NYU-MMVC-LAB Weekly Report

Jing ZHU

April 24, 2017

1 Motivation

Since the efficient and effective backpropagation algorithm was proposed, neural network becomes much more popular for machine learning problems, especially for classification problems.

We aim to learn an effective cross-domain non-linear distance metric for depth image-based 3D model retrieval. By minimizing the within-class distances and maximizing the between-class distances for the outputs of the metric networks, we develop a novel pairwise metric learning network to learn a non-linear transformation to retrieve 3D models based on depth image queries.

In this week, we analyze the derivation of back propagation and conduct experiments on the NYU Depth V2 dataset and the SHREC 2014 benchmark.

2 Methods

In general, neural networks are trained only based on within-class distance. In our past research work, we have tried to train a pairwise neural network for the depth image-based 3D shape retrieval, however, we did not get a really good performance by using two 3-layer neural network. Therefore, it would be more reliable and desirable that between-class distance is taken into consideration.

As shown in Figure 1, two 3-layer neural networks are exploited for our task. One takes extraced ScSPM features from depth image as input and the other takes extracted 3D SIFT features from 3D models as input. We define the error function based on the outputs from both 3D neural network and depth neural network and apply popular backpropagation by minimizing the value of error function. Detailed derivation process will be given in Implementation section.

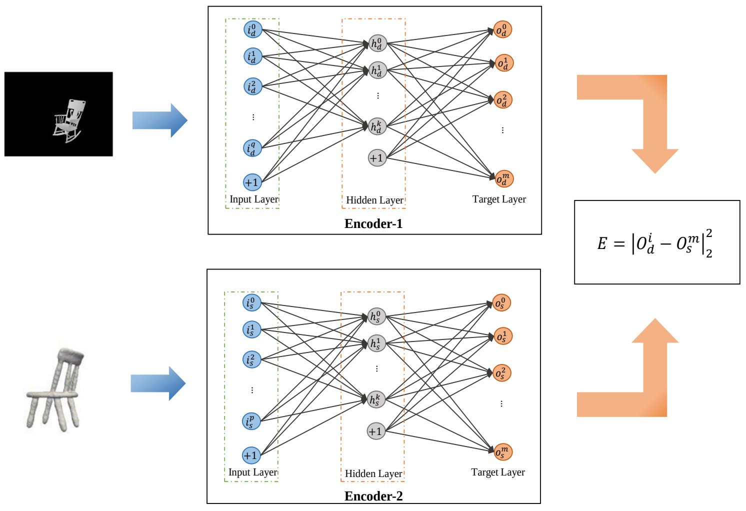

Figure 1: The framework of our proposed pairwise metric learning network

3 Implementation

We consider a 3-layer neural network, which has an input layer, a hidden layer and a target layer. We adopt sigmoid function $f(z)$ as activation function for each neuron in our neural network:

$$f(z) = \frac{1}{1+exp(-z)}. \tag{1}$$

The activations of layer k from the pairwise neural network are denoted as a_d^k and a_s^k, respectively:

$$\begin{aligned} a_d^{k+1} &= f(W_d^k a_d^k + b_d^k) = f(r_d^{k+1}) \\ a_s^{k+1} &= f(W_s^k a_s^k + b_s^k) = f(r_s^{k+1}) \end{aligned} \tag{2}$$

where $r_d^{k+1} = W_d^k a_d^k + b_d^k$ and $r_s^{k+1} = W_s^k a_s^k + b_s^k$. Let $\boldsymbol{O}_d^i = \{\boldsymbol{o}_d^0, \boldsymbol{o}_d^1, \boldsymbol{o}_d^2, \cdots, \boldsymbol{o}_d^{M_d}\} \in \mathbf{R}^{N_d \times M_d}$ and $\boldsymbol{O}_s^i = \{\boldsymbol{o}_s^0, \boldsymbol{o}_s^1, \boldsymbol{o}_s^2, \cdots, \boldsymbol{o}_s^{M_s}\} \in \mathbf{R}^{N_s \times M_s}$ be output values at the K^{th} layer (in our case, $K = 3$) based on the depth ($\boldsymbol{I}_d^i$) and 3D model domain inputs ($\boldsymbol{I}_s^i$) respectively. Then,

$$O_d^i = a_d^K \quad O_s^i = a_s^K. \tag{3}$$

A unified objective function for learning the pairwise neural networks can be formulated as:

$$E(W_d, b_d, W_s, b_s) = \underset{\boldsymbol{W_d}, \boldsymbol{b_d}, \boldsymbol{W_s}, \boldsymbol{b_s}}{\arg\min} \frac{1}{N_1} \sum_{i=1}^{M_d} \sum_{j=1, C(j)=C(i)}^{M_s} \frac{1}{2} \|\boldsymbol{o_d^i} - \boldsymbol{o_s^j}\|_2^2 - \frac{1}{N_2} \sum_{i=1}^{M_d} \sum_{j=1, C(j)\neq C(i)}^{M_s} \frac{1}{2} \|\boldsymbol{o_d^i} - \boldsymbol{o_s^j}\|_2^2$$

$$+\frac{1}{2}\lambda \sum_{l=1}^{2} (\|\boldsymbol{W_d^l}\|_F^2 + \|\boldsymbol{W_s^l}\|_F^2), \tag{4}$$

We adopt the back-propagation algorithm to optimize each term in the objective function Eq.4. The derivative of the objective function E with respect to W_d^k, b_d^k, W_s^k and b_s^k are denoted as $\frac{\partial E}{\partial W_d^k}$, $\frac{\partial E}{\partial b_d^k}$, $\frac{\partial E}{\partial W_s^k}$, $\frac{\partial E}{\partial b_s^k}$. Taking $\frac{\partial E}{\partial W_d^k}$ and $\frac{\partial E}{\partial W_s^k}$ as an example, $\frac{\partial E}{\partial b_d^k}$ and $\frac{\partial E}{\partial b_s^k}$ could be derived similarly.Let $E_1 = \frac{1}{2}\|\boldsymbol{o_d^i} - \boldsymbol{o_s^j}\|_2^2$. Then,

$$\frac{\partial E}{\partial W_d^k} = \frac{1}{N_1} \sum_{i=1}^{M_d} \sum_{j=1, C(j)=C(i)}^{M_s} \frac{\partial E_1}{\partial W_d^k} - \frac{1}{N_2} \sum_{i=1}^{M_d} \sum_{j=1, C(j)=C(i)}^{M_s} \frac{\partial E_1}{\partial W_d^k} + \lambda W_d^k \tag{5}$$

$$\frac{\partial E}{\partial W_s^k} = \frac{1}{N_1} \sum_{i=1}^{M_d} \sum_{j=1, C(j)=C(i)}^{M_s} \frac{\partial E_1}{\partial W_s^k} - \frac{1}{N_2} \sum_{i=1}^{M_d} \sum_{j=1, C(j)=C(i)}^{M_s} \frac{\partial E_1}{\partial W_s^k} + \lambda W_s^k. \tag{6}$$

$\frac{\partial E_1}{\partial W_d^k}$ and $\frac{\partial E_1}{\partial W_s^k}$ can be rewritten as:

$$\frac{\partial E_1}{\partial W_d^k} = \frac{\partial E_1}{\partial r_d^{k+1}} \frac{\partial r_d^{k+1}}{\partial W_d^k} = \frac{\partial E_1}{\partial r_d^{k+1}} (a_d^k)^T, \tag{7}$$

$$\frac{\partial E_1}{\partial W_s^k} = \frac{\partial E_1}{\partial r_s^{k+1}} \frac{\partial r_s^{k+1}}{\partial W_s^k} = \frac{\partial E_1}{\partial r_s^{k+1}} (a_s^k)^T. \tag{8}$$

For $k = K - 1$, $\frac{\partial E_1}{\partial r_d^{k+1}}$ and $\frac{\partial E_1}{\partial r_s^{k+1}}$ can be represented as:

$$\begin{aligned} \frac{\partial E_1}{\partial r_d^{k+1}} &= (a_d^K - a_d^K) \bullet f'(r_d^K) \\ \frac{\partial E_1}{\partial r_s^{k+1}} &= (a_d^K - a_s^K) \bullet f'(r_d^K), \end{aligned} \tag{9}$$

where $f'(r_d^K)$ is the derivative of the activation function in the output layer and $\bullet$ denotes the element-wise multiplication. For layer $k = 1, 2, \cdots, K-3, K-2$, with the backpropagation algorithm, $\frac{\partial E_1}{\partial r_d^{k+1}}$ and $\frac{\partial E_1}{\partial r_s^{k+1}}$ can be obtained as:

$$\begin{aligned}\frac{\partial E_1}{\partial r_d^{k+1}} &= \frac{\partial E_1}{\partial r_d^{k+2}}\frac{\partial r_d^{k+2}}{\partial r_d^{k+1}} = (W_d^{k+1})^T\frac{\partial E_1}{\partial r_d^{k+2}} \bullet f^{'}(r_d^{k+1}) \\ \frac{\partial E_1}{\partial r_s^{k+1}} &= \frac{\partial E_1}{\partial r_s^{k+2}}\frac{\partial r_s^{k+2}}{\partial r_s^{k+1}} = (W_s^{k+1})^T\frac{\partial E_1}{\partial r_s^{k+2}} \bullet f^{'}(r_s^{k+1})\end{aligned} \tag{10}$$

Similarly, $\frac{\partial E}{\partial b_d^k}$ and $\frac{\partial E}{\partial b_s^k}$ can be calculated as:

$$\begin{aligned}\frac{\partial E}{\partial b_d^k} &= \frac{\partial E_1}{\partial r_d^{k+1}}\frac{\partial r_d^{k+1}}{\partial b_d^k} = \frac{\partial E_1}{\partial r_d^{k+1}} \\ \frac{\partial E}{\partial b_s^k} &= \frac{\partial E_1}{\partial r_s^{k+1}}\frac{\partial r_s^{k+1}}{\partial b_s^k} = \frac{\partial E_1}{\partial r_s^{k+1}}\end{aligned} \tag{11}$$

By substituting Eq.(7) and (8) into Eq.(5) and (6), we can obtain the derivative of the objective function E with respective to W_d^k and W_s^k. Similarly, with Eq.(11), we can get the derivative of the objective function E with respective to b_k^s and b_k^t. Then W_d^k, W_s^k, b_k^s and b_k^t can be updated with the gradient descent algorithm (as shown in Algorithm 1).

Algorithm 1 Training algorithm of proposed pairwise neural network

Input: depth image training set S and 3D model dataset D; layer size K of the neural network; regularization term λ ; learning rate β.
Output: W_d, W_s, b_d and b_s

```
Do
    a. Compute the outputs of two neural networks with forward propagation for all the
       training pair-wise example $I_d^i$ and $I_s^i$;
    b. For $k = K-1, K-2, ...1$ Compute $\frac{\partial E}{\partial W_d^k}$, $\frac{\partial E}{\partial b_d^k}$, $\frac{\partial E}{\partial W_s^k}$ and $\frac{\partial E}{\partial b_s^k}$ with Eq.(5) and
       Eq. (6);
    c. Update $W_d^k$, $b_d^k$, $W_s^k$ and $b_s^k$ for $k = 1, 2, ..., K-1$:
       $W_d^k = W_d^k - \beta\frac{\partial E}{\partial W_d^k}$,    $b_d^k = b_d^k - \beta\frac{\partial E}{\partial b_d^k}$
       $W_s^k = W_s^k - \beta\frac{\partial E}{\partial W_s^k}$,    $b_s^k = b_s^k - \beta\frac{\partial E}{\partial b_s^k}$
Until convergence
```

Table 1: Number of samples of each category in 3D model datasets and depth image dataset

	bathtub	bed	chair	desk	dresser	monitor	stand	sofa	table	toilet
3D model	109	467	712	204	203	433	218	602	402	324
Depth image	57	318	654	197	111	118	148	307	539	68

Table 2: Performance metrics comparison of depth image-based 3D model retrieval on the NYU Depth V2 dataset and the SHREC 2014 benchmark.

	NN	FT	ST	DCG	E	AP
10 categories						
NT	0.10	0.11	0.23	0.67	0.01	0.13
CM	0.11	0.13	0.25	0.67	0.02	0.13
SCM	0.12	0.13	0.25	0.67	0.02	0.13
PNN	0.13	0.14	0.27	0.68	0.02	0.14
Metric Learning	**0.89**	**0.76**	**0.79**	**0.92**	**0.10**	**0.78**

4 Experimental Results

The NYU Depth V2 dataset contains frames of video sequences in a variety of indoor scenes, from where objects in depth images are extracted as queries in our experiments. The database is constructed by selecting 3D models in correponding categories from the large-scale extended SHREC 2014 benchmark. The number of samples in 10 categories (*bathtub, bed, chair, desk, dresser, monitor, stand, sofa, table, toilet*) of both datasets are given in Table 1.
We evaluate the proposed approach using 6 common evaluation metrics, Nearest neighbor (NN), First Tier (FT), Second Tire (ST), E-Measure (E), Discounted Cumulated Gain (DCG) and Average Precision (AP). We conduct experiments using all 10 categories, and compare with the state-of-art transfer learning approaches CM, SCM and PNN. We also compare with the non-transfer (NT) approach, which directly utilizes the original depth image and 3D model representations for retrieval. Results are reported in Table 2.

5 Conclusion

As we can see from performance metrics, our new approach achieves better performance on the NYU Depth Dataset V2 and the extended SHREC 2014 3D shape retrieval benchmark.