Learning the alignment through Denoising Autoencoders

October 18, 2019

1 Introduction

We are motivated by a new training principle of unsupervised learning of a representation based on the idea of making the learned representations robust to partial corruption of the input pattern. We investigate the following criterion: **Robustness to partial destruction of the input**, i.e. partially destroyed inputs should yield almost the same representation. The training procedure for the denoising auto-encoder (DAE) involves learning to recover a clean output from a corrupted version, a task known as a denoising. Our goal is to learn the constraints of an optimization problem using DAE. More precisely, we want to optimize the position of N rectangles in $\mathbb{R}^2$ in order these objects to look "aesthetically" right. First constraint we learn is the alignment. As a first experiment we want to learn how to align two rectangles using DAE. For this reason, we create two data sets. The original data: image with two rectangles aligned in the y-axis or x-axis (i.e horizontal or vertical alignment). The corrupted input: image with two rectangles slightly deviated from the alignment. The desired output: image with the two rectangles aligned, i.e. the original data. The idea is to learn the alignment through a small distortion of the original dataset.

2 Description of the Algorithm

2.1 Notation and set-up

The set-up we consider is the typical supervised learning setup with a training set of n pairs (input, target), $D_n = \{(x^{(1)}, t^{(1)}), \ldots, (x^{(n)}, t^{(n)})\}$, that we suppose to be an i.i.d. sample from an unknown distribution $q(X, T)$ with corresponding marginals $q(X), q(T)$. We use the following notation for the training of AE: s is the activation function, $\mathbf{W}, \mathbf{W}'$ are the weight matrices of the encoding and decoding part with dimensions $d' \times d$, $d \times d'$ respectively and $\mathbf{b}, \mathbf{b}'$ are the biases vectors of the encoding and decoding part with dimensions $1 \times d'$ and $1 \times d$ respectively.

2.2 The Denoising Autoencoder

To test our hypothesis and enforce robustness to partially destroyed inputs we modify the basic autoencoders. We will now train it to reconstruct a clean repaired input from a corrupted, partially distorted one. This is done by first corrupting the original input $\mathbf{x}$ to get a slightly distorted version $\tilde{\mathbf{x}}$, by means of a stochastic mapping $\tilde{\mathbf{x}} \sim q(\tilde{\mathbf{x}}|\mathbf{x})$. In our experiments we consider the following corrupting process: for each input $\mathbf{x}$, i.e. two aligned rectangles (vertically or horizontally), we slightly distort the alignment. The corrupted input $\tilde{\mathbf{x}}$ is then mapped, as with the basic autoencoder, to a hidden representation $\mathbf{y} = f_\theta(\tilde{\mathbf{x}}) = s(\mathbf{W}\mathbf{x} + \mathbf{b})$, parametrized by $\theta = (\mathbf{W}, \mathbf{b})$, from which we reconstruct $\mathbf{z} = g_{\theta'}(\mathbf{y}) = s(\mathbf{W}'\mathbf{y} + \mathbf{b}')$, parametrized by $\theta' = (\mathbf{W}', \mathbf{b}')$. The parameters are trained to minimize the reconstruction error over a training set, i.e. to have $\mathbf{z}$ as close as possible to the uncorrupted input. The objective function minimized by the stochastic gradient descent becomes:

$$\theta^*, \theta'^* = \underset{\theta,\theta'}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} L\left(\mathbf{x}^{(i)}, f_\theta(g_{\theta'}(\tilde{\mathbf{x}}))\right) \tag{1}$$

where $L(x, y) = \|x - y\|^2$. We take a gradient step towards reconstructing the uncorrupted version from the corrupted one.

3 Experiment

3.1 Dataset

We performed experiment with the proposed algorithm on learning the alignment through slight distorted rectangles. Input dimensionality $\mathbf{x} \in \{0, 1\}^d$, where $d = 28 \times 28 = 784$. We generate $n = 50,000$ data. First we produce a dataset with two aligned rectangles. More precisely, we generate randomly a rectangle and then we aligned another one either vertically or horizontally. Note that the process of alignment is random like the size of rectangle. Figure 1 shows the original data. Therefore, we corrupt this dataset by a slight distortion which is also random. Note that the distortion is controlled. We only allow a slight corruption. Figure 2 shows the corrupted input of the AE, from which we want to learn the alignment between two rectangles. The desired output should be as close as possible to the original dataset.

3.2 Implementation

We used a stacked AE as a building block to train our neural network, with the representation of k-th layer as an input for the $(k + 1)$-th layer trained after the k-th has been trained. Each layer is trained in order to minimize the criterion in (1). We applied a stacked AE with 2 encoding and decoding layers. Each layer is connected with a Conv2D net. Further, we worked with the ReLU as an activation function. Finally, we used the mean square error for

the reconstruction error and SGD for the optimizer step. The model has been trained for 200 epochs and batch size 100.

3.3 Experiment results

Figure 3 shows the output of AE after the training. It is clear that the AE learns alignment using corrupted data, i.e. two rectangles aligned. However, we notice that it does not learn the position of rectangles.

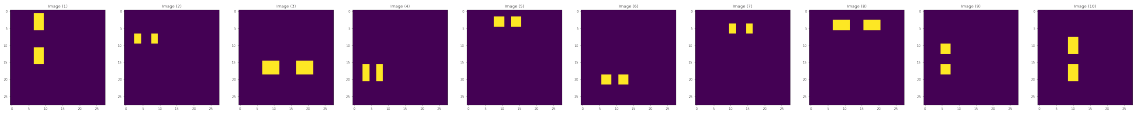

Figure 1: Original dataset, aligned rectangles

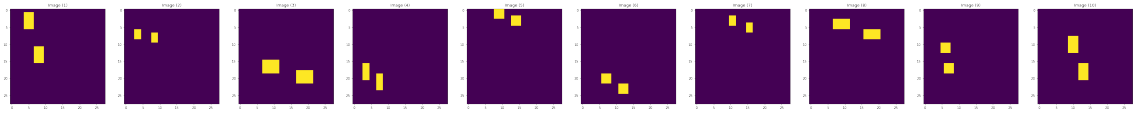

Figure 2: Corrupted input of the AE

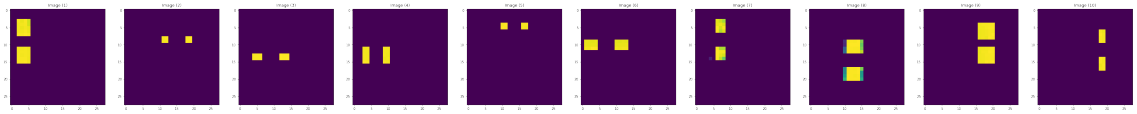

Figure 3: Output of the AE

4 Conclusion and Next Steps

We worked with a very simple training principle for autoencoders, based on the objective of undoing a corruption process, i.e. to align two rectangles using slight distortion of the position. The goal of learning representation is robustness of the output to small irrelevant changes. The empirical results support the following conclusions that the DAE learns alignment, but the output does not retain information regarding the original position of rectangles.