Final Exam: Deep Learning: Spring 2015

Yann LeCun and Christian Puhrsch

May 18, 2015

1. (20 points) **General questions:**
   (a) (5 points) A number of theorems tell us that, under mild conditions, any reasonably well-behaved function $y = g(X)$ can be approximated as close as we want by a two-layer network, i.e. a function of the form $y = \sum_{i=1}^{K} U_i \sigma(W_i, X)$, where σ is a non-linear scalar function. Then what's the point of deep learning and why do we need multiple layers?
   (b) (5 points) why do the layers in a deep architecture need to be non linear?
   (c) (5 points) What are the basic modules that are used in a typical convolutional nets?
   (d) (5 points) What is a siamese network?

2. (20 points) **Pooling:** Pooling units take n values x_i, $i \in [1, n]$ and compute a scalar output whose value is invariant to permutations of the inputs.
   (a) (10 points) The Lp-pooling module takes positive inputs and computes $y = (\sum_i x_i^p)^{1/p}$. Assuming we know $y' = \frac{\partial L}{\partial y}$ what is $x_i' = \frac{\partial L}{\partial x_i}$
   (b) (10 points) The log-average module computes $y = 1/\beta \log(1/n \sum_i \exp(\beta x_i))$. Assuming we know $y' = \frac{\partial L}{\partial y}$ what is $x_i' = \frac{\partial L}{\partial x_i}$

3. (20 points) **ConvNet basics:** A 1D convolutional net has an input of size $p \times n$ (i.e. p features of size n each). The first layer is a convolutional layer with f output feature maps and kernels of size k that connect all p input feature maps to all f output feature maps.
   (a) (5 points) give a formula for the size of the output feature maps m?
   (b) (5 points) give a formula for the number of parameters (independent weights) in this layer (not counting biases)?
   (c) (5 points) give a formula for the number of multiply-accumulate operations to do a forward propagate (not counting biases)?
   (d) (5 points) We decide to use a convolutional layer with stride s. Give a formula for the size of the output feature map

4. (30 points) **ConvNets: Object Detection** We want to build a safety system for cars that detects pedestrians from a single camera image. We have a large dataset of images some with pedestrians labelled by a bounding box, and a much larger set of images without any pedestrian. We want to be able to detect and locate pedestrians regardless of their distance and apparent size in the image. In our dataset, the images have a resolution of 900x300, and the height of the pedestrians varies between 30 pixels and 120 pixels, and their width is always less than half their height.

(a) (5 points) You must build a ConvNet architecture for training. What input size would you give it?

(b) (10 points) How would you prepare the data to train the ConvNet, i.e. how would you perform size normalization, cropping, selection of negative examples, class frequency equalization, etc).

(c) (5 points) Building a detector generally requires to use a "bootstrapping" method to reduce false negatives. Describe the method.

(d) (10 points) Once the ConvNet is trained, we can use it inside a detection system. Describe the system's overall architecture. Draw a block diagram that includes the major modules, indicating how the system deals with size variations and the possible presence of multiple pedestrians. Give some idea of the ConvNet architecture (no need for a lot of details, but give some information about the network architecture (input size, output size, etc) and the preprocessing and postprocessing.

5. (10 points) **ConvNets: Weak Supervision** We want to train a ConvNet to not only classify, but also localize objects in images. Unfortunately, our training set only has images labelled with the list of objects present in the image (we assume that at most one instance of each category is present in the image). We do not have any bounding box information. There is a solution to this "weak supervision" problem that involves using a log-sum-exp pooling layer with units of the type: $y = 1/\beta \log\left(\sum_i \exp(\beta z_i)\right)$.

   (a) (10 points) Describe what the network architecture would look like, and how a layer of these pooling units would be used.

6. (30 points) **Word, Text, and Image Embedding:**

   (a) (10 points) Image Embedding: We want to train a ConvNet to do face recognition or image similarity search. The idea is to train a ConvNet $Y = G(X, W)$ so that the output vectors $G(X_{pi}, W)$ and $G(X_{pj}, W)$ for two images of the same person p are nearby, while for two different persons p and q the output vectors $G(X_{pi}, W)$ and $G(X_{qk}, W)$ are far away from each other. Write a DrLIM-style loss function to do so.

   (b) (10 points) Word Embedding: GloVe performs a kind of factorization of the log of the word co-occurence matrix $M_{ij} = \log X_{ij}$, where X_{ij} is the number of times word j appears in the context of word i. Each word is associated with two vectors: W_j represent word j itself while $\bar{W}_i$ represent the influence of word i when it appears near other words. Write the typical objective function used to compute these vectors in GloVe.

   (c) (10 points) Text Embedding for Question-Answering: We want to train a question-answering system using embedding techniques. We already have vectors trained in an unsupervised manner that represent questions Q_i and answers A_j. We want to train two mapping $G_Q(Q)$ and $G_A(A)$ with a WSABIE-style ranking loss so that after training, the distance between output vectors can be used to determine if an answer matches a question. Write down a possible loss function.

7. (20 points) **Recurrent Nets:**

   (a) (5 points) What is the "vanishing or exploding gradient problem" in recurrent nets?

   (b) (5 points) Give a weight initialization method that can mitigate the vanishing or exploding gradient problem.

(c) (10 points) Recurrent nets are notoriously bad at "remembering" things for more than a few iterations. Give the names and quick descriptions of two methods that augment RNNs with a memory.

8. (30 points) **Energy-Based Learning: Weakly supervised object localization** We wish to build a pedestrian detector, but the dataset we are given is weakly labelled. Images are given to us together with a label (pedestrian or no pedestrian in the image), but we are not given any bounding box information. We decide to use an energy-based model with a latent variable for the location (we assume the pedestrians are all more or less the same size in the training images). The system can be seen as an energy function $E(X, Y, Z, W)$ where X is the input image, Y is the label (1 = pedestrian, -1 = no pedestrian), Z is the location of the pedestrian in the image, and W is a trainable parameter vector. Training images are all 450x150 pixels and pedestrians are never larger than 60 pixel tall.
   (a) (10 points) describe a possible architecture for the overall system.
   (b) (10 points) Give a possible loss function and describe the inference and learning procedures.

9. (20 points) **Unsupervised Learning and Auto-Encoders**
   (a) (10 points) Energy-based unsupervised learning consists in finding the parameters W of an energy function $F(Y, W)$ so that it takes the right "shape" in the space of Y. Describe what this "shape" should be.
   (b) (5 points) write the objective function and constraints for a sparse auto-encoder.
   (c) (5 points) what is a denoising auto-encoder?

10. (20 points) **Optimization:**
   (a) (10 points) A unit in a deep network has two inputs, both of which have zero mean. Input 1 has a standard deviation 1, while input 2 has standard deviation 3. To maximize the speed of convergence, what should be the ratio of the learning rate for Weight 1 (connected to Input 1) to the learnning rate for Weight 2 (connected to Input 2)? (you can assume that Input1 and Input2 are uncorrelated).
   (b) (5 points) A network $y = G(X, W)$ has K layers with ReLU non-linearities and a hinge loss at the top. For a particular sample pair (X, y) and a particular weight vector W, the loss function $L(X, y, W)$ is a polynomial in W. What is its degree?
   (c) (5 points) In a fully-connected network that has been trained to a local minimum of the objective, one can swap the position of two hidden units, carrying their incoming and outgoing weight with them, without changing the input-output function. This points to existence of multiple equivalent local minima. What other characteristics of the objective function landscape does this symmetry suggest?