Assignment 2

CS 512: Data Mining Principles (Spring 2021)
Instructor: Hanghang Tong

Release date: Mar. 20th, 2021
Due date: Apr. 25th, 2021

- This assignment will cover the content from lecture slides #3 (Clustering), #4 (Deep Learning), #5 (Outlier Detection), and #6 (Network and Graph Connectivity).
- Feel free to talk to other members in the class when doing the homework. We care more about whether you learn how to solve the problem entirely on your own than you demonstrate that you solved it. You should, however, write down your solution yourself. Please try to keep the solution brief and clear.
- Please use Piazza first if you have questions about the homework. Also feel free to send us e-mails and come to office hours.
- The homework is due at 11:59 PM on the due date. We will be using Compass (`http://compass2g.illinois.edu`) for collecting assignments. **Please do not hand in a scan of your handwritten solution, only the typed solution (e.g., Microsoft Word, Latex, etc) will be graded**. Contact the TAs if you are having technical difficulties in submitting the assignment. We do **NOT** accept late homework!
- The solution report should be submitted as a **single** pdf file using the name convention `yourNetid_HW2.pdf`. If you use additional source code (Python/Java/C++ are all acceptable) for solving problems, you are required to submit them and use the file names to identify the corresponding questions. For instance, '`yourNetid_HW2_problem1.py`' refers to the python source code for Problem 1 for HW 2. Compress all the files (pdf and source code files) into one zip file. Submit the compressed file ONLY.
- For each question, you will NOT get full credits if you only give out a final result. Necessary calculation steps are required. If the result is not an integer, round your result to 2 decimal places.

Problem 1. Deep Learning (35 points)

(a) **Short Answers.**

(1) (3 points) What is overfitting? List two techniques that can mitigate this issue and briefly introduce why they can handle overfitting.

(2) (3 points) Compare LSTM and GRU, what do they have in common and what are the differences between them?

(b) **Backpropagation.** Given the neural network architecture as shown in Figure 1 where U_1, U_2 are from the input layer, $U_3 - U_9$ are hidden units and U_{10} is output unit. Assume *sigmoid* is the activation for all units (i.e., $O_k = \frac{1}{1+e^{-I_k}}$) and $L(T, O) = \frac{1}{2}(O - T)^2$ is the loss where T is the target value and O is the final output (i.e., $O = O_{10}$).

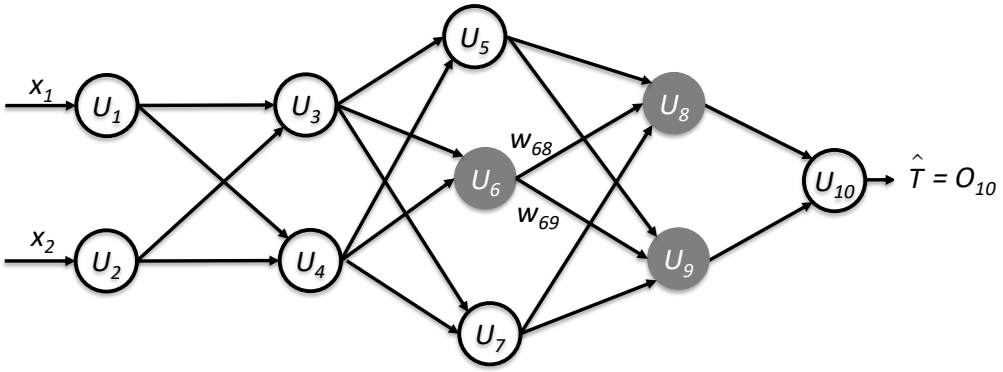

Figure 1: Feed-forward Neural Network

(1) (6 points) Read the section about backpropagation algorithm (11.1.2) in Chapter 11 of the textbook. Prove (1) Eq. (11.4) for the hidden unit U_6 in Figure 1, i.e., $\delta_6 = O_6(1 - O_6)(\delta_8 w_{68} + \delta_9 w_{69})$; (2) Eq. (11.5) for updating the weights, i.e., $\Delta w_{ij} = \eta \delta_j O_i$ where η is the learning rate. (hints: (1) chain rule; (2) take the derivative of loss w.r.t. w_{ij}.)

(2) (3 points) Now we change the loss function to $L(T, O) = -T \log O - (1 - T) \log (1 - O)$ (i.e., cross-entropy loss), derive the equation which computes the error δ for the output unit (U_{10}).

(c) **2D Convolution.** Given the input and two kernels as shown in Figure 2.

(1) (4 points) Compute the feature maps by applying kernel K_1, K_2 (stride = 1), respectively.

(2) (4 points) Compute the new feature maps after we apply average pooling with 2×2 filter to the obtained feature maps from (1).

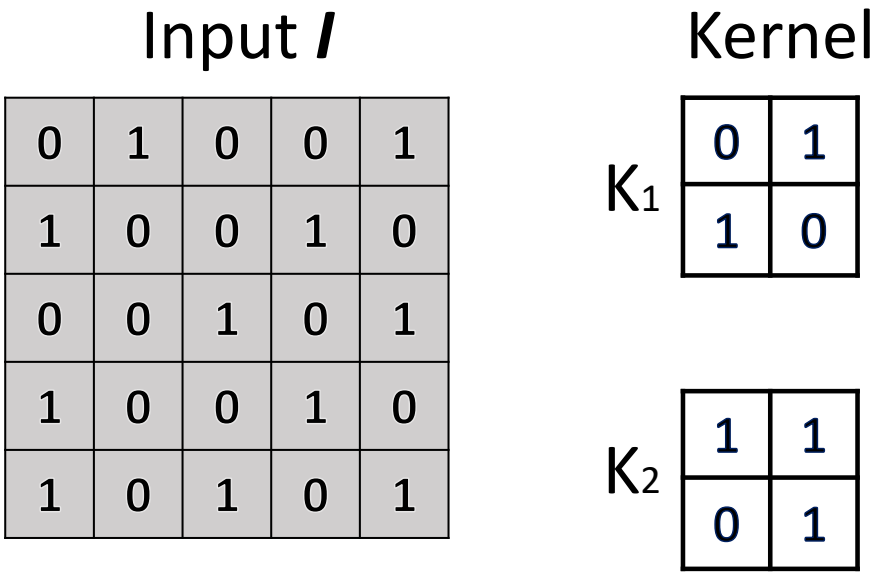

Figure 2: Input data and kernels for 2D Convolution.

(d) (12 points) **Graph Convolutional Networks.** In this question, we aim to implement the graph convolutional networks (GCNs)[1] equipped with a two-layer architecture for semi-supervised node classification on *Cora* dataset (refer to README for detailed data description). (*Remarks.* Deep learning frameworks are allowed to use.)

First, construct the undirected graph based on citation relations (i.e., there exists one undirected edge between two papers if one paper cites another). The node attribute can be represented by word occurrence. Randomly split the nodes for training, validation and testing (e.g., $60\%, 20\%, 20\%$, respectively). Use ReLU and softmax functions as the nonlinear activation functions of the first and second layers, respectively. For semi-supervised node classification, use cross-entropy loss function.

(1) Is this task inductive learning or transductive learning?

(2) Implement this model, then evaluate and report the node classification accuracy.

(3) Try different optimizer with different adaptive learning rate strategies and compare the results.

(4) If the number of GCN layers increases, how the model performance changes in terms of quality and efficiency?

[1]Kipf, Thomas N., and Max Welling. "Semi-Supervised Classification with Graph Convolutional Networks." (2016).

Problem 2. Outlier Detection (15 points)

(a) **Short Answers**

(1) (2 points) From the perspective of outlier detection methods, give two possible definitions of outliers.

(2) (3 points) Given a set of scalar values (e.g., $\{0.3, -0.1, 0.5, 0.8, -0.6, 15.5\}$), we now define an outlier is the value that is extremely larger than the majority of values (15.5 in this case). What is the disadvantage of this definition and how to modify this definition for outliers? Try providing an example to elaborate.

(b) (10 points) **Implementation.** In this question, we aim to get familiar with outlier detection algorithms through practice. The two multi-dimensional point datasets used in this question are **vowel.mat** and **satimage-2.mat**[2]. PyOD[3] is a Python toolkit for outlier detection and has included many classical outlier detection algorithms. For each dataset, try at least three algorithms including at least one neural networks-based method and compare the performance under the same setting (e.g., same training/validation/testing splitting, learning setting). Since the dataset is highly imbalanced, the following evaluation metrics should be used,

- Precision/Recall @K (e.g., precision/recall @10, 20, 50), which measure the precision/recall for the top-K ranked points according to algorithm output.
- Area under the ROC curve where ROC curve is obtained by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings.

[2]the detailed description can be found at http://odds.cs.stonybrook.edu/japanese-vowels-data/ and http://odds.cs.stonybrook.edu/satimage-2-dataset/, respectively.

[3]https://github.com/yzhao062/pyod

Problem 3. Gaussian Mixture Model (GMM) (25 points)

(a) (5 pts) Compared with **K-Means**, what are the advantages and disadvantages of Gaussian mixture model?

(b) (5 pts) **E step.** The intuition of Gaussian mixture model is that the observed data samples are all generated based on the K latent Gaussian models. Given K Gaussian models, we have:

$$P(Z_i = k) = \pi_k \tag{1}$$

and

$$P(x_i|Z_i = k) = \mathcal{N}(x_i|\mu_k, \sigma_k) \tag{2}$$

where $P(Z_i = k)$ denotes the prior probability. μ_k and σ_k are the mean and variance of the k-th component. Try to prove that the posterior probability of the k-th component $\tau_{i,k}$ is:

$$\tau_{i,k} = P(Z_i = k|x_i) = \frac{\pi_k \mathcal{N}(x_i|\mu_k, \sigma_k)}{\sum_k \pi_k \mathcal{N}(x_i|\mu_k, \sigma_k)} \tag{3}$$

(c) (15 pts) **M step.** We have the likelihood of GMM as follows:

$$P(X, Z|\pi, \mu, \sigma) = \prod_i \prod_k \pi_k^{I(Z_i=k)} \mathcal{N}(x_i|\mu_k, \sigma_k)^{I(Z_i=k)} \tag{4}$$

where $\pi = \{\pi_1, \ldots, \pi_k\}$, $\mu = \{\mu_1, \ldots, \mu_k\}$, $\sigma = \{\sigma_1, \ldots, \sigma_k\}$, $I(Z_i = k)$ denotes the true probability that x_i belongs to the k-th component. By replacing $I(Z_i = k)$ with $\tau_{i,k}$ we have the likelihood as follows:

$$P(X, Z|\pi, \mu, \sigma) = \prod_i \prod_k \pi_k^{\tau_{i,k}} \mathcal{N}(x_i|\mu_k, \sigma_k)^{\tau_{i,k}} \tag{5}$$

Based on the maximum likelihood estimation (MLE), try to prove that the new estimations of μ_k, σ_k, π_k are:

$$\mu_k = \frac{\sum_i \tau_{i,k} x_i}{\sum_i \tau_{i,k}}, \tag{6}$$

$$\sigma_k^2 = \frac{\sum_i \tau_{i,k}(x_i - \mu_k)^2}{\sum_i \tau_{i,k}}, \tag{7}$$

$$\pi_k = \frac{1}{N} \sum_i \tau_{i,k}, \tag{8}$$

where N is the total number of data samples. [Hint: $\sum_k \pi_k = 1$]

Problem 4. Graph Connectivity and SIS model (25 points)

(a) [10 pts] Implement the Susceptible-Infected-Susceptible (SIS) model on the given **social network** dataset (which is an edge list representing an unweighted and undirected network). The infection rate $\beta = 0.08$ which indicates that if a susceptible node has n infected neighbors, the probability that this susceptible node will be infected in the next time step is $1 - (1 - \beta)^n$. The recovery rate $\sigma = 0.02$ which indicates from time step t to time step $t + 1$, the probability of the infected nodes recovering into the susceptible nodes is σ. This specific procedure from time step t to time step $t + 1$ is that (1) the virus propagates along the network based on β (the infected nodes will keep infected) and (2) all the infected nodes will recover into susceptible nodes probability σ. We set the first 300 nodes (i.e. #0 to #299 nodes) as the initially infected nodes and set the remaining nodes as the initially susceptible nodes. Report the results with two parts: (1) a line chart whose x-axis denotes time steps, y-axis denotes the number of nodes, one line represents the number of susceptible nodes and the other line represents the number of infected nodes and (2) the average number of infected nodes after the convergence (e.g., the average number of infected nodes from iteration 200 to 300). [*Remarks:* General 3-rd party packages are allowed but directly calling existing SIS model on network is **NOT allowed**.]

(b) [5 pts] Based on problem (a), we have 200 vaccinations which can prevent 200 nodes from infection (i.e., always stay in the 'susceptible' state). Try following two strategies: (1) distribute vaccinations randomly, and (2) distribute vaccinations to 200 nodes with the highest degree. For each of the strategies, report the result with the same format as in problem (a).

(c) [10 pts] Try other strategies (excluding random and degree-based methods in problem (b)) to distribute the 200 vaccinations and report results with the line charts. The score is based on (1) [4 pts] the final number of infected nodes and (2) [6 pts] the number of strategies in the implementation (2 points for each strategy). [*Remarks:* General 3-rd party packages are allowed but directly calling existing strategies is **NOT allowed**.]