Unsupervised CNN Variational Auto-encoder for Image-based Recommender Systems

(Option 1)

20150880 Ji Yea Shim
20150531 KangHoon Lee
20193387 SungWon Yun

I. Introduction

Nowadays the predominant method of purchase is online shopping. Online shopping websites have recommender systems that suggest other products to customers to encourage purchases. Recommendation is based on product labels, or the visit or purchase history of previous customers. Imagine you check a pair of white athletic shoes with red stripes, and you would like to see other shoes of similar style. But that pair may be the only kind in the athletic shoes group. Or, all previous visitors may have checked other shoes of totally different style, such that the suggested recommendations does not satisfy your need. This example illustrates the weakness of the current recommender system, which is that it is not fit for recommending products based on their style.

We suggest our model, an unsupervised CNN variational autoencoder (VAE) for image-based recommender systems to compensate for the aforementioned problem. Style is a visual property where textual labeling is inefficient and varies by person. Unsupervised CNN-VAE network is able to train on image data without labeling, and so can be used for image (style) based recommender systems. The similarity is calculated based on the latent vector. In this project, we limited our database pool to shoes and show promising results. The model is available in our Github repository [1].

II. Method

1. Dataset preparation

UT Zappos50K provides a large collection of shoe images, pictured in the same orientation in white background [2]. We chose the ‘ut-zap50k-images-square.zip’ dataset, which has more than 50K images of size 136*136. There are four shoe categories (shoes, sandals, slippers, and boots), and subcategories based on shoe brands. We put the images of the four shoe categories together in one directory, and removed some faulty images that were NoneType. Among these images, 5k random images were chosen for training.

2. Variational Auto-encoder (VAE)

Figure 1 shows the structure of a conventional VAE network. VAE is a generative model whose purpose is to reconstruct input data by learning probability distribution.

Encoder network learns parameters of probability distribution from input data. The parameters are mean (μ) and standard deviation (σ) for a Gaussian distribution. The decoder network recreates the input image based on the sampled latent vector from the probability distribution.

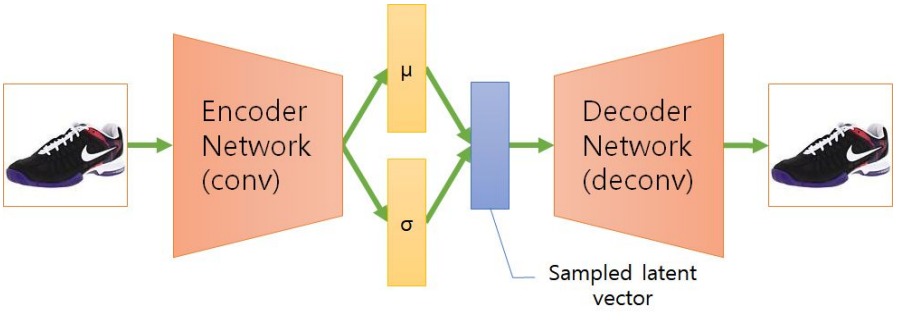

Figure 1. Variational Auto-encoder

It is hard to immediately pinpoint the ideal probability distribution immediately [3]. Hence, VAE uses variational inference for optimization to find the distribution of potential variables that successfully describe the input data and reconstruct the input through the potential variables. Variational inference is a method that first assumes a typical distribution like Gaussian and modifies the variable of distribution to approach the ideal probability distribution.

3. Network realization

The paper “Auto-Encoding Variational Bayes” constructs a fully connected network for encoder and decoder networks. In contrast, our model uses CNN network for the encoder and decoder networks, which greatly reduces number of learning parameters and training time. Figure 2 shows the code of our encoder and decoder networks. Encoder (convolutional) layer encodes input image to a simplified probability distribution. Depth of five resulted in a recommender system that functions reasonably. Decoder (deconvolutional) network is symmetric to the encoder network. Symmetricity allows optimized recreation of input image. The kernel size varies slowly over each layer so that data is not lossy when de-sampling in the decoder network. Each latent vector has size 100. After training, the top-five similar shoes are calculated from the latent vector.

```
self.encoder = nn.Sequential(
    nn.Conv2d(image_channels, 6, kernel_size=4, stride=2),
    nn.ReLU(),
    nn.Conv2d(6, 12, kernel_size=3, stride=2),
    nn.ReLU(),
    nn.Conv2d(12, 24, kernel_size=3, stride=2),
    nn.ReLU(),
    nn.Conv2d(24, 30, kernel_size=4, stride=1),
    nn.ReLU(),
    nn.Conv2d(30, 40, kernel_size=3, stride=2),
    nn.ReLU(),
    Flatten()
)
```

```
self.decoder = nn.Sequential(
    UnFlatten(),
    nn.ConvTranspose2d(40, 30, kernel_size=3, stride=2),
    nn.ReLU(),
    nn.ConvTranspose2d(30, 24, kernel_size=4, stride=1),
    nn.ReLU(),
    nn.ConvTranspose2d(24, 12, kernel_size=3, stride=2),
    nn.ReLU(),
    nn.ConvTranspose2d(12, 6, kernel_size=3, stride=2),
    nn.ReLU(),
    nn.ConvTranspose2d(6, image_channels, kernel_size=4, stride=2),
    nn.Sigmoid(),
)
```

Figure 2. Encoder (left) and decoder (right) network

III. Discussion

1. Qualitative results

(a) Original image

(b) Reconstructed image

Figure 3. Reconstructed image after

One may observe that after 1000 epochs, hue, saturation, and colors are reconstructed quite accurately while texture and detailed styles are not reproduced perfectly as can be seen in figure 3. This may be due to the simplicity of the network with only depth of five. Figure 4 shows the shoe-shaped image data constructed when random noise ε is put as input data in the trained network.

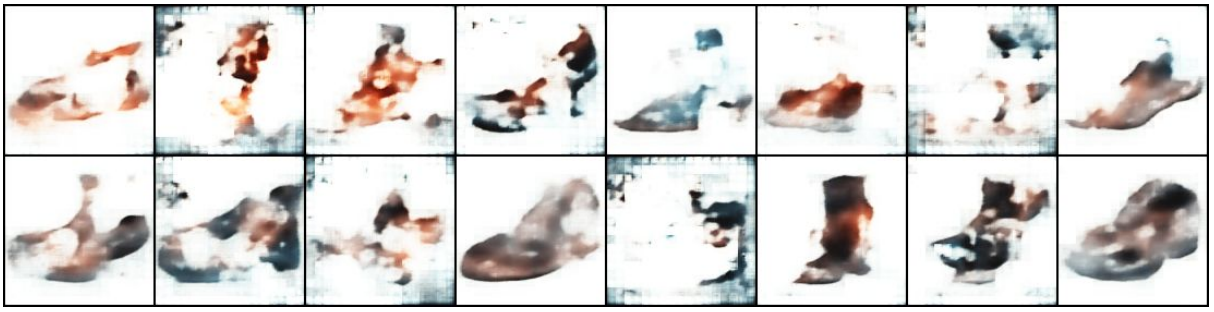

Figure 4. Image reconstructed from random noise

The recommender system suggests five similar products from the 5k input data, as shown in figure 5. Colab Notebook “final_project/final_project(1000 epoch).ipynb” takes an integer input from 0 to 4999 and displays five products of similar style.

Figure 5. Top five recommendations

2. Quantitative results

Two losses, binary cross entropy and Kullback-Leibler divergence, are tracked during network training. Binary cross entropy is related to reconstruction accuracy and Kullback-Leibler divergence is related to the normal distribution of latent vector and the difference between input and reconstructed data. The sum of loss successfully drops over 1000 epochs despite the increase in loss of Kullback-Leibler divergence loss because the binary cross entropy loss is weighted much heavier than the Kullback-Leibler divergence loss.

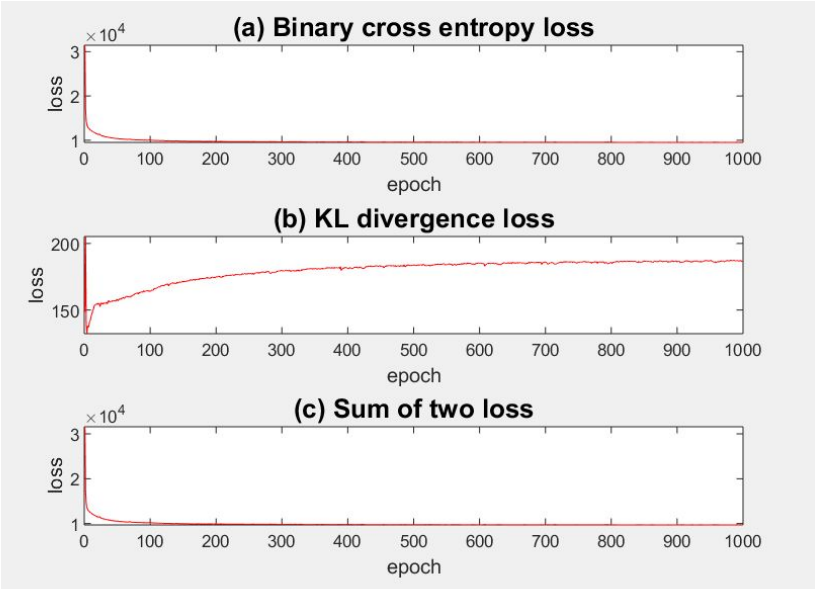

Figure 6. Loss over epoch

When the weights of the two losses are made similar by multiplying the Kullback-Leibler loss by a factor of 100, reconstructions whether from input image or random noise are undistinguishable, which makes it unfit for our recommender system, as can be seen in the “final_project_KLstrong” directory of our github repository.

IV. Conclusion & Future work

We successfully built a VAE model with CNN network as encoder and decoder networks that recommends products of similar style. Future work would include building a website that provides a convenient UI for users.

V. Contribution

Ji Yea Shim : Idea development, Data preprocessing, tuning VAE network, hyper-parameter tuning, Recommendation system implementation, PPT, Report, Presentation
Kanghoon Lee : Idea development, VAE network implementation, tuning VAE network, Training network, Report, Presentation
SungWon Yun : VAE network implementation, Recommendation system implementation, Report, Presentation

Reference

1. **https://github.com/leehoon7/CS470_project.git**
2. http://vision.cs.utexas.edu/projects/finegrained/utzap50k/
3. Kim, D., Yang, H., Chung, M., Cho, S., Kim, H., Kim, M., . . . Kim, E. (2018). Squeezed convolutional variational autoencoder for unsupervised anomaly detection in edge device industrial internet of things. In 2018 international conference on information and computer technologies (icict) (pp. 67–71). IEEE.
4. Kingma, Diederik P and Welling, Max. Auto-Encoding Variational Bayes. In The 2nd International Conference on Learning Representations (ICLR), 2013.