CS231n: CNNs for Visual Recognition **Spring 2017**

Lecture 3

Course Coordinator: Prof. Fei-Fei Li *Scribes: Akash Gupta*

- **Multi - class SVM for image classification** :

- Multiclass SVM Loss (Hinge Loss): $L_i = \sum_{j \neq y_i} max(0, s_j - s_{y_i} + 1)$
- L = $\frac{1}{N} \sum_{i=1}^{N} L_i$

Q) *Why would you ever consider using a squared loss instead of a non-squared loss?*
A) Depends on how much we care about different categories of errors. If an example has been classified as bad (huge value of loss), using squared loss will square this loss - so really really bad. Whereas using Hinge loss will not make much difference.

Q) *Are there multiple Ws for which L = 0? If Yes, how to choose W?*
A) Yes, 2W also gives L = 0. We need such W so that the classifier performs well on the test data. Sometimes W fits too well on the training data but not on test data (Case where we need a simpler classifier for test examples). We solve this by adding the regularization term:

$$L(W) = \frac{1}{N} \sum_{i=1}^{N} L_i(f(x_i, W), y_i) + \lambda R(W)$$

** Occam's Razor - "*Among competing hypothesis, the simplest is the best*"

- Types of regularization: L1, L2, Elastic net (L1 + L2), Max Norm, Dropout, Batch Norm, Stochastic depth

Q) *x = [1,1,1,1], Two Ws given W1: [1,0,0,0], W2: [0.25,0.25,0.25,0.25]. Which one should L2 regression prefer?*
A) It will prefer W2 since the L2 norm is smaller. The idea for L2 regularization is kinda spreading the influence of weights over all the Xs rather than depending on only few Ws. On the contrary, L1 regularization has opposite interpretation where more 0s in W(less parameters) correspond to a simpler model. L1 prefers sparse solution whereas L2 prefers spreading over all the Xs to get a less complex model.

- **(Softmax Classifier) Multinomial Logistic Regression**:

$$L_i = -log(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}})$$

- min value = 0 (when normalized prob = 1) - practically not possible
- max value = 0 (when normalized prob = 0) - practically not possible

- **Optimization**:

Various methods -

- Random search - really bad, don't use
- Follow the slope → calculate the derivative → calculate gradient (vector of partial derivatives) for multiple dimensions → Numerical Gradient calculation using finite differences method → really slow, but a really good debugging tool (Gradient checking)
- Analytic gradient - exact, fast, error - prone
  - Vanilla Gradient Descent.
  - Gradient Descent with Momentum.
  - Adam Optimizer
  - and many more....
- Use Mini-batch GD as it is much faster than plain GD. In plain GD, one update to weights has to take a whole pass through the dataset which is time-consuming. Favourable values of batches - in powers of 2 (64, 128, 256,....)

Note: Checkout the interactive Web Demo for better intuition on above concepts.

- **Image features**:

- Feeding raw pixels to linear classifiers is not a good idea instead do some feature representation tasks and then feed it there.
- E.g. - Histogram of Oriented Gradients(HOGs) (histogram of edges in the image) - common for object recognition
- E.g. - Bag of Words(Extract random patches → cluster them → create histogram of visual words (codebook) → BoWs) - common in NLP.

- Pipeline: Image → Image Feature Extraction → Linear Classifier training → Prediction.

- Teaser: CNNs are not much different than this as everything is combined into CNN training.

Inline Questions:

− > The SVM does not care about the details of the individual scores whereas the Softmax classifier is never fully happy with the scores it produces: the correct class could always have a higher probability and the incorrect classes always a lower probability and the loss would always get better.