

Lecture Two - Stanford CS231N

Name: Eli Andrew

• Image Classification

- $L1$ Distance: $d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$
- $L2$ Distance: $d_2(I_1, I_2) = \sqrt{\sum_p (I_1^p - I_2^p)^2}$
- $L2$ distance prefers many small differences to one big one (it's more forgiving)
- $L1$ and $L2$ are most common forms of the pnorm: $\|x\|_p = (|x_1|^p + \dots + |x_n|^p)^{\frac{1}{p}}$

• A Few Useful Things to Know about Machine Learning

- Key criteria for selecting representation is which kinds of knowledge are easily expressed in it
- For example: if we know a lot about what makes examples similar then instance-based methods (SVM, kNN) would be a good choice. If we know about probabilistic dependencies then graphical models are a good fit (CRF). And if we know about preconditions for each field then IF ... THEN rules may be best.
- Strong false assumptions (assuming independence) can be better than weak true ones, because the learner with the latter needs more data to avoid overfitting.
- Counter to the curse of dimensionality is the blessing of non-uniformity. This states that examples are not uniformly spread throughout the instance space, but rather on or near a low-dimensional manifold. For example: hand written digit images make up a space much smaller than the space of all possible images of the same size.
- Features may be irrelevant in isolation but relevant in combination. For example: if the function is an XOR.
- Dumb algorithm with lots of data beats a clever algorithm with modest amounts.

• Linear Classification

- Linear classifier form: $f(x_i, W, b) = Wx_i + b$
- Example:
 - * Training data: images $x_i \in R^D$ each associated with label y_i
 - * Here: $i = 1 \dots N$ and $y_i \in 1 \dots K$ so we have N training examples and K distinct categories (in CIFAR-10 $N = 50,000$ and $K = 10$)
 - * Each picture x has it's pixels flattened out into size $[D, 1]$
 - * Weight (or parameter) matrix W is of size $[K, D]$, and bias b is $[K, 1]$
- Important things to note:

- * W is evaluating all K classifiers in parallel, where each row of W corresponds to the classifier for that class (row 0 is classifier for class 0)
- * W, b are our the only things we can control (data x_i is fixed)
- Linear classifiers can be interpreted as template matching, where each classifier learns a template for its class and uses the inner product between the example and its class to determine its score
- The weights of the class can actually be plotted as a picture to view the template
- **Bias Trick**
 - * One way to simplify our classifier form is to absorb the bias into the Wx_i term
 - * This can be done by (1) adding the bias term as another column in our W so that it is now $[K, D + 1]$, and then (2) adding a one value row to the end of the x_i to make it $[D + 1, 1]$
 - * Now, we have $f(x_i, W) = Wx_i$ where we still end up with the shape: $[K, 1]$ but we don't have the extra bias term
- Image data preprocessing: it is important to both (1) center your data by subtracting the mean image from each image, and (2) scale each feature so that it ranges from $[-1, 1]$

• Loss functions and Other Classifiers:

– Multiclass SVM:

- * SVM loss is setup in a way that the classifier “wants” **the correct class for each image to have a score higher than the incorrect classes by some fixed margin Δ**
- * For each x_i (flattened image pixels) and y_i (correct class for image), we compute $s = f(x_i, W)$ where s_j is the j^{th} entry of s
- * The multiclass SVM loss for the i^{th} example is then:

$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + \Delta)$$

– Regularization:

- * Any classifier parameters W that correctly classify a set of points can be duplicated by replacing it with λW where $\lambda > 1$ (since it will uniformly stretch the loss)
- * We want to have a preference for one particular set of W
- * To do this we use a Regularization Penalty:

$$R(W) = \sum_k \sum_l W_{k,l}^2$$

* So, full loss function becomes:

$$L = \frac{1}{N} \sum_i L_i + \lambda R(W)$$

- **Softmax Classifier:**

- Generalization of the logistic regression classifier to multi-class