Computers & Graphics (2021)

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

HCMUS-3 at SHREC 2021: Retrieval of Cultural Heritage Objects

Minh-Khoi Pham, Dinh-Huan Nguyen[a]

[a] University of Science, VNU-HCM

ARTICLE INFO

Article history:

ABSTRACT

In the Retrieval of Cultural Heritage Objects (RCHO) track of the 3D Shape Retrieval Challenge (SHREC) 2021, we, HCMUS-3, propose a supervised method using the original 3D data in the mesh format. We utilize two state-of-the-art models for feature extraction of the 3D objects and transform into embedding vectors for the retrieval process.

1. Overview

Since the collection is fully labeled, we consider the retrieval by category challenge as a multi-class classification problem. For each 3D object, we simplify and perform feature extraction of the mesh data using state-of-the-art network architectures, specifcally the MeshWalker [1] and the MeshNet [2]. The output of this process is an embedding vector for each of the objects in those sets. The distance between 3D objects is equivalent to the distance between corresponding embedding vectors.

2. Data processing

2.1. Simplification

Originally, the mesh data for each 3D object in the given dataset is simplified to around 40,000 triangle faces by the organizer. However, it is still a large number to work with our limited resource. Therefore, in the processing stage, we continue to simplify them using the Quadric Error Metric Decimation [3]. Examples can be seen in Figure 1. We apply this to both the collection and the query set.

2.2. Random walks generation

Additionally, we generate random walks for each 3D object. A random walk is a sequence of steps, which are pairs of source and destination vertices (not necessarily adjacent), starting from a randomly chosen vertex. Each step is represented as the 3D translation between the source and the destination vertex $(\Delta x, \Delta y, \Delta z)$. In other words, R random walks on a 3D object, each of length L, is represented as a $[R, L, 3]$ tensor. A random walk is visualized in Figure 2.

3. Using MeshWalker [1]

3.1. Network architecture

MeshWalker deals directly with mesh data through intermediate representation of multiple random walks. The network consists of three components: the first component consists of fully-connected layers which changes the feature space; the second component consists of recurrent units which aggregates the information along the walk; and the third component, which is a fully-connected layer followed by a softmax layer, computes the classification score vector. Since the recurrent unit does not work well with long sequential data, gated recurrent units are used to achieve better performance with less trainable parameters.

3.2. Training process

For each 3D object, we simplify the mesh data into around 4000 faces and generate 1 walk of 100 steps.

We use the same training pipeline as in the original paper. We train the network from scratch for 3000 epochs using the Adam optimizer with an initial learning rate of 0.0003.

e-mail: pmkhoi@selab.hcmus.edu.vn (Minh-Khoi Pham)

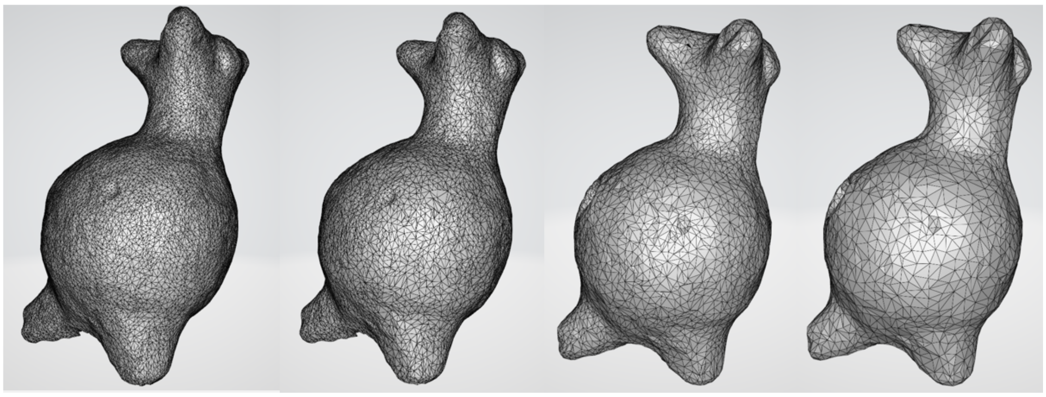

Fig. 1. Mesh simplification examples. From left to right: Original, 20k faces, 10k faces, 5k faces

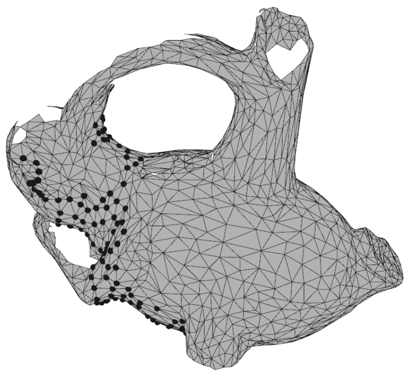

Fig. 2. A random walk on a 3D object

4. Using MeshNet [2]

4.1. Network architecture

MeshNet deals directly with mesh data without any convert in pre-processing stage. This method regards the face as the unit cell and defines a connection between two faces if they share a common edge. To capture features of faces more explicitly, face features are fed through spatial descriptor and structural descriptor. These descriptors enables us to consider the spatial distribution and local structure of shapes respectively. Features are then passed through mesh convolution blocks to aggregate neighboring information. Therefore, the network is able to capture detailed features of faces and learn the 3D shape representation well. Then a multilayer perceptron (MLP) is applied to fuse the neighboring features and output new structural feature. Finally, global features, which are ensembled by concatenating and feeding all structural features to another MLP, are forwarded to fully-connected layers to output classification score vector.

4.2. Training process

For each 3D object, we simplify the mesh data into around $20,000$ faces, and augment the data by jittering the positions of vertices by a Gaussian noise with zero mean and 0.01 standard deviation during training. We use the same training pipeline as in the original paper.

We load MeshNet with pretrained weights on the Model40 dataset [4] and train 200 epochs for each fold. We use the AdamW optimizer for training, with an initial learning rate of 0.001, scaling by 0.1 for every 50 epochs, weight decay at 0.0005.

5. Retrieval process

Using the trained model, we perform feature extraction on the objects in the collection and the query set. The features are either the classification score vector or the feature vector (the input of the classifer).

To generate the distance matrix, we use different different distance metrics. Specifically, we use the Euclidean distance for the feature vectors and the inverse dot product for the classification score vectors.

If we have multiple trained models, we ensemble the generated distance matrices through averaging them.

6. Submissions

6.1. Retrieval by Shape

We submitted 4 runs for the Retrieval by Shape task.

- Run 1: using the feature vector produced by a trained MeshWalker;
- Run 2: using the classification score vector produced by a trained MeshWalker;
- Run 3: using the classification score vector produced by a MeshNet trained on 1 fold;
- Run 4: using the classification score vector produced by an ensemble of 5 trained MeshNet.

6.2. Retrieval by Culture

We submitted 1 runs for the Retrieval by Culture task.

- Run 1: using the classification score vector produced by an ensemble of 5 trained MeshNet.

References

[1] Lahav, A, Tal, A. Meshwalker: Deep mesh understanding by random walks. arXiv preprint arXiv:200605353 2020;.

[2] Feng, Y, Feng, Y, You, H, Zhao, X, Gao, Y. Meshnet: Mesh neural network for 3d shape representation. AAAI 2019 2018;.

[3] Garland, M, Heckbert, PS. Surface simplification using quadric error metrics. In: Proceedings of the 24th annual conference on Computer graphics and interactive techniques. 1997, p. 209–216.

[4] Wu, Z, Song, S, Khosla, A, Yu, F, Zhang, L, Tang, X, et al. 3d shapenets: A deep representation for volumetric shapes. 2015. arXiv:1406.5670.