Special Topics in Data Science Spring 2024

Lecture 10 — Data Augmentation

Prof. Qi Lei *Scribe: Zichun Xia, Jiawei Xu*

1 Overview

We will explore Data Augmentation, a method for generating new data from existing datasets while preserving their semantic meaning.

Topic today

1. Different types of data augmentation.
2. Different views to understand it.
3. How to make full use of it.

Goal

Reduce overfitting/improve generalization

2 Common Ways of Data Augmentation

Imagery data:

- Operating on a single input $T : X \to X$
  - Geometric Transform: Flipping, Rotation, Stretch, Zoom in/out, Cropping
  - Randomly Change: RGB color channels, contrast, brightness
  - Kernel Filters: Sharpness, blurring
  - Random Erasing
- Mixup: $T : X \times X \to X$
  - For instance:
  $$x_1, x_2 \to ax_1 + (1-a)x_2$$
  $$y_1, y_2 \to ay_1 + (1-a)y_2$$
  soft-labeled w/classification
- GAN(generative model): Generative Synthetic Data
- Neural Style Transfer: Improve robustness of the trained model.

Audio data:

- increasing noise
- change in pitch/speed
- shifting

Text Data Augmentation

- Word/sentence shuffling
- Paraphrasing
  - Word replacement
  - Syntax-tree manipulation
- Random word insertion
- Random word deletion

3 Theoretical Analysis

3.1 Adding Gaussian noise

Using l_2-loss:

$$(L = \mathbb{E}_{XY \sim P_{XY}}[(f(x) - y)^2]$$

$$L_\epsilon = E_{x_i \sim P_{X_i}} E_{\epsilon \sim N(0,\sigma^2 I)} \left[(f(x+\epsilon) - y)^2\right]$$

$$L_{n,m} := \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{m} \left(f(x_i + \epsilon_j) - y_i\right)^2 .$$

Notice:

$$x_i, y_i \sim P_{x,y}, \quad \epsilon_j \sim \mathcal{N}(0, \sigma^2 I).$$

3.1.1 Population Loss

Objective function when we add noise:

$$\begin{aligned}
L_\epsilon &= \mathbb{E}_{XY \sim P_{XY}} \mathbb{E}_{\epsilon \sim N(0,\sigma^2 I)}[(f(x+\epsilon) - y)^2] && (1) \\
&\approx \mathbb{E}_{XY \sim P_{XY}} \mathbb{E}_\epsilon [f(x) + \epsilon^T \nabla f(x) + ||\epsilon||^2 - y)^2] && (2) \\
&= const(O(\sigma^4)) + \mathbb{E}_{XY \sim P_{XY}} \mathbb{E}_\epsilon [(f(x) - y)^2 + 2\epsilon^T \nabla f(x)(f(x) - y) + (\epsilon^T \nabla f(x))^2] && (3) \\
&= \mathbb{E}_{XY} \mathbb{E}_\epsilon [(f(x) - y)^2] + 2\epsilon^T \mathbb{E}_{XY} \mathbb{E}_\epsilon [\nabla f(x)(f(x) - y)] + \mathbb{E}_{XY} \mathbb{E}_\epsilon [(\epsilon^T \nabla f(x))^2] && (4) \\
&= L + 0 + \sigma^2 \mathbb{E}_X ||\nabla f(x))||^2 && (5)
\end{aligned}$$

3.1.2 Empirical Loss

$$L_{nm} = \frac{1}{n}\sum_{i,j}(f(x_i, \epsilon_j) - y_j)^2 \tag{6}$$

3.2 Introducing invariance

? Considering G: a group of transformations (e.g., all rotations). For $g \in G$: X → X : x → gx, and $e \in G$ is the identity element of the group. Then, we have the orbit average of f(take the average of all the rotations on X):

$$\bar{f}_{nm} = E_g[f(gx)] = E_g[f(gx)] \tag{7}$$

4 Using/Training with Data Augmentation

? Empirical loss with DA:

4.1 Adding augmented data to the training data

The empirical loss would be as original:

$$\hat{L}_n = \frac{1}{nm}\sum_{i=1}^{n}\sum_{A\in\mathcal{A}} l(f(A(x_i)), y_i) \tag{8}$$

with (x_i, y_i) the original training data points, $\mathcal{A}$ is the set of augmentations, and m=$||\mathcal{A}||$.

4.2 Encourage data augmentation consistency

To encourage DA consistency, we can train the model based on an alternative empirical loss:

$$\hat{L}_n = \frac{1}{n}\sum_{i=1}^{n} l(f(x_i), y_i) + \lambda \cdot \sum_{A\in\mathcal{A}} d(h(x_i), h(A(x_i))) \tag{9}$$

Here, h is some function h is some invariant representation, predictor $f = w{\cdot}h$. $\lambda\Sigma_{A\in\mathcal{A}} d(h(x_i), h(A(x_i)))$ encourages the representation augmented data to be similar.

It is shown that DAC handles stronger data augmentation (with misspecification) more than DA.

Without misspecification:

$$A(x) = \begin{bmatrix} A_1(x_1) \\ A_2(x_1) \\ \vdots \\ A_m(x_1) \end{bmatrix}_{n\times(mtimesd} \quad , \quad M(x) = \begin{bmatrix} x_1 & x_1 & \dots & x_1 \\ x_2 & x_2 & \dots & x_2 \\ \vdots & \vdots & \ddots & \vdots \\ x_n & x_n & \dots & x_n \end{bmatrix}_{n\times d} .$$

With $m = |A|$, let $d_{aug} = rank(A(x) - M(x))$,
With DAC, we have:
(a) For linear function class:

$$E\left[L(\hat{f}_{DAC}) - L^*\right] \leq O\left(\frac{(d - d_{\text{aug}})\sigma^2}{n}\right).vs(DA : O\left(\frac{d\sigma^2}{n}\right)),$$

which shows a tighter upper bound compared to the naive case (without DAC, ie. using empirical loss with only the first term of the equation we have excess risk of DA > risk of DAC.

(b) w/ two-layer neural network

$$\text{(DAC)} \leq O\left(C_\omega\sqrt{\frac{(d - d_{\text{aux}})\sigma^2}{n}}\right). \leq O\left(C_\omega\sqrt{\frac{d\sigma^2}{n}}\right).$$

$$f_0 = (X \cdot B)_+ \cdot w.$$

where the predictor of the shallow network is $f_0 = (X \cdot B)_+ \cdot w$, B is orthogonal and $\|w\|_1 \leq C_w$.

5 DA as feature manipulation

? Imaging an image of a car driving on a straight road in the daytime, you can view the blue sky as the background. We have the features:
(a) easy and good to learn: the car body.
(b) hard but good to learn: small features.
(c) easy but bad to learn: spurious feature and irrelevant feature.
(d) hard and bad to learn: spurious feature and irrelevant feature which contribute to the gradient.

Analyze the trending dynamics of GD:
GD fit data with (a) and (c) first, which may leads to overfit to noise/ bad features.

DA: (b)=>(a), (c)=>(d)
Still, take example of an image of a car driving on a straight road in the daytime, you can view the blue sky as the background.

$$x = \text{a single sample point}$$

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{\text{patch}} \\ \vdots \\ x_p \end{bmatrix} \Rightarrow \begin{array}{ll} \text{corresponds to} & \text{D-patches} \\ \text{corresponds to} & \text{cor unde} \\ \text{corresponds to} & \text{cor body} \\ \text{corresponds to} & \text{road} \\ \text{corresponds to} & \text{cloud} \end{array}$$

Here x_p is the k-th good feature.

We have good features

$$x_k = \rho_k \cdot y \cdot k.$$

which includes small features hard to learn, and large features easy to learn.

We also have bad features

$$\sigma_\xi^2 \text{ is large. } \xi \sim \mathcal{N}\left(0, \frac{\sigma_\xi^2}{d} I\right).$$

Define the network as:

$$f(W; x) = \sum_c \sum_p \psi(w_c \cdot x_p)$$

With gradient flow on logistic regression objective of f:
Learning dynamics on good features:

$$\frac{d}{dx}(w_c \cdot v_k) \approx \rho_k \psi'(|w_c \cdot v_k|).$$

Learning on noise:

$$\frac{d}{dt} w_{c,\xi}^{(i)} \approx \frac{1}{n} \sigma_\xi^2 y^{(i)} \psi'(|w_{c,\xi}^{(i)}|).$$

6 Summary

DA (without misspecification):

good and hard converted to good and easy features

bad and easy converted to bad and hard features.

References