International Journal of Artificial Intelligence,
ISSN 0974-0635; Int. J. Artif. Intell.
2021, Volume 19, Issue Number 2

A Complete Content-based 3D Shape Retrieval System Using Deep Learning

Mohammed Benjelloun[1], **El Wardani Dadi**[2] **and El Mostafa Daoudi**[3]

[1]Department of Computer Science
Faculty of Engineering
University of Mons, BELGIUM
mohammed.benjelloun@umons.ac.be

[2] LSA laboratory
National School of Applied Sciences of Al-Hoceima
University of Abdelmalek Essaadi, MOROCCO
w.dadi@uae.ac.ma

[3] LaRi Laboratory
Faculty of Sciences
University of Mohammed First, MOROCCO
e.daoudi@ump.ac.ma

ABSTRACT

This work addresses the problem of content-based 3D shape retrieval. Despite the variety of approaches proposed in the literature, the challenge that still lies ahead is to design a method that allows large-scale retrieval and respects relevance. In this paper, we offer a complete system that takes full advantage of deep learning. Using Convolutional Neural Network (CNN) for shape indexing, our key idea is to consider the final output (softmax layer) of a well-trained model as a given 3D object's descriptor. This solution is employed not only to improve relevance but also to accelerate the shape matching process. In addition to serving as the search key, the considered descriptor is used to predict the models' list to match. Hence, this will avoid a sequential and systematic comparison with all the 3D models in the database. To evaluate our approach with state-of-the-art methods, we use the SHREC'17 event specifications and ShapeNetCore subset data. We demonstrate that the proposed shape indexing technique improves the relevance and accelerates the retrieval process significantly.

Keywords: CBR, 3D models, 3D shape retrieval, deep learning, shape indexing, CNN.

Computing Classification System (CCS): Information systems–Information retrieval–Document representation, Computing methodologies–Computer graphics–Shape modeling–Shape analysis, Artificial intelligence–Computer vision–Computer vision representations–Shape representations.

1 Introduction

Creating, using, and storing three-dimensional models is facilitated by technological advancements such as the ramp-up of depth sensors and the rapid improvements in 3D modeling and acquisition tools. Many 3D models are currently available and accessible on the Web, making larger and larger databases. However, when the dataset's size becomes vast, it is not easy to organize, manage, and navigate it.

This paper is interested in developing a content-based 3D shape retrieval system that consists of retrieving similar 3D models to a given 3D shape query. This topic on which the scientific community work for more than a decade remains active and challenging. Despite the variety of proposed approaches in the literature (Tangelder and Veltkamp, 2004; Lian, Godil, Sun and Xiao, 2013; Osada, Funkhouser, Chazelle and Dobkin, 2002; Savva, Yu, Su, Kanezaki, Furuya, Ohbuchi, Zhou, Yu, Bai, Bai, Aono, Tatsuma, Thermos, Axenopoulos, Papadopoulos, Daras, Deng, Lian, Li, Johan, Lu and Mk, 2017; Su, Maji, Kalogerakis and Learned-Miller, 2015; Veltkamp, Giezeman, Bast, Baumbach, Furuya, Giesen, Godil, Lian, Ohbuchi and Saleem, 2010), the challenge that still lies ahead is to design a method that allows large-scale retrieval and respects relevance.

The recent astonishing results of deep learning (LeCun, Bengio and Hinton, 2015; Savva et al., 2017; Su et al., 2015), particularly the convolutional neural network (CNN), have yielded a significant impact on the field of 3D shape analysis (including recognition and retrieval). This machine learning algorithm has proven to be very efficient in the domain of content-based 3D shape retrieval. Indeed, all recent works presented in (Savva et al., 2017) adopted the CNN approach to shape indexing and have shown greater accuracy.

In this work, we propose a new approach that takes full advantage of deep learning to satisfy both requirements, i.e., relevance and speed-up. Our solution's fundamental idea is an alternative CNN-based approach of shape indexing that considers the final output (softmax layer) of a well-trained model as the descriptor. In contrast, almost view-based shape indexing methods in literature (Savva et al., 2017; Su et al., 2015) describe a 3D shape using the first part output of the CNN architecture.

The significant interest that represents our approach compared to the state-of-the-art methods is that it is employed to improve the relevance and to accelerate the shape matching process. In addition to serving as a key in the process matching, the considered descriptor permits predicting a reduced list of 3D models to match a given query. Thus, our approach avoids a sequential and systematic comparison with all the 3D models in the database.

This paper's remainder organized as follows: In Section 2, we summarize related works. Section 3 presents how deep learning is applied to shape indexing. Our proposed system is presented in Section 4. Section 5 describes related experiments and presents the results. Finally, we conclude the paper in Section 6.

2 Related Works

Content-based 3D shape retrieval is the process of retrieving similar 3D models to a given 3D shape query. This process is performed in two essentials phases: shape indexing and shape matching.

- Shape Indexing is a process that consists of designing a canonical characterization of the 3D shape of the given object. This characterization, referred to as a descriptor, serves as a key in the matching process. The main step of this phase is feature extraction.
- Shape matching is a process that consists of comparing the query with other models in the database. This comparison is performed using a similarity measure that computes distances between pairs of descriptors.

The shape indexing phase is the principal problem of content-based retrieval systems that the scientific community still works from a decade. Designing an efficient shape descriptor of a given 3D model remains a significant challenge; this is a critical point that directly influences retrieval performance (i.e., relevance and speed-up). The bulk of this challenge lies in the shape indexing process's main step, which is feature extraction.

Various feature extraction methods have been proposed in the literature (Benjelloun, Dadi and Daoudi, 2014; Lian et al., 2013; Osada et al., 2002; Savva et al., 2017; Su et al., 2015; Veltkamp et al., 2010; Pozna and Precup, 2014). Currently, the scientific community distinguishes between two types approaches: classical ones (Bai, Bai, Zhou, Zhang and Latecki, 2016; Benjelloun et al., 2014; Lian et al., 2013; Osada et al., 2002; Veltkamp et al., 2010) such as SIFT (Lowe, 2004), SURF (Bay, Tuytelaars and Van Gool, 2006), etc., and deep learning-based ones (Savva et al., 2017; Su et al., 2015). The recent astonishing results of deep learning, particularly the convolutional neural network (CNN) (LeCun et al., 2015) have yielded a significant impact on the field of 3D shape analysis (including recognition and retrieval). This machine learning algorithm has proven to be very efficient in the domain of content-based 3D shape retrieval. Indeed, all recent works presented in (Savva et al., 2017; Su et al., 2015) adopted the CNN approach to shape indexing and have shown greater accuracy.

Recently, many annotated large-scale 3D shape datasets have been made publicly available (Savva et al., 2017; Chang, Funkhouser, Guibas, Hanrahan, Huang, Li, Savarese, Savva, Song, Su, Xiao, Yi and Yu, 2015). These have contributed to developing more accurate and realistic approaches related to the analysis of 3D shapes. We can distinguish at least four approaches for large scale 3D shape retrieval (Savva et al., 2017): Multi-view projections, Point sets, Volumetric grids, and Traditional 3D shape descriptors. Except for the last, Deep learning can be used on top of all the approaches mentioned above. Multi-view projection-based techniques significantly outperform full 3D techniques, such as point sets and volumetric grids.

In this context, (Su et al., 2015) proposed a successful approach based on CNNs to 3D shape retrieval. They developed a deep learning architecture to aggregate multi-view 2D representations of 3D shapes for accurate recognition. They compared multiple training strategies and

evaluated their efficiency on both classification and retrieval. They experimented with two camera setups for view extraction, with 20 and 80 views, and each view was passed to a CNN network pre-trained on ImageNet (Russakovsky, Deng, Su, Krause, Satheesh, Ma, Huang, Karpathy, Khosla, Bernstein, Berg and Li, 2014) and fine-tuned on the dataset in question. Therefore, all views information aggregated using view pooling layers. This technique yields a significant speed-up for learning (CNNs shared the same weights) and retrieval (the number of features is not multiplied by the number of views). Their top results were 90.1% accuracy for classification and 80.2% mAP.

In the SHREC'17 track, an annual large-scale 3D shape retrieval competition (Savva et al., 2017), most of the leader board contributions were based on deep learning techniques. The top-performing team (Kanezaki et al.) proposed an invariable rotation network that takes 60 separate views and treats the pose estimation as an unsupervised problem. Zhou et al. proposed an approach, based on (Bai et al., 2016), which randomly selects a subset of 30 views and passes them through a fine-tuned VGG like CNN, then aggregates them using average pooling. Tatsuma proposed a similar approach using a reduced VGG network without aggregation. In contrast, Furuya et al. proposed a 3D point set-based method composed of four stages: generating 3D oriented points set using an algorithm proposed by (Osada et al., 2002) followed by the extraction of local rotation-invariant features, and finally 3D CNN and aggregation blocking. This method surpassed the others for the retrieval task on the second dataset (perturbed) of the challenge with a high precision p@n of 0.81. Qi et al. proposed two CNN networks for volumetric and multi-view 3D shape classification. Both networks outperformed state of the art on the MedelNet40 dataset at the time of publication, using a multiresolution technique.

3 CNN for shape indexing

Most current methods are based on the CNN for shape indexing. They use a well-trained CNN model for feature extraction instead of using other classical algorithms employed for a long time (example, SIFT, SURF). Thanks to a long learning operation, the CNN model becomes an efficient feature extractor. It is necessary here to recall that the CNN architecture generally has two parts (Figure 1):

- Feature learning: consisting of a series of convolution and pooling layers.
- Classification: consisting of fully connected layers as is the case of an ordinary neural network.

The proposed methods in (Savva et al., 2017), consider, to shape indexing, the output of the feature learning part, called CNN code, as the descriptor.

3.1 CNN architecture

Several CNN architectures proposed in the literature (Khan, Sohail, Zahoora and Qureshi, 2020) generally differ on their depth and the number of parameters to train. In this paper,

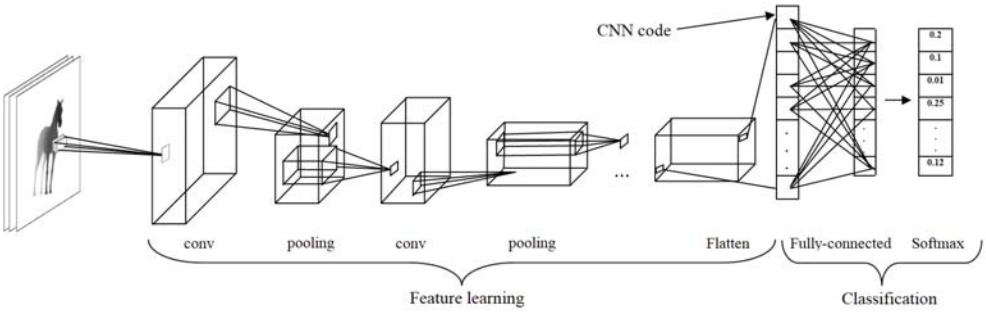


Figure 1: Convolutional Neural Network (CNN) architecture.

we investigate new deep learning architectures. When we first applied a VGG network to the problem at hand, we found that this architecture, even though it contains a limited depth, had many parameters (about 120M) that raise memory constraints (for example, when batch size = 32). Inception architecture (v1-3) brings the idea of an inception module, which consists of factoring convolution tasks into multiple branches that operate first on channels and then on space. Recently, Xception (Chollet, 2016) introduced a CNN architecture based on depthwise separable convolution, inception modules, and residual connections. This architecture uses a low number of parameters more efficiently (about 20M) and is well suited to large scale image analysis.

4 Proposed solution

This paper proposes a new approach of content-based 3d shape retrieval that meets and satisfies the two constraints of an efficient retrieval system (Relevance and computational efficiency). Our solution deals with the two phases of the retrieval process (shape indexing and shape matching).

- For the first one, we present an alternative shape indexing technique. We use the output of the classification part (softmax layer) as the descriptor instead of the CNN code used in (Savva et al., 2017).
- For the second phase, we reduce the retrieval process's search space by ignoring 3D models not similar to the query. In this case, the shape matching is performed only with 3D models susceptible to being similar to the target query. This solution's fundamental idea is that the shape indexing process is deep learning-based. Therefore, it is possible to estimate, a priori, the top-k classes to which a given object belongs.

4.1 Improving the relevance: a new shape indexing method

Our shape indexing method inspired by the CM-VGG method presented in 2017 by Deng et al. in (Savva et al., 2017) at a well-known 3D Shape Retrieval Contest (SHREC'17 Track). This

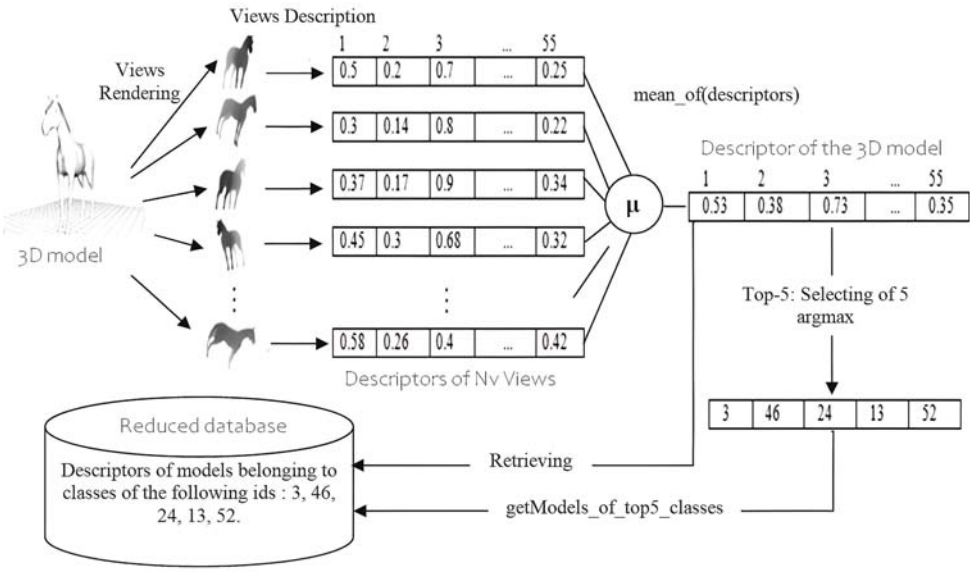


Figure 2: An overview of our retrieval system CNN-based: shape indexing and reduced database selecting.

method is also inspired by CM-BOF (Lian et al., 2013), which is a non-deep learning-based 3D shape retrieval method.

The view-based method describes a given 3D object by capturing 2D views around its 3D shape using a virtual camera. Each 2D view is characterized as a one-dimensional feature vector. As is the case of CM-VGG, we use the CNN algorithm instead of the classical version (CM-BOF), based on a bag-of-features approach, and uses a SIFT algorithm to feature extractor.

4.1.1 Our method vs CM-VGG

We have presented several improvements and modifications to the CM-VGG, which make our method different. The principal one is that we consider the classification part's output as the descriptor for a given 2D view instead of considering the CNN code (output of feature learning part) as the case of other methods. In addition to this important idea that greatly improves the relevance of the retrieval system, other improvements were carried out which concern all steps, from training to feature extraction:

- For the training step, the first improvement concerns the architecture used, which is Xception proposed by Chollet (Chollet, 2016) instead of the VGG16 used for CM-VGG (Savva et al., 2017). Compared to many recent architectures, Xception characterized by its high accuracy obtained (Chollet, 2016) with fewer weight parameters architectures. The second improvement is that we trained our model from scratch, contrary to Deng et al. in (Savva et al., 2017) who used ImageNet as the weight initialization. Unlike ImageNet, our dataset presents specific characteristics (shape, textures, gray level).

- For the number of 2D views, we use 66 views per 3D model instead of the six used by Deng et al in (Savva et al., 2017). This data augmentation technique is used to increase the accuracy.

4.1.2 shape indexing and matching

To summarize, the steps of shape indexing and shape matching are performed as follows:

- Shape indexing: as mentioned, to describe a given 3D object, we use a view-based approach that consists of capturing a set of 2D views around a 3D shape using a virtual camera. Then, each 2D view is represented as a descriptor. This description is performed using a deep learning approach. To do this, firstly, we train our chosen model, which, in this case, is Xception (Chollet, 2016). This operation is performed on a training split of the ShapeNetCore subset (Savva et al., 2017; Chang et al., 2015). Then, we use this trained model to predict the classes of each 2D image. The obtained vector of 55 elements for each is considered as the descriptor of the concerned 2D view. Note that the number 55 corresponds to the number of classes in the ShapeNetCore subset.

- Shape matching: for this step, as is the case of CM-BOF and CM-VGG methods, we use the same similarity metric called "Clock Matching". To be gemometrcally invariant of rotation and in order to solve the problem of fixing the exact positions and directions of the 3D object to the canonical coordinate frame, this dissimilarity measurement use a specific multi-view metric scheme that consist to taken into account all possible poses during the shape matching stage. For this, 24 different poses exist for a normalized model. When comparing two 3D models, one is fixed in the original orientation while the second one may appear in 24 different poses. The dissimilarity between two 3D objects is measured by the minimum distance of their all (24) possible matching pairs. The dissimilarity measurement used is :

$$\boldsymbol{D\,(O_1, O_2) = \min_{0 \le i \le 23} \sum_{j=0}^{N_v - 1} Dist(\, dV_j^{O_1}, dV_{p^i(j)}^{O_2});} \tag{4.1}$$

Where O_1 and O_2 are the two objects to be compared, N_v is the number of 2D views. $Dist$ is the distance between the descriptors $dV1$ and $dV2$ of two 2D views which is defined as follows :

$$\boldsymbol{Dist\,(dV1, dV2) = 1 - \frac{\sum_{k=0}^{N_w - 1} min(dV1\,(k)\,, dV2\,(k)\,)}{max(sum(dV1), sum(dV2))};} \tag{4.2}$$

Where N_w is the size of each 2D view.

4.2 Improving computational efficiency: Reduced retrieval space

The solution proposed above, which consists of using the output of the classification part as the descriptor of a 2D view, is used not only for shape indexing but also to accelerate the retrieval

process. Since the shape descriptor is the last-layer output that contains the probability of belonging to each class, it is used to estimate, a priori, the top-k classes to which a 3D object belongs. This allows selecting a reduced part from the whole database with which a query will be matched. This reduced part is defined as a set of 3D models belonging to the top-k classes.

To get the top-k classes of a given 3D object, we proceed as follows:

- Capture a set 2D view around a 3D shape;
- Launch the prediction for each 2D view;
- Calculate the average (we can use the max function) between all predictions;
- Sort the classes according to the obtained prediction probabilities;
- Return the top-k classes.

For a database of m 3D models categorized into C classes, the retrieval process in our case will be done as follows:

- Calculate the descriptor and estimate the top-k classes of the query object using a trained model.
- Compare the query descriptor with only models belong to the top-k classes.

Algorithm 1: Our retrieval approach

```
Function Retrieval(Query, 3D_DB, model, k) : similar 3D models
    desciptor = Initialize(55)
    foreach 2d_View of Query do
        desciptor + = model.predict(2d_View)
    end
    top_k = get_k_argmax(desciptor, k)
    foreach class_id in top_k do
        Reduced_DB + = getModels(class_id, 3D_DB)
    end
    return Compare(Query, Reduced_DB)
end
```

Example: assume we have a database of m 3D models categorized into 55 classes (as is the case of shapeNET dataset). Taking an Airplane as an object query. After capturing 66 2D views around the query's 3D shape, we execute for each one the prediction using the well-trained model. The obtained output of size 55 is considered as the descriptor of the 2D view. To predict the top-k classes of the concerned query, we calculate the average vector between the 66 descriptors of its 2D views. The predicted vector is then used to select the top-k classes' ids to which the query belongs. For example, if the selected ids for top-5 are 3, 46, 24, 13, 52, our query will be compared only with 3D models belonging to the selected ids' classes.

Table 1: Accuracy: 2D dataset.

	Top-1	Top-5	Top-10
Trained on perturbed data split for 26 epochs	76.23	92.38	94.92
Trained on normal data split for 8 epochs	75.26	91.14	94.27

Table 2: Accuracy: 3D dataset.

	Top-1	Top-5	Top-10
Trained on perturbed data split for 26 epochs	85.45	95.85	97.95
Trained on normal data split for 8 epochs	84.63	96.20	98.32

5 Experimental results & discussion

For our test, we used the SHREC'17 track specifications (Savva et al., 2017), which consist of calculating a set of standard retrieval evaluation metrics:

- Precision and Recall
- F-score
- mAP: Mean Average Precision
- NDCG: Normalized Discounted Cumulative Gain

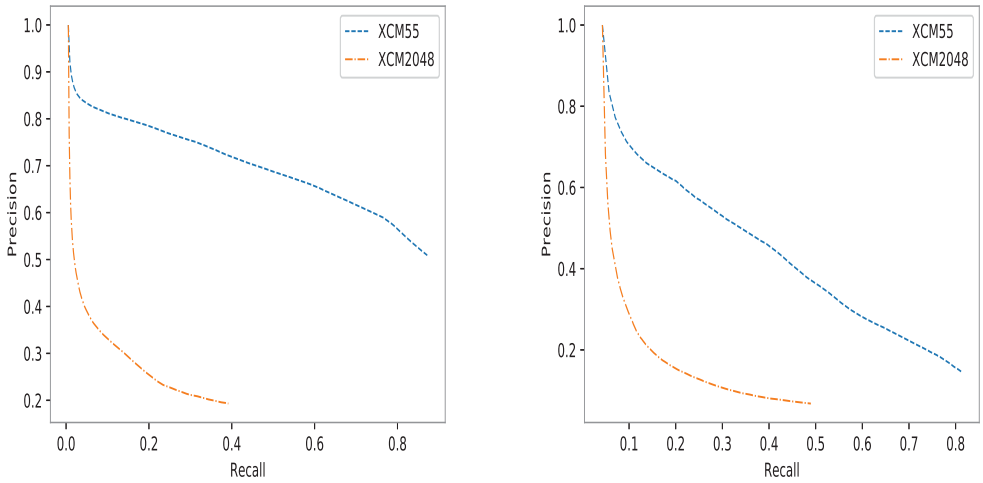
We used the Evaluator code offered by the organizer's team of the SHREC'17 track to generate the results. This code calculates two versions of the above metrics: macro- and micro-averaged. The macro-averaged is used to give an unweighted average over the whole dataset (Savva et al., 2017) while micro-averaged, is used to give a representative performance metric average across the categories (Savva et al., 2017).

5.1 Training phase & accuracy

For CNN architecture, we used the Xception model proposed by (Chollet, 2016) since it gives the best accuracy with fewer parameters (about 20M) than several other models, such as VGG16, VGG19, Inception V3, ResNet. This CNN model has 36 convolutional layers.

To train our model, we used the normal and perturbed training split of the ShapeNetCore subset dataset (Chang et al., 2015). This data used in the SHREC 2017 track (Savva et al., 2017) contains about 51,162 3D models over 55 common categories. The training dataset part contains 35,764 3D models. For each one, we captured 66 2D views. Thus, our training data contains $2,360,424$ 2D images of size 256 * 256 * 3.

To evaluate our approach and show its efficiency, we have executed eight epochs on the training normal and 26 epochs on the perturbed training parts. Table 1 shows the obtained accuracy



(a) MicroAll (b) MacroAll

Figure 3: Precision-recall comparaison of XCM55(our approach) vs XCM2048.

Table 3: Summary table of evaluation metrics

		P@N	R@N	F1@N	mAP	NDCG
MicroAll	XCM55	0.51	0.86	0.58	0.75	0.82
	XCM2048	0.25	0.49	0.29	0.31	0.45
MacroAll	XCM55	0.14	0.75	0.19	0.53	0.58
	XCM2048	0.08	0.59	0.12	0.31	0.42

on normal validation data. Note that this accuracy concerns the database of 2D images (since a set of 2D views represents a 3D object). Therefore, to calculate our dataset's good accuracy, we need to consider the average function between the 66 descriptors.

Our problem's appropriate accuracy must be calculated based on the predictions of the 3D model itself and not that of its 2D views. For our case, since the training is view-based, we calculate the accuracy by considering the average of its N_v views predictions:

$$Predict(O) = \underset{1 \leq j \leq N_v}{Mean}(predict(V_j)) \tag{5.1}$$

Where O represent a given 3D model, V_j represent jth view of O,

5.2 Relevance of our system

As is performed in (Savva et al., 2017), to evaluate the relevance of our retrieval system, we used normal-test and perturbed-test parts of the ShapeNET dataset, where each model was

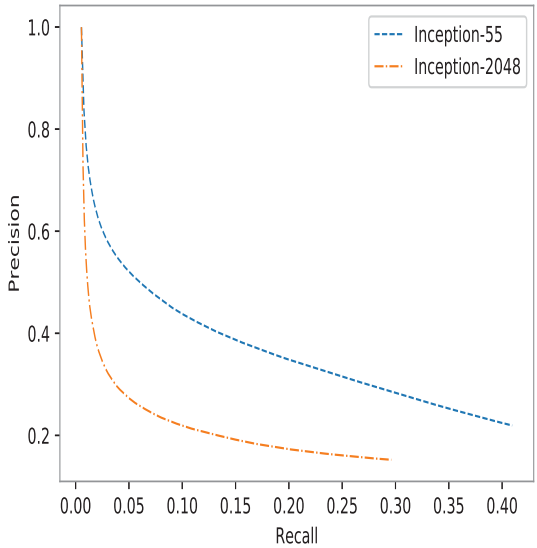


Figure 4: Our approach vs classical approach when using Inception architecture.

treated independently as a query and retrieval corpus. On this dataset, we printed out a ranked and sorted (in descending order) list of 1000 retrieved models for each model as *"Query"*.

To demonstrate the performance of our approach, we have performed multiple tests:

- The first one consists of comparing our approach with the classical one :
  - Our approach is denoted by XCM55, while 55 is the descriptor's size, corresponding to the Xception classification part's output (55 is the number of classes in the ShapeNET dataset).
  - The classical approach is denoted by XCM2048, while 2048 is the descriptor's size, corresponding to the Xception feature learning part's output.
- The second one is to prove that our approach's efficiency does not solely depend on the Xception model. We have executed the same comparison for this test but using a different architectural model, which is InceptionV3.

For the first test, the obtained results are given in Figure 3 and Table 3. Both results (Precision-Recall curve and other retrieval metrics) show that our approach (XCM55) is very relevant to the classical one(XCM2048), which means that our approach improves the relevance significantly.

For the second test, the obtained results are given in Figure 4, it shows clearly that our approach (XCM55) is efficient than the classical one even if another model is used.

Table 4: Execution time of our system for top-5.

Query id	Number of compared 3d models	Time(s)
026703	1021	0.985
012059	3463	2.720
041042	4228	3.427

5.3 Computational efficiency

Sequentially matching a given object query with the whole database that contains 10,265 3D models (the case of test split of the ShapeNET dataset) takes about 10 seconds. However, with our approach, we can benefit from a reduction up to 70% for top-5, as shown in Table 4 which presents the execution time for three examples of queries selected randomly. Note that the percentage of reduction varies from one query to another because the number of 3D models in each class is not equal and the performed test (Table 4) each query was compared to models belonging to five classes.

5.4 Choosing k

Our technique of retrieval space reducing 4.2, compares the query only with a limited part of the database. Thus, to ensure similar models for each query, it is necessary to choose the best k when selecting top-k classes. Obviously, the greater the value of k, the longer the execution time, then k must be chosen to have the best compromise ”precision/speed-up”. The figure 5, presents the accuracy for different value of k ($1 <= k <= 55$). The curve shows that from the top-10, we have an accuracy very close to 100%. For this case, for example, we can choose $k = 10$. Alternatively, instead of choosing a fixed value of k, it is possible to set a threshold. The goal is that the query will be compared to classes with a probability lower than the set one. Another method proposed by (Raziyeh and Kangavari, 2019) can be used.

6 Conclusion

This paper proposed an efficient content-based 3D shape retrieval system using deep learning. Our solution is an alternative CNN-based technique for shape indexing that provides relevant results, as well as, more importantly, a reduction in the number of models to compare in the database and consequently a reduction in execution time. Instead of comparing the query to all the models in the dataset, our approach allows predicting models likely to be similar to the query.

According to the results obtained, our approach improves the relevance and precision significantly compared to the literature. Besides, the obtained execution time is exciting. Indeed, our technique of reducing the retrieval space in the database allows a significant gain in execution time compared to systematically matching all models in the database, which means that our approach is very suitable for large databases.

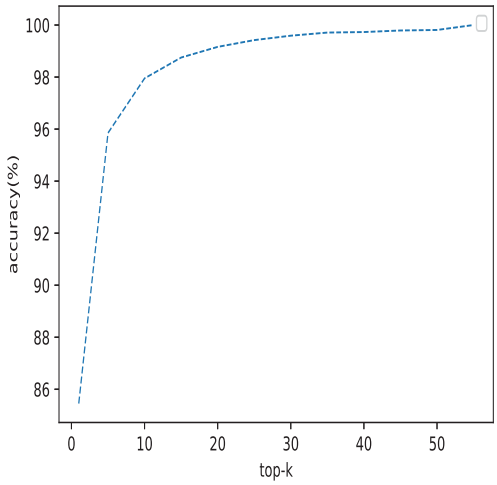


Figure 5: Top-k Accuracy.

References

Bai, S., Bai, X., Zhou, Z., Zhang, Z. and Latecki, L. J. 2016. Gift: A real-time and scalable 3d shape search engine, *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5023–5032.

Bay, H., Tuytelaars, T. and Van Gool, L. 2006. Surf: Speeded up robust features, *Computer Vision – ECCV 2006*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 404–417.

Benjelloun, M., Dadi, E. and Daoudi, E. 2014. New approach for efficiently retrieving similar 3d models based on reducing the research space, *International Journal of Imaging* **13**(2).

Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L. and Yu, F. 2015. Shapenet: An information-rich 3d model repository.

Chollet, F. 2016. Xception: Deep learning with depthwise separable convolutions, *CoRR* **abs/1610.02357**.
URL: *http://arxiv.org/abs/1610.02357*

Khan, A., Sohail, A., Zahoora, U. and Qureshi, A. S. 2020. A survey of the recent architectures of deep convolutional neural networks, *Artificial Intelligence Review* **53**(8): 54555516.
URL: *http://dx.doi.org/10.1007/s10462-020-09825-6*

LeCun, Y., Bengio, Y. and Hinton, G. 2015. Deep learning, *Nature* **521**(7553): 436–444.
URL: *https://doi.org/10.1038/nature14539*

Lian, Z., Godil, A., Sun, X. and Xiao, J. 2013. Cm-bof: visual similarity-based 3d shape retrieval using clock matching and bag-of-features, *Machine Vision and Applications* **24**(8): 1685–1704.
URL: *https://doi.org/10.1007/s00138-013-0501-5*

Lowe, D. G. 2004. Distinctive image features from scale-invariant keypoints, *International Journal of Computer Vision* **60**(2): 91–110.
URL: *https://doi.org/10.1023/B:VISI.0000029664.99615.94*

Osada, R., Funkhouser, T., Chazelle, B. and Dobkin, D. 2002. Shape distributions, *ACM Trans. Graph.* **21**(4): 807832.

Pozna, C. and Precup, R.-E. 2014. Applications of signatures to expert systems modelling, *Acta Polytechnica Hungarica* **11**: 21–39.

Raziyeh, Z. and Kangavari, M. R. 2019. On the construction of multi-relational classifier based on canonical correlation analysis, *International Journal of Artificial Intelligence* **17**(2): 23–43.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M. S., Berg, A. C. and Li, F. 2014. Imagenet large scale visual recognition challenge, *CoRR* **abs/1409.0575**.
URL: *http://arxiv.org/abs/1409.0575*

Savva, M., Yu, F., Su, H., Kanezaki, A., Furuya, T., Ohbuchi, R., Zhou, Z., Yu, R., Bai, S., Bai, X., Aono, M., Tatsuma, A., Thermos, S., Axenopoulos, A., Papadopoulos, G. T., Daras, P., Deng, X., Lian, Z., Li, B., Johan, H., Lu, Y. and Mk, S. 2017. Large-scale 3d shape retrieval from shapenet core55: Shrec17 track, *Proceedings of the Workshop on 3D Object Retrieval*, 3Dor 17, Eurographics Association, Goslar, DEU, pp. 39–50.

Su, H., Maji, S., Kalogerakis, E. and Learned-Miller, E. 2015. Multi-view convolutional neural networks for 3d shape recognition, *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV 15, IEEE Computer Society, USA, pp. 945–953.

Tangelder, J. W. H. and Veltkamp, R. C. 2004. A survey of content based 3d shape retrieval methods, *Proceedings Shape Modeling Applications, 2004.*, pp. 145–156.

Veltkamp, R. C., Giezeman, G.-J., Bast, H., Baumbach, T., Furuya, T., Giesen, J., Godil, A., Lian, Z., Ohbuchi, R. and Saleem, W. 2010. SHREC'10 Track: Large Scale Retrieval, *in* M. Daoudi and T. Schreck (eds), *Eurographics Workshop on 3D Object Retrieval*, The Eurographics Association.