Disentangling hidden factors of variations in Deep Networks

Chaitra Jambigi (16951) [1]

Abstract

The aim of this project is to implement a model which can disentangle hidden factors of variations (or latent information) in data from the class label information. Thus we are trying to train an Autoencoder so that it can efficiently learn how to encode the latent information.

1. Introduction

Whenever we use a dataset, which has some label infomation, we generally use the label information for classification or any other particular task. Example using face dataset, by using label as identity information, we can effectively classify the face. However, the data has lot more information than just class labels. Example, in a person dataset, the images with same label might have different illumination conditions, or a single person's angle of view might be different in different images, or pose variations. These variations don't have any label information, but still they are present in data. Such variations in data can be called as hidden variables or latent variables and the labelled information can be called as the observed variables.

1.1. Need for Disentanglement

Now that we acknowledge the fact that hidden factors of variations are present in data, we need to question can that hidden data be useful for us in some way. For eg, in Computer vision applications, for Person re-identification problem, if we have same identity person with different pose, different head tilts, we need our algorithm to still classify that data as same person. If we can correctly disentangle the pose or tilt or basically variations in data from the actual labelled information, then the classification problem becomes much easier. Also if we can learn style, slant and height from the fonts, we can perform style transfer on new data. Thus, these applications motivate us to find some method such that Neural networks can learn to effectively disentangle factors of variations from data.

Approach used: This project implements the paper (Cheung et al., 2014) for disentangling the hidden factors of variations.

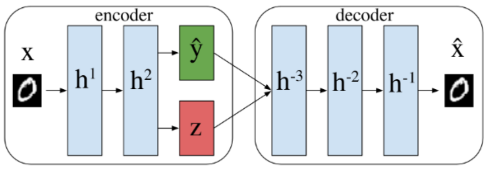

Figure 1. Encoder Decoder model which are jointly trained to reconstruct the input and produce the output

2. Implementation

We have tried to solve the above problem using an Autoencoder. Given an input $x \in \mathcal{R}^D$ and it's class label $y \in \mathcal{R}^L$ where L is the number of classes, class label contains some information about x. However using only the class labels, we cannot reconstruct back the data x. Thus here we are trying to feed the network a data x and training it to learn a latent representation or encoding or hidden data. The class labels are used to train the network to classify the data in a Supervised manner. Thus we have an output $\widehat{y}$ for every input class label y. We also have a latent variable output from the encoder $z \in \mathcal{R}^P$ z would be encoding the latent variable information. Thus in case of different font style data, p dimensional z would encode style, slant or height information. Using z and $\widehat{y}$, input x is reconstructed and network is trained to minimize this loss.

Encoder outputs: As shown in figure 1, Encoder gives 2 outputs, $\widehat{y}$ which is used to reduce the Cross entropy loss and z which encodes latent information and finally $\widehat{y}$ and z and given to decoder to reconstruct the input back $\widehat{x}$

$$F(x;\theta) = \{\widehat{y}, z\}$$

$$G(\widehat{y}, z; \phi) = \widehat{x}$$

where θ and ϕ are parameters of Encoder and Decoder respectively.

2.1. Learning

The Objective function to minimize during training is defined as the sum of 3 Loss functions,

$$\widehat{\theta}, \widehat{\phi} = \arg\min_{\theta,\phi} \sum_{\{x,y\}\in D} \| x - \widehat{x} \|^2 + \beta \sum_i y_i log(\widehat{y}_i) + \gamma C$$

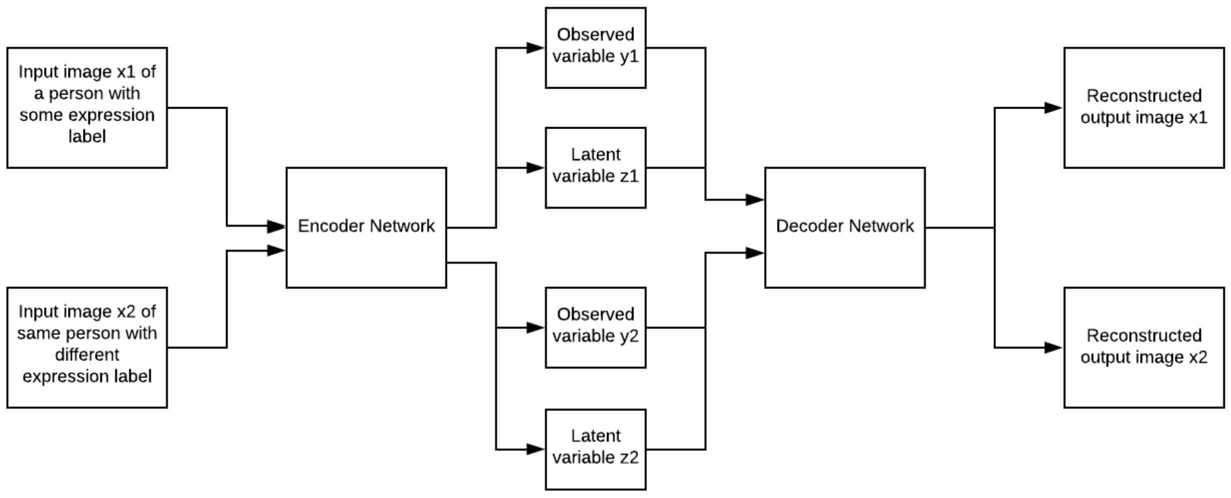
Input image x1 of a person with some expression label
Input image x2 of same person with different expression label
Encoder Network
Observed variable y1
Latent variable z1
Observed variable y2
Latent variable z2
Decoder Network
Reconstructed output image x1
Reconstructed output image x2

Figure 2. Block diagram for the proposed model

The first Loss is the mean squared error between the input image x and reconstructed output image $\widehat{x}$. The second term is the Cross entropy loss error or the supervised cost, to maximise the classification accuracy. The third loss used is the unsupervised Cross Covariance loss (XCov loss).

$$C(\widehat{y}^{1...N}, z^{1...N}) = \frac{1}{2}\sum_{ij}[\frac{1}{N}\sum_{n}(\widehat{y}_i^n - \overline{\widehat{y}}_i)(\widehat{z}_j^n - \overline{\widehat{z}}_j)]^2$$

This is basically added to keep the Covariance between the two components z and $\widehat{y}$ as low as possible. By keeping the XCov loss minimal, we are ensuring that the two variables don't mix with each other and are as seperate as possible. Thus this Autoencoder is trained in a semi-supervised approach where the class labels are provided.

3. Proposed Novelty

We know that the Encoder must embed or encode the latent information from given data. If we can ease it's task of doing correct encoding we can get better performance. So while training the network to learn the encoding weights, we can give 2 images which have same latent information, but having different labels. These 2 images x_1 and x_2 having similar latent information with different labels are given to same encoder and the embeddings or latent variable z_1 and z_2 obtained from both the images. And we know that both z_1 ad z_2 must be similar to each other because they of same identity. So we can add this similarity loss in the total Loss function so that the Encoder learns to encode the latent variable effectively. Block diagram for proposed approach is shown in Figure 2 Since now 2 images are taken in input, we have 2 Reconstruction loss, 2 Cross entropy loss, 2 XCov loss and a Similarity loss term. So modified Loss function becomes,

$$\widehat{\theta}, \widehat{\phi} = \underset{\theta,\phi}{\arg\min} \sum_{\{x,y,z\}\in D} \alpha \left\| x_1 - \widehat{x}_1 \right\|^2 + \alpha \left\| x_2 - \widehat{x}_2 \right\|^2$$

$$+\beta \sum_i y_{1i} log(\widehat{y_1}_i) + \beta \sum_i y_{2i} log(\widehat{y_2}_i) + \gamma C_1 + \gamma C_2$$

$$+\zeta \left\| z_1 - z_2 \right\|^2$$

4. Experimental Results

2 datasets were used to train the network. The first one was MNIST and the second one was, The Japanese Female Facial Expression (JAFFE) Dataset. Architectures used for the 2 datasets is shown in Table 1.

4.1. Datasets

MNIST Dataset

The MNIST database (LECUN) contains 60,000 training and 10,000 test images of handwritten digits from 0-9 of size 28x28.

JAFFE Database The JAFFE database (Cheng et al., 2010) contains 213 images of 7 facial expressions (6 basic facial expressions + 1 neutral) posed by 10 Japanese female models. Each image has been rated on 6 emotion adjectives by 60 Japanese subjects.

Table 1. Network Architectures Softmax (SM), Rectified Linear (ReLU)

MNIST	JAFFE DATASET	PROPOSED MODEL
500 RELU	2000 RELU	2048 RELU
500 RELU	2000 RELU	1024 RELU
10 SM, 2 LINEAR	7 SM, 793 LINEAR	512 RELU
500 RELU	2000 RELU	7 SM, 256 LINEAR
500 RELU	2000 RELU	512 RELU
784 LINEAR	2304 LINEAR	1024 RELU
		2048 RELU

4.2. Results

MNIST Results The 2 dimensional latent variable output from the encoder follows a Normal distribution with mean 0. The scatter plot shown in Figure 3 shows that z takes values from -0.5 to 0.5. Thus by varying the values of z in any 1 dimension, and freezing other dimension we can see different variations in the MNIST dataset. Figure 4 shows a test case, where the number 0 has higher width and number 9 has lower width. Latent variable of both the images is found by passing them through the encoder and the latent variables of both the images are swapped and decoded back. We can see that 0 has now lower width and 9 has higher width. Thus we have encoded the style information effectively.

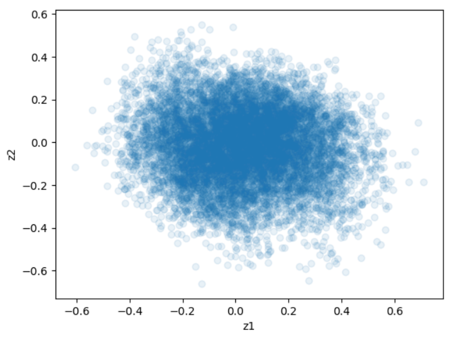

Figure 3. Scatter plot for 2 dimensional latent variable z

JAFFE Results JAFFE dataset with labels of expression was used to disentangle expression and identity information of person. Thus at the encoder output, we got 1 latent variable and 1 label variable. When the label variable was modified the decoder correctly changed the expression information on the identity. In short, it kept the identity information intact, which means latent variable z has correctly encoded the latent information. Test was conducted on 3 batches each of size 32, and output was observed. We got an accuracy of around 55% i.e. out of all 32 images with labels, the model successfully changed the expressions

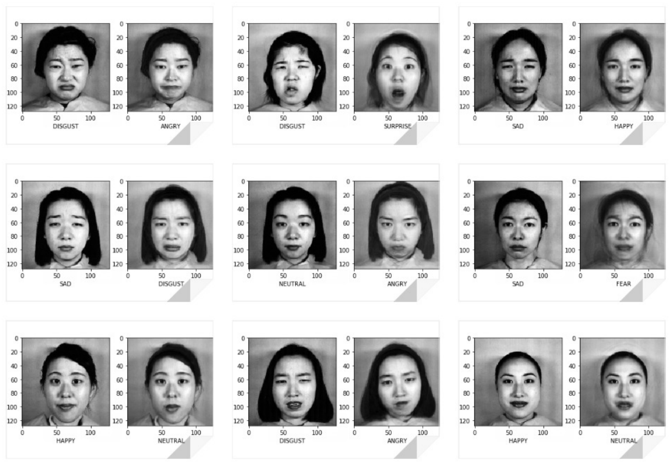

Figure 4. Results showing facial label change keeping latent variable intact on JAFFE dataset z

keeping identity information intact for just 55% of images. Classification accuracy was around 93%

Results for Proposed Model on JAFFE We need to give a pair of images for training in this model, such that the pair x_1 and x_2 belongs to same person but has different expression. Thus the entire dataset is split as pairs of 2 images manually. Thus for 213 images in this dataset we got around 2152 pairs of images. This data was split into training and testing data. Again test was conducted on 3 batches each of size 32, and output was observed. We got an accuracy of around 95% with this model, i.e. out of all images, 95% of them the model could change expression keeping identity intact. Thus we can say we have learnt the encoding effectively

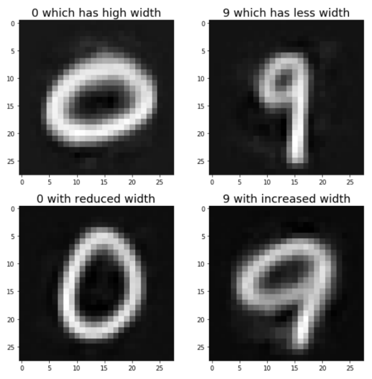

Figure 5. Swapping the latent variable of top left image with the top right image. Bottom left and bottom right images are the results

5. Conclusion

Thus we have shown that using Encoder Decoder pair we can effectively disentangle factors of variation in the data. Also if we know what are the factors of variation in data, then using the proposed model, we can train the model in such a way that the encoder learns to encode the latent variable effectively.

References

Cheng, F., Yu, J., and Xiong, H. Facial expression recognition in jaffe dataset based on gaussian process classification. *Trans. Neur. Netw.*, 21(10):1685–1690, October 2010. ISSN 1045-9227. doi: 10.1109/TNN.2010.2064176. URL `http://dx.doi.org/10.1109/TNN.2010.2064176`.

Cheung, B., Livezey, J. A., Bansal, A. K., and Olshausen, B. A. Discovering hidden factors of variation in deep networks. *CoRR*, abs/1412.6583, 2014.

LECUN, Y. The mnist database of handwritten digits. *http://yann.lecun.com/exdb/mnist/*. URL `https://ci.nii.ac.jp/naid/10027939599/en/`.