Assignment 2

CS5370: Deep Learning for Vision
IIT-Hyderabad
Jul-Nov 2019

Max Marks: 25
Due: 22nd Sep 2019 11:59 pm

Instructions

- Please use Google Classroom to upload your submission by the deadline mentioned above. Your submission should comprise of a single ZIP file, named `<Your_Roll_No>_Assign2`, with all your solutions, including code.
- For late submissions, 10% is deducted for each day (including weekend) late after an assignment is due. Note that each student begins the course with 7 grace days for late submission of assignments. Late submissions will automatically use your grace days balance, if you have any left. You can see your balance on the CS5370 Marks and Grace Days document under the course Google drive.
- You have to use PYTHON for the programming questions.
- Please read the department plagiarism policy. Do not engage in any form of cheating - strict penalties will be imposed for both givers and takers. Please talk to instructor or TA if you have concerns.

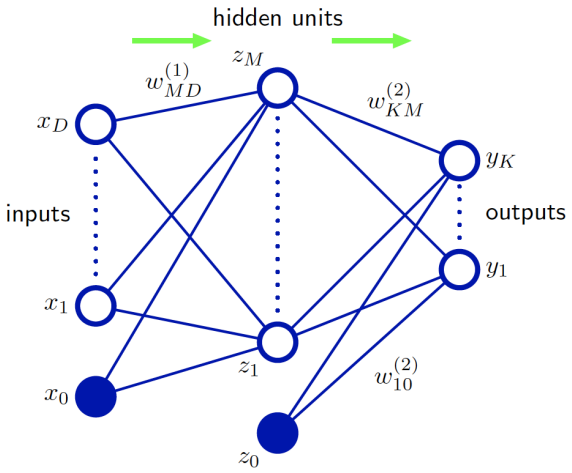
1 Exercises

- For theory questions denoted by **(T)** against the question, we would prefer you type your answer in LaTeX and upload your PDF - it is a good learning by itself. If you absolutely need, you can handwrite and upload your scanned report, but please ensure the answers are readable - you will be accountable for the readability if you choose this option!
- For programming questions, denoted by **(P)** against the question, please upload your code, and include your observations in a report.

1. **(T)** ***(2 marks)*** Consider a two-layer network of the form shown the figure below. Let us add extra parameters corresponding to skip-layer connections that go directly from the inputs to the outputs. Write down the equations for the derivatives of the error function with respect to these additional parameters.

2. **(T)** ***(3 marks)*** Consider an error function $E(\mathbf{w})$ of a neural network, and the Taylor series approximation of $E(\mathbf{w})$ around a particular weight configuration $\widetilde{\mathbf{w}}$ as below (ignoring higher order terms):

$$E(\mathbf{w}) \approx E(\widetilde{\mathbf{w}}) + (\mathbf{w} - \widetilde{\mathbf{w}})^T \nabla E|_{w=\widetilde{w}} + \frac{1}{2}(\mathbf{w} - \widetilde{\mathbf{w}})^T \mathbf{H} (\mathbf{w} - \widetilde{\mathbf{w}})$$



where $\mathbf{H}$ is the Hessian. Show that as a consequence of the symmetry of the Hessian matrix $\mathbf{H}$, the number of independent elements in the quadratic error function $E(\mathbf{w})$ is given by $W(W+3)/2$.

3. **(T)** ***(4 marks)*** Given a set of data points $\{\mathbf{x}_i : i = 1, \cdots, n\}$, we define the *convex hull* to be the set of all points $\mathbf{x}$ given by:

$$\mathbf{x} = \sum_i \alpha_i \mathbf{x}_i$$

where $\alpha_i \geq 0$ and $\sum_i \alpha_i = 1$. Consider a second set of data points $\{\mathbf{z}_i : i = 1, \cdots, n\}$ and its corresponding convex hull. The two sets of points will be linearly separable if there exists a vector, $\mathbf{w}$ and a scalar w_0 such that $\mathbf{w}^T\mathbf{x}_i + w_0 > 0$ for all $\{\mathbf{x}_i\}$, and $\mathbf{w}^T\mathbf{z}_i + w_0 < 0$ for all $\{\mathbf{z}_i\}$. Show that: (i) if their convex hulls intersect, the two sets of points cannot be linearly separable, and (ii) conversely that if they are linearly separable, their convex hulls do not intersect.

4. **(P)** ***(4+2+3=9 marks)*** Generate a two-dimensional data set consisting of a small number of vectors ($\sim 100-200$), each belonging to one of two classes. Write a numerical implementation of the perceptron learning algorithm, and plot both the data points and the decision boundary after every iteration.

   - Explore the behaviour of the algorithm both for TWO data sets which are linearly separable and for TWO datasets which are not linearly separable.
   - How dependent are the decision boundaries on the initial values of the weights? What do you see for each of the datasets when the initial weights are changed, and why?
   - Compare the final decision boundary in each of the above datasets against the decision boundary obtained using a SVM and logistic regression. What differences do you see and why?

   Include both your code and your observations in your submission.

5. **(P)** ***(7 marks)*** Consider the House Price Prediction dataset (an ongoing competition) on Kaggle:
   `https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data`
   Your objective is to use multi-layer perceptrons (multi-layer feedforward neural networks) for this regression problem. You can use `PyTorch` (preferred) or `sklearn` to work on this problem. You are free to decide your network architecture, activation functions, loss function, regularizer, etc - but only fully connected layers (no convolutional layers). (You are welcome to upload your results and participate in the Kaggle competition if you like!)

   Please upload your top 3 results and the corresponding configurations (network architecture, loss function, regularizer, optimizer, and relevant choice details). Give reasons for what helped your winning result.