Homework 2

CS 525: Deep Learning

Jerry Duncan
jdunca51@vols.utk.edu

April 23, 2020

1 Problem 1

a) Data compression (compressing data to a latent representation) and data generation (we can generate data by generating a random latent vector)

b) This question depends on the use case of the auto-encoder and the task we're trying to complete. For example, if we're generating images then we might use Mean Squared Error to measure how close we are to the original image. Or if we're generating a vector of probabilities then we might use Cross Entropy to measure how many of our probabilities have changed.

c) Reconstruction loss + KL Divergence loss. KL Divergence loss is trying to enforce that all nodes in the latent space have a mean of 0 and variance of 1. It forces the network to learn wider, more normal distributions and prevents the network from "cheating" by learning a narrow distribution that isn't useful in generating data.

d) We can add noise to the inputs to prevent the auto-encoder from simply copying the input (and also over-fitting).

2 Problem 2

a) The loss for the discriminator is trying to maximize the categorical cross entropy between the real and generated distributions. Where the left term is the discriminator's prediction on the image from the dataset being real and the right is the discriminator's prediction on the generated image being real.

The loss for the generator is the right term in categorical cross entropy and is trying to minimize the probability of the discriminator's prediction on the generated image being correct.

b) Inception Score measures the quality and diversity of generated images. It's advantages are that it uses both an ideal label distribution and an ideal marginal distribution. It's disadvantages are that it's limited by what the classifier can detect, it doesn't measure interclass diversity, and it doesn't measure the difference from the training data.

Fretchet Inception Distance the difference of the distributions between the real data and the created data. It's advantages are that it's more robust to noise than Inception Score and won't deteriorate when images are highly noisy. It's disadvantages are that it assumes that the features are of Gaussian distribution which is often not guaranteed.

c) The first application is Single Image Super-Resolution. The input is a low resolution image. The loss function for the generator is comprised of two parts, one is the content loss that tries to make the image look similar to the high resolution image and the second is adversarial loss which tries to make the image look real. The architecture is a very deep generator network using residual blocks.

The second application is generating images from text. The input is a text embedding instead of class information for the generator. The loss function for the discriminator changes the right term of the cross entropy to instead be the mean of examples with real images with incorrect text and examples of fake images with correct text. The architecture for the generator accepts text embeddings and the discriminator has text embedding concatenated as extra channels when reaching 4x4 resolution.

3 Problem 3

a) In order to calculate the update to W_{hy}^{00}, we need to first calculate the loss of all outputs, $L_{total} = \sum_1^2 L_i$ where L_i is $-\sum_1^2 y_i log(\hat{y_i})$. The total update to W_{hy} is then calculated as $W_{hy} = W_{hy} - \frac{\partial L_{total}}{\partial W_{hy}}$. And $\frac{\partial L_{total}}{\partial W_{hy}^{00}} = \frac{\partial L_1}{\partial W_{hy}^{00}} + \frac{\partial L_2}{\partial W_{hy}^{00}}$. We'll need to define a few terms, c is the cross entropy loss, s is the softmax, and t is the tanh activation.

$$L_1 = c_1(s_1(t_1(x_1 \cdot W_{xh} + h_0 \cdot W_{hh}) * W_{hy}))$$
$$\frac{\partial L_1}{\partial W_{hy}} = \frac{\partial L_1}{\partial c_1}\frac{\partial c_1}{\partial s_1}\frac{\partial s_1}{\partial W_{hy}}$$
$$L_2 = c_2(s_2(t_2(t_1(x_1 \cdot W_{xh} + h_0 \cdot W_{hh}) \cdot W_{hh} + x_2 \cdot W_{xh}) \cdot W_{hy}))$$
$$\frac{\partial L_2}{\partial W_{hy}} = \frac{\partial L_2}{\partial c_2}\frac{\partial c_2}{\partial s_2}\frac{\partial s_2}{\partial W_{hy}}$$
$$\frac{\partial L_{total}}{\partial W_{hy}} = \frac{\partial L_1}{\partial c_1}\frac{\partial c_1}{\partial s_1}\frac{\partial s_1}{\partial W_{hy}} + \frac{\partial L_2}{\partial c_2}\frac{\partial c_2}{\partial s_2}\frac{\partial s_2}{\partial W_{hy}}$$

b) If we only have one output at $t = 2$ then we remove the loss from the first time step above.

$$L_2 = c_2(s_2(t_2(t_1(x_1 \cdot W_{xh} + h_0 \cdot W_{hh}) \cdot W_{hh} + x_2 \cdot W_{xh}) \cdot W_{hy}))$$

$$\frac{\partial L_2}{\partial W_{hy}} = \frac{\partial L_2}{\partial c_2}\frac{\partial c_2}{\partial s_2}\frac{\partial s_2}{\partial W_{hy}}$$

$$\frac{\partial L_{total}}{\partial W_{hy}} = \frac{\partial L_2}{\partial c_2}\frac{\partial c_2}{\partial s_2}\frac{\partial s_2}{\partial W_{hy}}$$

c) Vanilla RNNs suffer from vanishing gradients and LSTM is intended to solve this problem. They have internal gates that regulate the flow of information, like the cell state that is passed to each time stamp and the forget gate that decides what information is relevant to keep around.

d) The forget gate determines what information is no longer needed for the next time stamp. The input gate determines what information from the candidate cell state should be added to the next cell state. The output gate determines what part of the cell state will become the hidden state.