+1} = j | X^t) \tag{11}$$

Combining equations 8 and 11 the gradient is:

$$\nabla_{T_{ij}} \log Z_X = \sum_{s=1}^{m-1} [\![ y_s = i, y_{s+1} = j ]\!] - p(y_s = i, y_{s+1} = j | X^t) \tag{12}$$

(1b) The expectation of some features with respect to $p(\mathbf{y}|X)$ will be :

$$\sum_{s=1}^{m} p(\mathbf{y}|X^t) x_s [\![ y_s = y ]\!] \tag{13}$$

The function will only be non-zero for terms s.t. $y_s = y$ and the others will go to zero. Therefore, we can say that the above is equivalent to equation 4 because the $p(\mathbf{y}|X)$ term will become $p(y_s = y|X)$ via marginalization, and $x_s$ will be selected by the feature function. The gradient with respect to $T$ has a similar argument. But instead what will be important is the feature $[\![ y_s = i, y_{s+1} = j ]\!]$ which will marginalize $p(\mathbf{y}|X)$ to $p(y_s = i, y_{s+1} = j)$.

(1c) The maximum objective value for 1c is 199.41772558210562

## Question 2 Solution

(2a) The value of $\frac{1}{n}\sum_{t=1}^{n} \log p(\mathbf{y}^t|X^t)$ for this case : -19.744078889052513

(2b) Current function value: 24594.932941
Iterations: 0
Function evaluations: 51
Gradient evaluations: 40
Time: 5028.738506555557

## Question 3 Solution

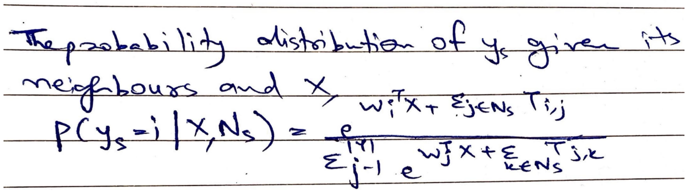
(3a) We see that as we increase C in all three models, the accuracy on a letter-by-letter basis improves. The SVM HMM model demonstrated the highest performance, closely trailed by the CRF model. On the other hand, the SVM MC scored the lowest.

(3b) A similar pattern emerges in word-wise prediction accuracy as seen in letter-wise accuracy. Once more, the models show improvement with higher values of C. However, accuracies decrease significantly due to the increased difficulty in accurately classifying entire words compared to individual letters. The SVM HMM model ranks highest, followed by the CRF and SVM MC models.

## Question 4 Solution

(4a) Refer to figures at the end of this document

(4b)(i) Considering $d$ neighbours for $y_s$, the conditional probability $p(y_s|\ldots, y_{s-3}, y_{s-1}, y_{s+1}, y_{s+3}, \ldots, X)$ will depend on $d$ neighbours on each side of $y_s$.
Let's denote the neighbouring states as $N_s = \{y_{s-d}, \ldots, y_{s-1}, y_{s+1}, \ldots, y_{s+3}\}$



The probability distribution of $y_s$ given its neighbours and X,

$$P(y_s=i\,|\,X, N_s) = \frac{e^{w_i^T X + \Sigma_{j\in N_s} T_{i,j}}}{\Sigma_{j-1}^{|Y|} e^{w_j^T X + \Sigma_{k\in N_s} T_{j,k}}}$$

Considering, one node $y_s$, considering $d$ neighbours and an alphabet of size $|Y| = 26$ the complexity is $O(m.|Y| + d.|Y|)$ as we have to compute $m$-dimensional dot products and then add $d$ terms for each of $|Y|$ states.

For one iteration of block Gibbs sampling(steps 4 and 5) which is a Markov Chain Monte Carlo sampler, if the length of the sequence is n, the complexity is $O((m + d).|Y|.n)$ as we are ignoring odd and even-indexed nodes.

Similarly, the complexity for dynamic programming such as forward-backward algorithm for sequence models like HMM is $O(n.m.|Y|^2)$. This includes the consideration of transition probabilities between all pairs of states, hence the $|Y|^2$ term.

(4c)(i) PSEUDO CODE for (i):

```
Initialization:

  – Set all counts for node and edge marginals to 0.
  – Choose an initial state for the sequence $y$ (e.g., randomly or based on some heuristic).

For each sample iteration $i$ from 1 to $N$:

  1. For each node $k$ in the sequence:
     (a) Calculate the conditional probability distribution $p = (p_a, \dots, p_z)$ of $y_k$ given all other nodes.
     (b) Update node marginal counts: For each possible state $a$ of $y_k$, increment the count of $y_k = a$ by $p_a$.
     (c) If $k > 1$, update edge marginal counts for $(y_{k-1}, y_k)$:
         – Given the current state $r$ of $y_{k-1}$ and for each possible state $a$ of $y_k$, increment the count for $P_{k-1,k}(r, a)$ by $p_a$.
     (d) If $k <$ length$(y)$, update edge marginal counts for $(y_k, y_{k+1})$:
         – Given the next state $s$ of $y_{k+1}$ and for each possible state $a$ of $y_k$, increment the count for $P_{k,k+1}(a, s)$ by $p_a$.
  2. Optionally, resample $y$ based on the updated probabilities to generate a new sequence for the next iteration.

Normalization:

  – After all iterations, normalize the counts for node and edge marginals by dividing by the total number of samples ($N$) to obtain estimated probabilities.
```

(4c)(ii) Observations from the graph:
Rao-Blackwellization Impact: Compared to node and edge marginals without RB (shown by squares and triangles), the KL divergence with RB (shown by circles and crosses) is noticeably smaller. This shows that by successfully lowering the variance of the MCMC estimates, RB produces an approximation that is closer to the actual distribution.

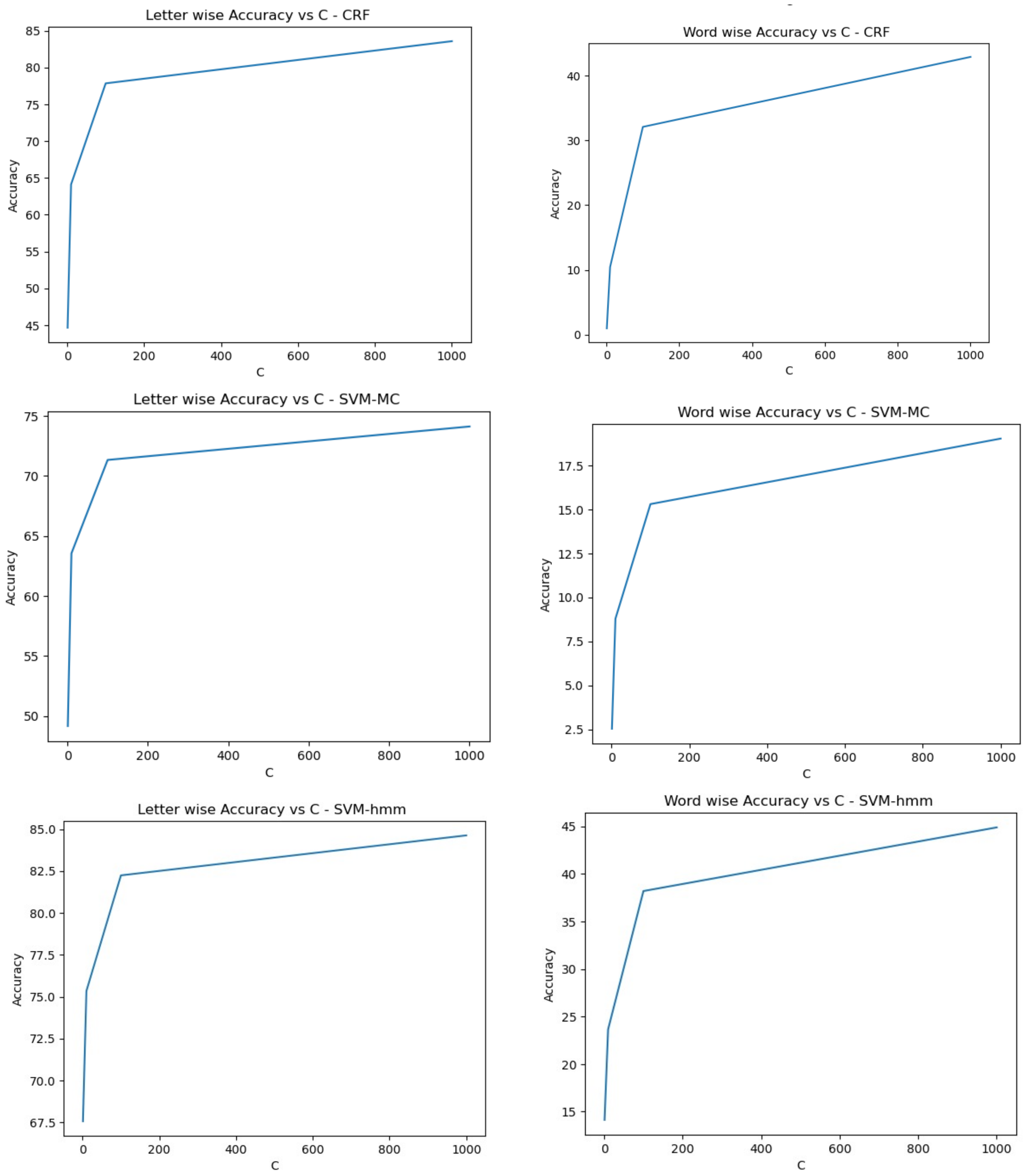
Convergence Behavior: As the number of samples rises, the curves for the node and edge marginals with RB seem to converge or stabilize, suggesting that the approach is producing increasingly precise estimates of the genuine distribution. However, there is no apparent pattern of convergence in the divergence for the node and edge marginals without RB, particularly for the edge marginals where the KL divergence is still rather large even after 100 samples.

The observation that RB improves MCMC sampling for both node and edge marginals suggests that fractional counting could be a generally applicable technique to improve MCMC estimation in graphical models.

## Question 5 Solution

(5a) The accuracy declines in both the models with an increase in transformations. This is expected since we are manipulating the images. The SVM MC model seems to be impacted more than the CRF model due to it not being structured.

(5b) Same phenomenon as 5a is observed even when taking word accuracy into account



Letter wise Accuracy vs C - CRF

Word wise Accuracy vs C - CRF

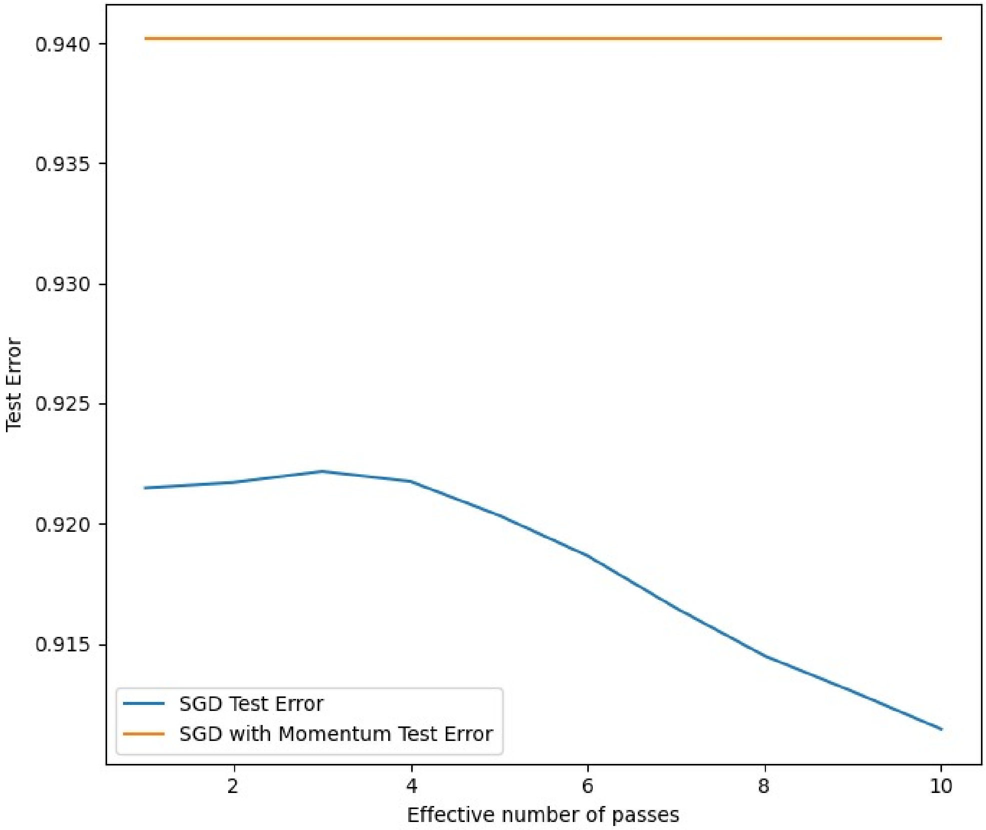
Letter wise Accuracy vs C - SVM-MC

Word wise Accuracy vs C - SVM-MC

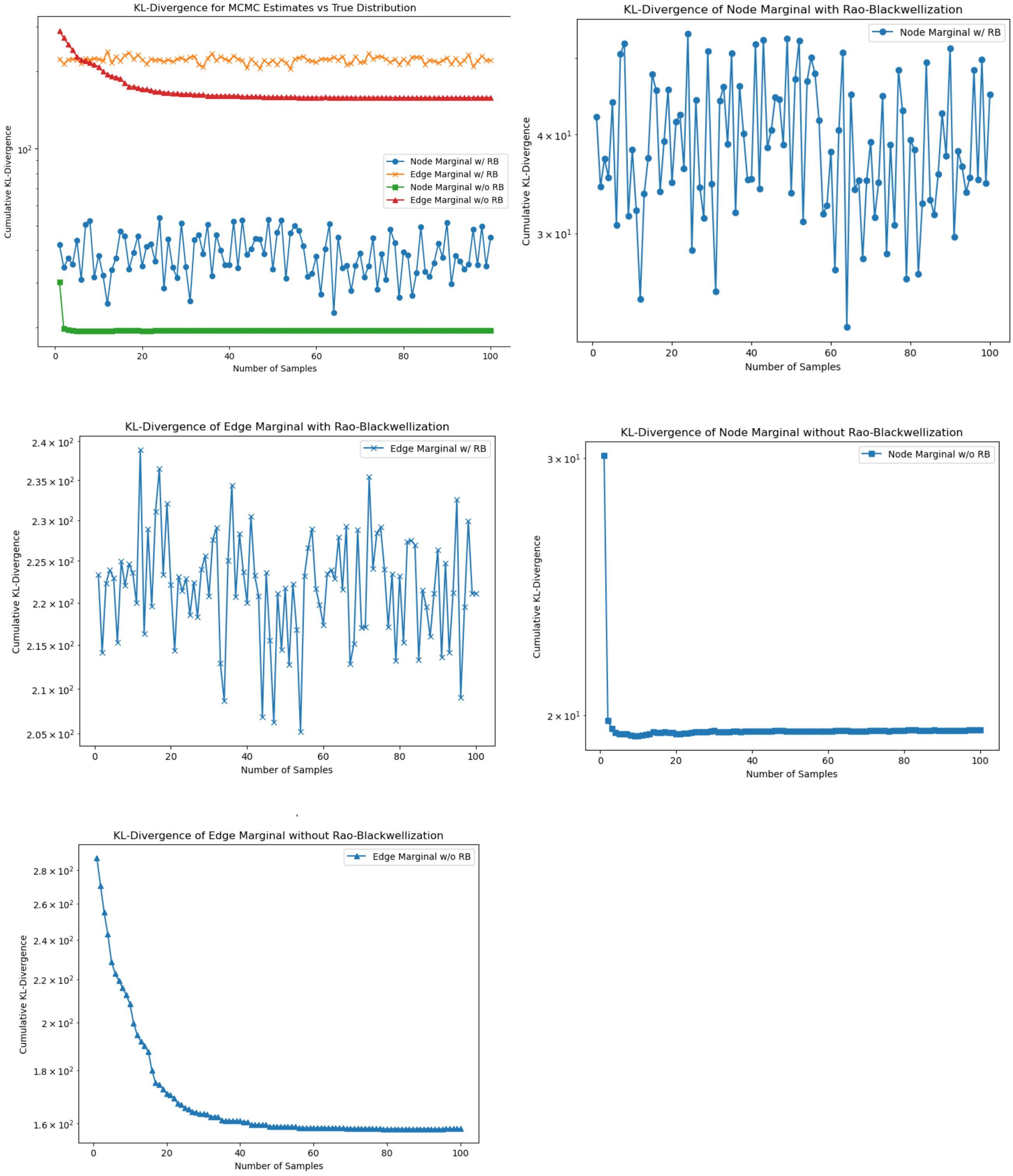
Letter wise Accuracy vs C - SVM-hmm

Word wise Accuracy vs C - SVM-hmm

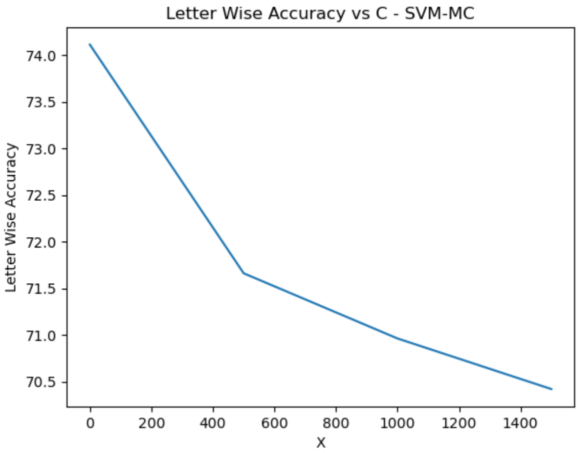
Q3 graphs

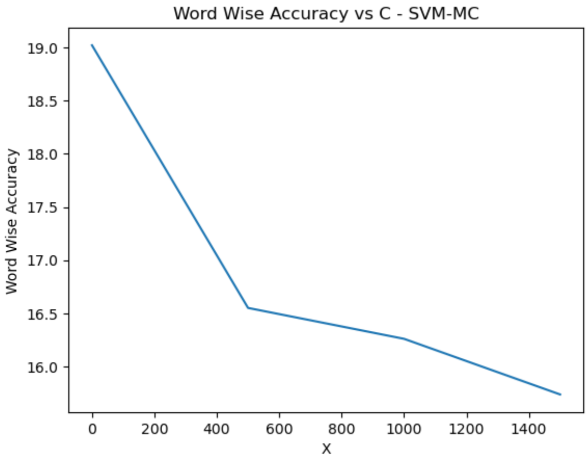

0.940
0.935
0.930
0.925
0.920
0.915
Test Error
2
4
6
8
10
Effective number of passes
SGD Test Error
SGD with Momentum Test Error

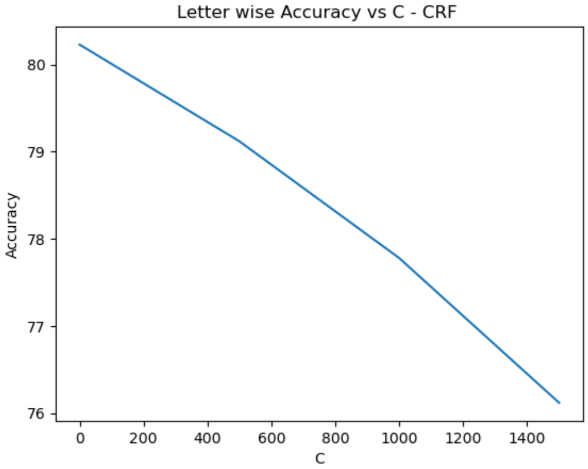
Q4 (a) Graph

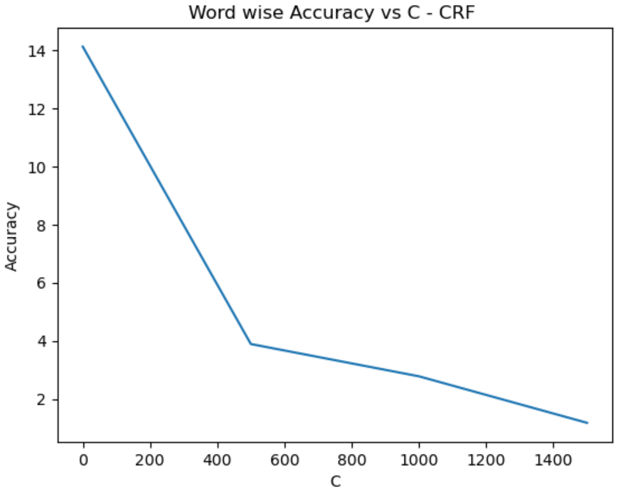


Q4 c(ii) Graphs


Letter Wise Accuracy vs C - SVM-MC
Letter Wise Accuracy
74.0
73.5
73.0
72.5
72.0
71.5
71.0
70.5
0
200
400
600
800
1000
1200
1400
X


Word Wise Accuracy vs C - SVM-MC
Word Wise Accuracy
19.0
18.5
18.0
17.5
17.0
16.5
16.0
0
200
400
600
800
1000
1200
1400
X


Letter wise Accuracy vs C - CRF
Accuracy
80
79
78
77
76
0
200
400
600
800
1000
1200
1400
C


Word wise Accuracy vs C - CRF
Accuracy
14
12
10
8
6
4
2
0
200
400
600
800
1000
1200
1400
C

Q5 (a,b) Graphs