CS688: Graphical Models - Spring 2017

Assignment 2

Assigned: Tuesday, Feb 28th. Due: Tuesday, Mar 14th, 1:00pm

General Instructions: Submit a report with the answers to each question at the start of class on the date the assignment is due. You are encouraged to typeset your solutions. To help you get started, the full LaTeXsource of the assignment is included with the assignment materials. For your assignment to be considered "on time", you must upload a zip file containing all of your code to Moodle by the due time. Make sure the code is sufficiently well documented that it's easy to tell what it's doing. You may use any programming language you like. For this assignment, you may not use existing code libraries for inference and learning with CRFs or MRFs. If you think you've found a bug with the data or an error in any of the assignment materials, please post a question to the discussion forum. Make sure to list in your report any outside references you consulted (books, articles, web pages, etc.) and any students you collaborated with.

Academic Honesty Statement: Copying solutions from external sources (books, web pages, etc.) or other students is considered cheating. Sharing your solutions with other students is also considered cheating. Any detected cheating will result in a grade of 0 on the assignment for all students involved, and potentially a grade of F in the course. See course policies on the course website.

Introduction: In this assignment, you will experiment with different aspects of modeling, learning, and inference with chain-structured conditional random fields (CRFs). This assignment focuses on the task of optical character recognition (OCR). We will explore an approach that bridges computer vision and natural language processing by jointly modeling the labels of sequences of noisy character images that form complete words. This is a natural problem for chain-structured CRFs. The node potentials can capture bottom-up information about the character represented by each image, while the edge potentials can capture information about the co-occurrence of characters in adjacent positions within a word.

Code: You may write code in any language you want. We suggest one of the dynamic numerical languages, such as Python, Matlab, Julia, or R. We have provided "suggested pseudocode" function signatures in "Latex-Source/functions.txt" as rough suggestions for how you could structure your code.

Data: The underlying data are a set of N sequences corresponding to images of the characters in individual words. Each word i consists of L_i positions. For each position j in word i, we have a noisy binary image of the character in the that position. In this assignment, we will use the raw (binary) pixel values of the character images as features in the CRF. The character images are 20×16 pixels. We convert them into 1×320 vectors. We include a constant bias feature along with the pixels in each image, giving a final feature vector of length $F = 321$. x_{ijf} indicates the value of feature f in position j of word i. The provided training and test files *train_img<i>.txt* and *test_img<i>.txt* list the character image $\mathbf{x}_{ij}$ on row j of file i as a 321-long space-separated sequence.[1] The data files are in the column-major format. Given

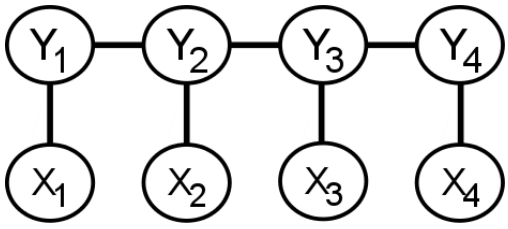
[1] Images are also provided for each training and test word as standard PNG-format files *train_img<i>.png* and *test_img<i>.png*. These are for your reference and not for use in training or testing algorithms.

the sequence of character images $\mathbf{x}_i = [\mathbf{x}_{i1}, ..., \mathbf{x}_{iL_i}]$ corresponding to test word i, our goal is to infer the corresponding sequence of character labels $\mathbf{y}_i = [y_{i1}, ..., y_{iL_i}]$. To reduce the computational complexity of exhaustive inference, we will use a limited set of characters corresponding to the 10 most frequently used characters in the English language: "**etainoshrd**". There are thus $C = 10$ possible labels for each word position. The character labels for each training and test word are available in the files *train_words.txt* and *test_words.txt*. The figure below shows several example words along with their images.

shoot indoor threee trait

Model: The conditional random field model is a conditional model $P_W(\mathbf{y}_i|\mathbf{x}_i)$ of the sequence of class labels $\mathbf{y}_i$ given the sequence of feature vectors $\mathbf{x}_i$ that depends on a collection of parameters W. The CRF graphical model is shown below for a sequence of length 4.

Conditional Random Field



The probabilistic model for the CRF we use in this assignment is given below. The CRF model contains one feature parameter W^F_{cf} for each of the $C = 10$ character labels and $F = 321$ features. The feature parameters encode the compatibility between feature values and character labels. The CRF also contains one transition parameter $W^T_{cc'}$ for each pair of character labels c and c'. The transition parameters encode the compatibility between adjacent character labels in the word. We parameterize the model in log-space, so all of the parameters can take arbitrary (positive or negative) real values. We have one feature potential $\psi^F_j(y_{ij}|\mathbf{x}_{ij}, W^F)$ for each position j in word i and one transition potential for each pair of adjacent labels $\psi^T_j(y_{ij}, y_{ij+1}|W^T)$ in word i.

$$\psi^F_j(y_{ij}|\mathbf{x}_{ij}, W^F) = \exp(\sum_{c=1}^{C}\sum_{f=1}^{F} W^F_{cf}[y_{ij} = c]x_{ijf})$$

$$\psi^T_j(y_{ij}, y_{ij+1}|W^T) = \exp(\sum_{c=1}^{C}\sum_{c'=1}^{C} W^T_{cc'}[y_{ij} = c][y_{ij+1} = c'])$$

Here we use $[\dots]$ as the notation for indicator functions, i.e. $[conditions]$ equals 1 when $conditions$ are all met, and 0 otherwise. For example, $[y_{ij} = c]$ equals 1 when $y_{ij} = c$, and 0 otherwise.

Note that these potentials are always positive. Please inspect the data files and make sure you understand the shapes of each variable and the resulting shapes of the potentials. As always with Markov networks, the joint distribution is given by the product of potentials and must be explicitly normalized, as shown below. In a CRF model, the feature vectors are always conditioned on, so the joint model shown below **must** be

transformed into a conditional model $P_W(\mathbf{y}_i|\mathbf{x}_i)$ using factor reduction before performing inference for the character labels.

$$P_W(\mathbf{y}_i, \mathbf{x}_i) = \frac{\prod_{j=1}^{L_i} \psi_j^F(y_{ij}|\mathbf{x}_{ij}, W^F) \cdot \prod_{j=1}^{L_i-1} \psi_j^T(y_{ij}, y_{ij+1}|W^T)}{\sum_{\mathbf{y}'_i}\sum_{\mathbf{x}'_i} \prod_{j=1}^{L_i} \psi_j^F(y'_{ij}|\mathbf{x}'_{ij}, W^F) \cdot \prod_{j=1}^{L_i-1} \psi_j^T(y'_{ij}, y'_{ij+1}|W^T)}$$

$$= \frac{\exp\left(\sum_{j=1}^{L_i}\sum_{c=1}^{C}\sum_{f=1}^{F} W_{cf}^F[y_{ij}=c]x_{ijf} + \sum_{j=1}^{L_i-1}\sum_{c=1}^{C}\sum_{c'=1}^{C} W_{cc'}^T[y_{ij}=c][y_{ij+1}=c']\right)}{\sum_{\mathbf{y}'_i}\sum_{\mathbf{x}'_i} \exp\left(\sum_{j=1}^{L_i}\sum_{c=1}^{C}\sum_{f=1}^{F} W_{cf}^F[y'_{ij}=c]x'_{ijf} + \sum_{j=1}^{L_i-1}\sum_{c=1}^{C}\sum_{c'=1}^{C} W_{cc'}^T[y'_{ij}=c][y'_{ij+1}=c']\right)}$$

Question 1. (*20 points*) **Exhaustive Inference:** In this question, you will implement simple exhaustive inference for the CRF model. The code packages provides a pre-trained model for the OCR task including the feature parameters (*feature-params.txt*) and the label-label transition parameters (*transition-params.txt*). Use these parameters to answer the following questions. For grading purposes, make sure to list results table rows and/or columns using the character ordering "etainoshrd".

1.1 (*2 points*): *For the first test word only, from $\psi_j^F(y_{ij}|\mathbf{x}_{ij}, W^F)$ compute the node potentials $\psi'(y_{ij})$ obtained by conditioning the CRF on the observed image sequence. After conditioning, there is one node potential per position in the test word. Each node potential is a vector with one entry per character label. Report the node potential* ***in log space*** *as a table for each position in the test word.*

1.2 (*2 points*): *Energy is a function which measures the "goodness" (or badness) of each possible configuration of the random variables (the lower the engergy, the better the configuration). We can define energy as the negative logarithm of (possibly unnormalized) probability. For the first three test words, compute the value of the negative energy of the true label sequence after conditioning on the corresponding observed image sequence:*

$$-E_W(\mathbf{x}_i, \mathbf{y}_i) = \sum_{j=1}^{L_i} \log \psi'(y_{ij}) + \sum_{j=1}^{L_i-1}\sum_{c=1}^{C}\sum_{c'=1}^{C} W_{cc'}^T[y_{ij}=c][y_{ij+1}=c'] = \sum_{j=1}^{L_i} \log \psi'(y_{ij}) + \sum_{j=1}^{L_i-1} W_{y_{ij}, y_{ij+1}}^T$$

.

1.3 (*6 points*): *For the first three test words, use exhaustive summation over all possible character label sequences to compute the value of the log partition function for the CRF model after conditioning on the corresponding observed image sequence. Report the value you compute.*

1.4 (*6 points*): *For the first three test words, compute the most likely joint labeling (character sequence) word. Report both the labeling and its probability under the model.*

1.5 (*4 points*): *For the first test word only, compute the marginal probability distribution over character labels for each position in the word. Report each marginal distribution using a table.*

Note: the results contain many very small values, so please report in scientific notation such as "6.06e-12" instead of "0". If you're having trouble with this, try using the "g" format code in Python str.format or "%g" in sprintf-style formatting (or other methods). This may hold for other questions too.

Question 2. (*40 points*) **Sum-Product Message Passing:** In this question, you will implement the sum-product inference algorithm for the CRF model. The code packages provides a pre-trained model for the OCR task including the feature parameters (*feature-params.txt*) and the label-label transition parameters (*transition-params.txt*). Use these parameters to answer the following questions.

2.1 (*10 points*): *For the first test word only, condition on the observed image sequence to obtain a chain-structured Markov network. Compute the* ***log-space*** *messages* $m_{1\to 2}(Y_2)$, $m_{2\to 1}(Y_1)$, $m_{2\to 3}(Y_3)$, $m_{3\to 2}(Y_2)$. *Report the value of each message in a table.*

2.2 (*15 points*): *For the first test word only, use the computed messages to compute marginal probability distributions. Report single variable marginals over each position in the word as a table. Represent pairwise marginals over each adjacent node pairs as three tables, and report only the* 3×3 *block of entries between the labels "t,a,h" in each table.*

2.3 (*15 points*): *Generalize the steps given above to compute the single variable and pairwise marginal probabilities for any sequence of input images. Apply your completed algorithm to compute the marginal probability distribution over the character labels given the image sequences for each word in the test set. For each position in each test word, predict the character with maximum probability. List your predictions for the first five test sequences. In addition, use the true character labels in test_words.txt to compute the average character-level accuracy over the complete test set. Report the accuracy that you find to three decimal places.*

Question 3. (*34 points*) **Maximum Likelihood Learning Derivation:** In this problem, you will derive the maximum likelihood learning algorithm for conditional random field models.

3.1 (*8 points*): *Write down the average log likelihood function for the CRF given a data set consisting of* N *image sequences* $\mathbf{x}_i$ *and label sequences* $\mathbf{y}_i$.

3.2 (*8 points*): *Derive the derivative of the average log likelihood function with respect to the feature parameter* W^F_{cf}. *Show your work.*

3.3 (*8 points*): *Derive the derivative of the average log likelihood function with respect to the transition parameter* $W^T_{cc'}$. *Show your work.*

3.4 (*8 points*): *Explain how, as a byproduct of the sum-product algorithm's computation of the single-variable and pairwise marginal probabilities, you can efficiently compute both the value of the log-likelihood function and the values of the above derivatives.*

3.5 (*2 points*): *Using a data set consisting of the first 50 training data cases only, compute the average log likelihood of the true label sequences given the image sequences using the supplied model parameters.*

3.6 (*0 points*): *Using a data set consisting of the first 50 training data cases only, compute the derivative with respect to each model parameter of the average log likelihood of the true label sequences given the image sequences using the supplied model parameters. There is nothing to hand in for this question, but we will provide the solution to help you debug your code. These files are in the model folder and called feature-gradient.txt and transition-gradient.txt.*

Question 4. (*6 points*) Numerical Optimization Warm-Up:

In part B of the assignment, you will implement the above learning algorithm using a numerical optimizer to maximize the log likelihood. In this question, you will experiment with optimizing a basic function.

4.1 (*3 points*): *Consider the objective function* $f_w(x, y) = -(1-x)^2 - 100(y-x^2)^2$. *Derive the derivatives of* $f(x, y)$ *with respect to* x *and* y *(the gradient function). Show your work.*

4.2 (*3 points*): *Select a numerical optimizer for the programming language you are using. If you haven't used it previously, study its documentation carefully. Implement the objective function and the gradient function in the form required by your numerical optimizer. Write code to use the optimizer to* ***maximize*** $f(x, y)$. *Report both the location of the maximum and the value of the objective function at the maximum.*

Note that most optimization functions minimize by default (e.g. functions in the scipy.optimize module). Getting this wrong is a very common bug.