3D Person Re-Identification

Pratik Gujjar, Geoff Nagy, Payam Nikdel
Faraz Shamshirdar and Rakesh Shrestha
Department of Computing Science
Simon Fraser University
{pgujjar, gnagy, pnikdel, fshamshi, rakeshs}@sfu.ca

Abstract

In this report, we describe and compare several deep-learning approaches for identifying people using 3D point cloud data. We also describe various enhancements that we have made to these networks in order to increase their classification accuracy. Our experiments show that we are able to determine individual identities with high accuracy.

1 Introduction

We chose to investigate the use of deep learning networks to identify human subjects scanned by a Microsoft Kinect sensor. Person identification has numerous applications in robotics (identifying an owner) and surveillance (distinguishing allies from enemies). Security applications (such as identifying criminals from security footage) are of particular interest, and depth-based identification will work even if an individual covers their face. 3D sensors are affordable and can be used almost anywhere. The success of deep learning suggests that such networks would be adept at distinguishing between different individuals scanned by a 3D sensor. We begin by discussing previous work, and describe the dataset used in our experiments. In the remaining sections, we discuss the networks we chose to employ, the modifications we made to them, and the results we achieved. Our report ends with concluding remarks.

2 Previous Work

Approaches for human reidentification generally involve the use of colour information in RGB images [1] [2]. With the availability of inexpensive RGB-D cameras, the use of full 3D information for human reidentification has also been explored. Previous work on human reidentification using 3D point clouds have mainly relied on anthropometrics and biometrics [3] [4] to model 3D human skeletal structures, and use hand-crafted features like arm length or torso width for identification. In contrast, we aim to learn these features using Convolutional Neural Networks, to distinguish one person from other by using their point cloud representations only.

3 Dataset and Preprocessing

To evaluate the effectiveness of our networks, we trained and validated them on the Berkeley MHAD dataset [5]. This is a database of 12 human subjects performing 11 different actions, with five repetitions each, captured from two differently-positioned Kinect sensors. Actions included throwing, sitting down in a chair, and jumping in place. We augment our data by generating mirrored representations of each image. This results in 300,000 images.

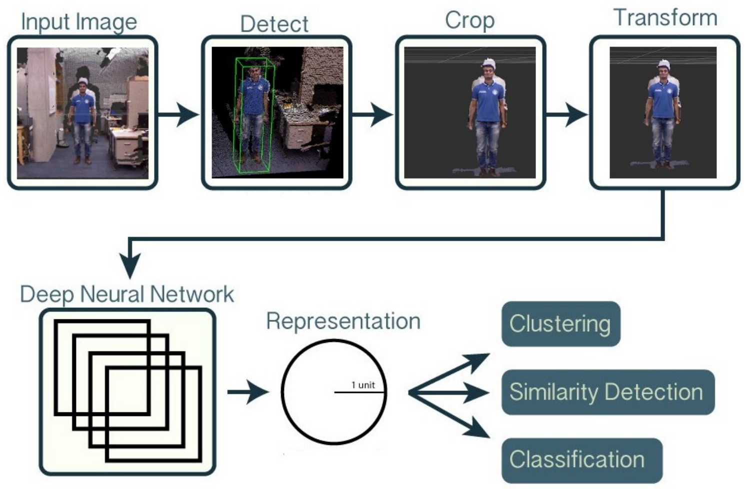

Figure 1: Dataflow Pipeline

3.1 3D reconstruction from RGB-D Data and Person Detection

We used the pinhole-camera model [6] to generate RGB point clouds from RGB-D images containing 12 subjects from the MHAD dataset. The coordinates of 3D point $\mathbf{X_c} = [X_c, Y_c, Z_c]$ can be computed from undistorted 2D image coordinates $[u, v]$ and depth d using the following:

$$X_c = \frac{d(u - c_u)}{f_u} \qquad Y_c = \frac{d(v - c_v)}{f_v} \qquad Z_c = d \tag{1}$$

where $[f_u, f_v]$ and $[c_u, c_v]$ are the focal length and principal point of the RGB camera, respectively (provided in the MHAD dataset).

We used the built-in person detection module from the Point Cloud Library (PCL) to extract human subjects from RGB-D images. The detection module is based on that of [7] and [8]. The module can detect people standing or walking on the ground plane in real time and is based on a pretrained SVM classifier. The detected person is cropped and translated by the bottom-most part of the person so that our method is invariant to the position of the person in the scene. The output of the detection module is a 3D point cloud. Figure 1 shows our overall dataflow pipeline (3D reconstruction part skipped for clarity).

4 Our Approaches

4.1 VoxNet

Maturana et al., [9] in their paper, propose to fully utilize information in a 3D representation of an object by integrating a volumetric occupancy grid representation with a supervised 3D convolutional network *VoxNet*. VoxNet is trained with LiDAR point clouds, RGB-D point clouds and CAD models for real-time object recognition. Our work, concerned with using RGB-D point clouds to identify a person, is similar to VoxNet's goal of exploiting spatial information in a volume.

Preprocessed point clouds for 12 people are used to train the network with a batch size of 32 for 80 epochs and 64 iterations per epoch. The network is trained with an intention to develop a feature extractor resulting in an array of 128 features. This would be achieved by removing the last fully connected layer from VoxNet. Differences between VoxNet and our implementation can be summarized as follows:

1. VoxNet trains the network with 12 rotations of each object to render the network invariant to orientation. We choose 4 rotations, two originally obtained from two Kinects used in the MHAD datset and two more as mirrored point clouds of these.

2. We keep to the occupancy grid size of 32x32x32 as used in the network. Our occupancy grids are deterministic maps of whether a voxel in the pointcloud is occupied. Also to retain aspect ratio of the shape, the occupancy grid is obtained by uniformly scaling down downsampled point clouds across all x, y and z dimensions. Scaling down is done by a factor of 2.3, the maximum height of our bounding box for extracting a person from a point cloud.
3. Occupancy grids themselves are generated from CAD models in VoxNet. We however use the point cloud itself. Although this results in a substantially sparse representation, we observe that information like height and torso width of a person can be distinguished.

We trained our adaptation of VoxNet with 2000 occupancy grids per person, 500 for every rotation and for 12 different people. Training error reported as a metric was observed to be inconclusive as it prominently took on values 0 and 1. On the labels determined from the test results, predicted labels for all test samples were the same, and always equal to one of the 12 labels for all 12 people. The predictions can be inferred to be incorrect for all of our test cases. This non-intuitive result can be attributed primarily to incomplete representation of the shape of a person by our occupancy grids. Person point clouds deliver only a surface of information and any rotation thereof would be increasingly sparse. We estimate that generating binary occupancy grids as whether or not a voxel is occupied, bestows undue emphasis on noisy voxels as well. Noise is also present due to imperfect person extraction.

While we originally attributed this lack of accuracy to the sparsity of our input (i.e., the low-resolution occupancy grid), the success of another occupancy-grid-based approach (4.2) convinced us that modifying the VoxNet structure might help us obtain better results. After changing all “leaky” Rectified Linear Unit (ReLU) layers to regular ReLU layers, and changing all volumetric drop-out layers to regular drop-out layers in VoxNet, we achieved 68.4% validation accuracy after only 30 epochs.

4.2 Volumetric Convolutional Neural Network

The Volumetric Convolutional Neural Network (VCNN) developed by [10] performs 3D object classification with an accuracy of approximately 88%. It is a 35-layer network consisting of several *blocks*, where each block contains a volumetric convolution layer (to encourage scale- and shift-invariant processing), a batch normalization layer, and a ReLU layer. Nine blocks are used, each containing increasingly smaller volumetric kernel sizes. Each group of three blocks is followed by a drop-out layer to prevent overfitting. The network ends with a view layer for examining the learned features, a linear layer to scale down the outputs from the view layer, an additional ReLU layer, and a drop-out layer whose outputs are mapped to the final network linear output layer. Figure 2 shows the high-level structure of this network.

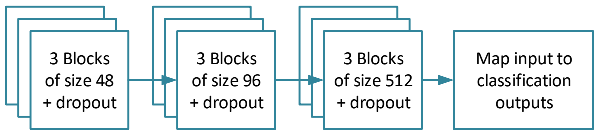
3 Blocks of size 48 + dropout
3 Blocks of size 96 + dropout
3 Blocks of size 512 + dropout
Map input to classification outputs

Figure 2: A high-level depiction of the structure of the VCNN.

4.2.1 Modifying Meta-Parameters

This VCNN was intended to classify common 3D objects. Our main concern was that person identification would be much harder since people do not generally vary in shape or size as much as, for example, household furniture. To address this, we tested our network with two sizes of occupancy grids (30x30x30, 35x35x35) to examine any differences in accuracy. Memory limitations prevented us from using higher resolutions.

Due to memory and time considerations, the VCNN was only evaluated using a subset of the MHAD dataset (36,000 images), with two-thirds of this used for training and the other third used for validation. We ran 30 training epochs of 128 batches each, with each batch containing 128 images.

Resolution	Accuracy
30x30x30	86.17%
35x35x35	88.33%

Table 1: Classification accuracy of the VCNN with varying occupancy grid resolutions.

Learning Rate	Accuracy
0.1	89.39%
0.25	88.24%
0.5	85.22%

Table 2: Network classification accuracy with three different learning rates. A learning rate of 0.1 yielded the best performance.

We chose 30 epochs as our limit after preliminary experiments indicated little to no change in our testing accuracy after that point.

There was only a minor difference in validation accuracy between the two resolution levels, as shown in Table 1. In fact, classification accuracy was similar to that of the author's own experiments (88%, [10]). We believe that the network was able to learn features and actions unique to each individual, such as how subject *A* throws a ball or sits down in a chair, compared to another subject *B*.

Since the VCNN performed well, our next goal was to determine if we could train the network faster. We increased the learning rate from 0.01 (the value used by [10]) to 0.1. After 30 epochs, our network achieved a classification accuracy of 89%. This slight improvement suggests that 30 epochs might not have been enough for the network to converge in our original experiments, and that additional training epochs may be beneficial. To explore this, we evaluated the performance of our network using two additional learning rates: 0.25 and 0.5. Our results are shown in Table 2. Learning rates of 0.25 and 0.5 resulted in large enough step sizes during SGD optimization that the network performed more poorly than it did with a learning rate of 0.1.

4.2.2 Modifying Network Structure

The original VCNN contained several convolutional layers. In an attempt to increase the accuracy of the network, we increased the kernel sizes of the first two convolutional layers: the first from 6x6x6 to 8x8x8, and the second from 5x5x5 to 6x6x6. Using a learning rate of 0.1, the network achieved approximately 89% classification accuracy, which is not a significant improvement.

For the second modification, we removed the last ReLU and drop-out layers in the network, using the modified kernel sizes described above. Again, this resulted in no significant change, and the network still performed with approximately 89% accuracy. This seemed to indicate that the *blocks* of layers in the VCNN are the main contributing factors to the accuracy of the network.

To explore this further, the network was modified to contain only 3 blocks of size 48, with the first having a kernel size of 12 and a convolution step value of 6. The other two blocks had kernel sizes of 1 and convolution steps of 1. These values kept the number of parameters reasonable (3072) so that the network could learn effectively given the number of training samples. With these modifications, the network achieved approximately 87% accuracy on our validation set. The high classification accuracy in this network configuration suggests that the original VCNN structure may be needlessly complex for identifying humans, and that high performance can be achieved with a simpler network. This is substantiated by the moderate success of the modified VoxNet network (Section 4.1), which performed reasonably well despite its relative simplicity.

4.3 Inception-based CNN models on Spatially Encoded 2D Images

The motivation behind this model was that high-resolution occupancy grids are prohibitively expensive, and low-resolution (but computationally feasible) grids result in significant loss of information. In the following subsection, we outline our approach of representing data in a way that maintains an image's spatial information at the original resolution.

4.3.1 Data Representation

For a denser representation, we exploit the fact every pixel in an image maps to a point $[X_c, Y_c, Z_c]$ in 3D space. This is done by reprojecting the 3D points into image coordinates $[u, v]$ and encoding the 3D coordinates in RGB channels of the image. This representation maintains the spatial information in absolute scale in all three directions while requiring less memory than an occupancy grid (96x96 versus 32x32x32, for example). An example of a spatially encoded image is depicted in Figure 3.

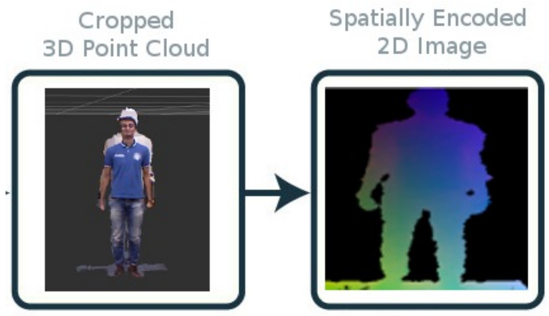

Figure 3: Spatially Encoded 2D Image from point cloud for Inception-based CNN models

A similar approach was followed by [11] in which the authors classify objects based on colorized depth images. This method does not represent absolute scale in the X and Y axes (considering depth as the Z-axis), while our representation does.

4.3.2 Inception-based CNN models

Standard Inception-based CNN models can perform high accuracy image classification, but do not trivially generalize to distinguishing between new samples. We would want to train a CNN on an existing dataset and use the network to do classification on different test subjects. For this, the output of the network should be a lower dimensional feature vector which can be used to distinguish between people that were not in the dataset.

For this, we use a FaceNet-based [12] CNN, which learns a 128-dimensional vector $f(x)$ from a spatially-encoded image x. This vector represents a feature space which can be used to discriminate between identities based on Euclidean distance, and is constrained to unit hypersphere of 1 i.e. $||f(x)||_2 = 1$. The loss of the network is implemented as triplet loss. For this, an image x_i^a is chosen as an *anchor*, and another image of the same person x^p is known as the *positive* image. Another image x^n of a different person is chosen as *negative* image. The *anchor*, *positive* and *negative* image form a triplet. Triplet loss is fomulated as:

$$L = \sum_{i}^{N} \left[||f(x_i^a) - f(x_i^p)||_2^2 - ||f(x_i^a) - f(x_i^n)||_2^2 + \alpha \right]_+ \tag{2}$$

where α is threshold that all triplet should satisfy (0.2 in our case), $||.||_2^2$ is squared Euclidean norm and $[z]_+ = max\{0, z\}$.

In each mini-batch, we sample at most P images per person from Q people in the dataset and send all $M \approx PQ$ images through the network in a single forward pass. We currently use $P = 5$ and $Q = 40$. We take all valid triplets and backpropagate the gradient of the triplet loss for training the network. This method was also used in the OpenFace [13] project.

4.3.3 Network architecture

We use a modified version of the FaceNet [12] nn4 network. It is based on the GoogLeNet [14] architecture. We used a method similar to that of OpenFace [13] to reduce the number of parameters.

Table 3: Details of the modified NN4 Inception based architecture

type	output size	#1x1	#3x3 reduce	#3x3	#5x5 reduce	#5x5
conv1 (7x7x3,2)	48x48x64					
max pool+norm	24x24x64					
inception (2)	24x24x192		64	192		
norm+max pool	12x12x192					
inception (3a)	12x12x256	64	96	128	16	32
inception (3b)	12x12x320	64	96	128	32	64
inception (3c)	6x6x640		128	256,2	32	64,2
inception (4e)	3x3x1024		160	256,2	64	128,2
inception (5a)	3x3x736	256	96	384		
inception (5b)	3x3x736	256	96	384		
avg pool	736					
linear	128					
ℓ_2 normalization	128					

4.3.4 Training on Berekely MHAD Dataset and Live Data Stream

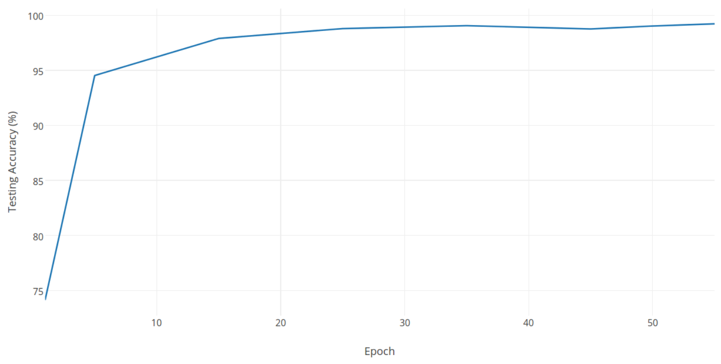

Figure 4: Plot of testing accuracy aganist the number of epochs.

We trained different versions of the modified FaceNet network with all 300,000 images from the MHAD dataset. The architecture with the highest performance is shown in Table 3. Since the size of our training data is much smaller than FaceNet's nn4 and OpenFace's nn4.small, we reduced the number of parameters in the network by removing some layers. We used OpenFace's implementation of FaceNet to train our data. Figure 4 shows test accuracy of the neural network against the number of epochs.

We experimented on live RGB-D data streams from the Kinect sensor for qualitative analysis of our model. We used the open-source Robot Operating System (ROS) framework [15] for integrating different parts of our project. Using nearest-neighbour classification, we were able to successfully classify 3 test subjects (that were not present in the dataset) against each other. A demonstration video is available at [1]. Note that the features are not invariant to depth and orientation. This can be expected as the training data consisted of only 12 subjects (300,000 RGB-D images) while FaceNet used a private dataset with over one million images.

5 Conclusion

In this report, we have presented our work on three deep-learning frameworks for person re-identification. We discussed our various approaches and modifications to these networks, and the results we obtained. Our work demonstrates that person identification is feasible even in the absence of facial features currently heavily relied upon. A blend of shape information and facial features however, looks very promising and is worthy of exploration.

[1] `https://youtu.be/pN6oPjqdJLg`

5.1 Contributions

Literature survey over existing person reidentification methods was done by everyone equally. The project was split into three approaches, each pursued separately. Faraz and Rakesh were vested in training and testing on spatially encoded images. They were also involved with preprocessing Berkeley MHAD point clouds. Pratik and Payam pursued VoxNet as the volumetric covolutional model. They explored effective voxelization options and interpreted person point clouds as occupancy grids. Geoff investigated VCNN as the other volumetric convolutional model, including experiments over changing activations and modifying layers from VCNN. The demo using ROS was done by Rakesh and Faraz, troubleshooting by Payam and Pratik, experiments were run with everyone's equal roles.

References

[1] B. C. Munsell, A. Temlyakov, C. Qu, and S. Wang, "Person identification using full-body motion and anthropometric biometrics from kinect videos," in *European Conference on Computer Vision*. Springer Berlin Heidelberg, 2012, pp. 91–100.

[2] D. Gray and H. Tao, "Viewpoint invariant pedestrian recognition with an ensemble of localized features," in *European conference on computer vision*. Springer Berlin Heidelberg, 2008, pp. 262–275.

[3] A. Albiol, A. Albiol, J. Oliver, and J. Mossi, "Who is who at different cameras: people re-identification using depth cameras," *IET computer vision*, vol. 6, no. 5, pp. 378–387, 2012.

[4] M. Farenzena, L. Bazzani, A. Perina, V. Murino, and M. Cristani, "Person re-identification by symmetry-driven accumulation of local features," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 2360–2367.

[5] F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy, "Berkeley mhad: A comprehensive multimodal human action database," in *Applications of Computer Vision (WACV), 2013 IEEE Workshop on*. IEEE, 2013, pp. 53–60.

[6] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.

[7] M. Munaro and E. Menegatti, "Fast rgb-d people tracking for service robots," *Autonomous Robots*, vol. 37, no. 3, pp. 227–242, 2014. [Online]. Available: http://dx.doi.org/10.1007/s10514-014-9385-0

[8] M. Munaro, F. Basso, and E. Menegatti, "Tracking people within groups with rgb-d data," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2012, pp. 2101–2107.

[9] D. Maturana and S. Scherer, "VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition," in *IROS*, 2015.

[10] C. R. Qi, H. Su, M. Niessner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view cnns for object classification on 3d data," *arXiv preprint arXiv:1604.03265*, 2016.

[11] A. Eitel, J. T. Springenberg, L. Spinello, M. A. Riedmiller, and W. Burgard, "Multimodal deep learning for robust RGB-D object recognition," *CoRR*, vol. abs/1507.06821, 2015. [Online]. Available: http://arxiv.org/abs/1507.06821

[12] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," *CoRR*, vol. abs/1503.03832, 2015. [Online]. Available: http://arxiv.org/abs/1503.03832

[13] B. Amos, B. Ludwiczuk, and M. Satyanarayanan, "Openface: A general-purpose face recognition library with mobile applications," CMU-CS-16-118, CMU School of Computer Science, Tech. Rep., 2016.

[14] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *CoRR*, vol. abs/1409.4842, 2014. [Online]. Available: http://arxiv.org/abs/1409.4842

[15] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.