Team Name: Deepverse

Github:

https://github.com/C-H-Chien/Database-Assisted-Object-Retrieval-3D-Room-Reconstruction

Title: Database Assisted Object Retrieval 3D Room Reconstruction

Contributors:

Chiang-Heng Chien (cchien3), Ramya Harikrishnan (rharikri), Sagarika Ramesh (srames19)

Introduction:

Without lifting a finger to move any furniture, what would you do if you are looking for a new look for your bedroom, living room, etc? Instead of imagining in your mind, it would be very convenient to have a 3D scene of your room such that 3D furniture can be moved around by yourself, by just taking one room picture from your phone! This project aims to build such a *database assisted object-retrieval 3D room reconstruction system* which fetches 3D furniture objects from a known database that represent very close to the furniture observed from the room photo to a 3D scene. Therefore, the system input is a single RGB image capturing some furniture, while the output is a set of database 3D object model ids. The mapping from a 2D image to a 3D scene can be formulated as a hierarchical classification with two levels: (i) the coarse level of classification estimates instances and classes of furniture objects from the input image, and (ii) the fine level classification estimates the category of the intra furniture class to find the best match 3D model id. Finally, the 3D object models are retrieved from the database and placed in a 3D scene based on user-defined object poses. The code of this project is written entirely in PyTorch, and is run on a Brown University CCV Oscars GPU node.

Methodology:

Dataset: The input 2D images we use are from ScanNet25k dataset which consists of tons of images of indoor environments such as bedrooms, living rooms, office rooms, etc. The 3D model database we are using is ShapeNet, specifically the ShapeNetCore which is a subset of ShapeNet, containing 51,300 unique 3D models of 55 common objects. As this project aims to reconstruct a room environment, we choose 8 furniture objects from the ShapeNetCore, including bed, bin, table, chair, bookcase, cabinet, display, and sofa.

Additionally, to train our fine-level classifier, we build our own furniture object dataset. This is described in detail in the next section.

System Architecture:

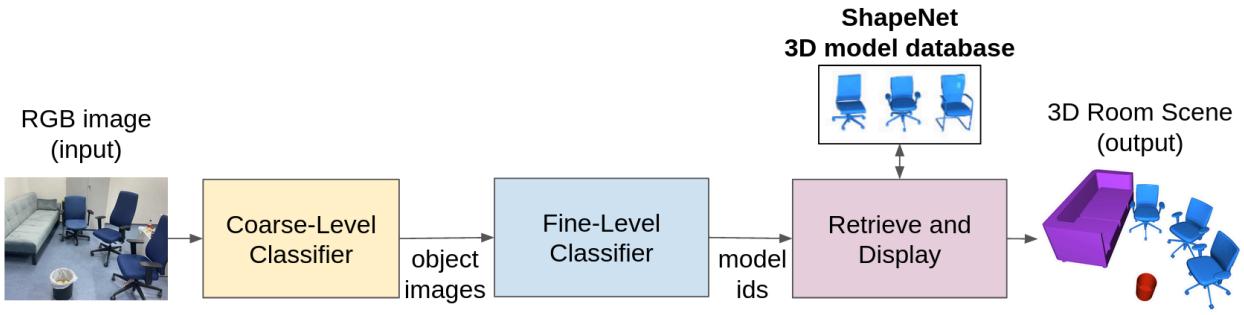


Fig. 1

A high level view of our system architecture is shown in Fig. 1. It consists of three blocks: a (i) *Coarse-Level Classifier* which estimates the instances, bounding boxes, and classes of the furniture objects from the input image; a (ii) *Fine-Level Classifier* which takes the resized, bounding box object images as input, does fine-grained classification for each object image to find the best match 3D object model id; and a (iii) *Retrieve and Display* which fetches 3D models from the ShapeNet database, takes user-input object pose information, and display 3D objects in a scene. For each block, detailed illustration is presented in Fig. 2, and the descriptions are provided below:

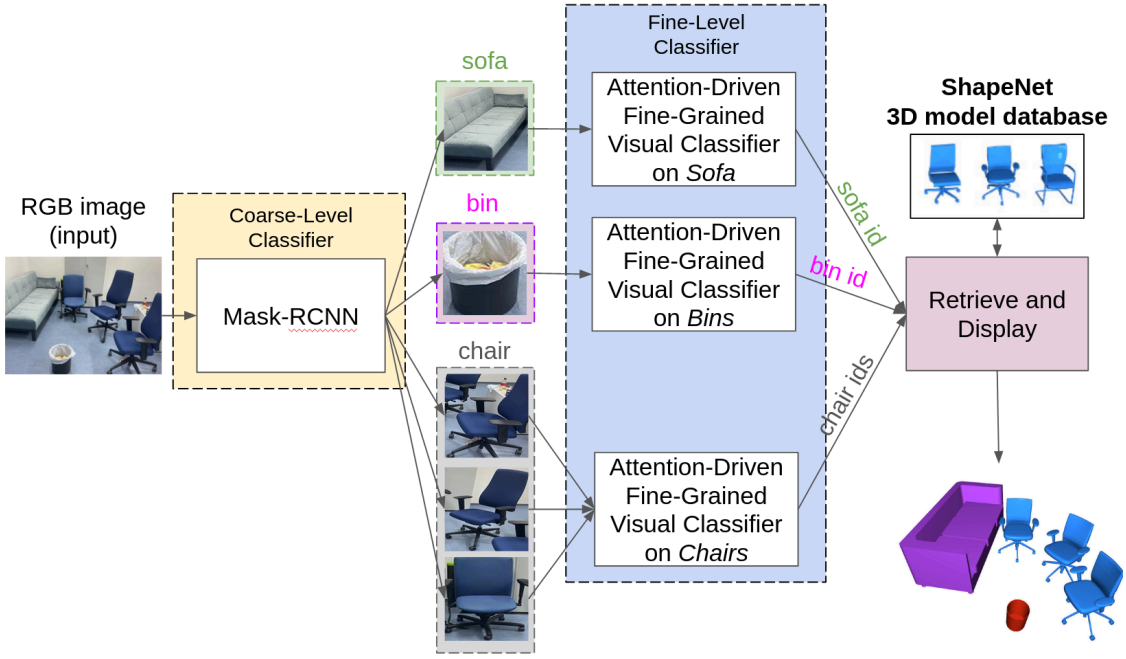


Fig. 2

(i) Coarse-Level Classifier: in this block, we use Mask-RCNN as the backbone to extract 2D instances, bounding boxes, and classes of the objects because it has been shown to be very effective in many literatures [1, 2]. The detected objects are further processed by cropping the

objects based on the detected bounding boxes.

(ii) Fine-Level Classifier: For each furniture, ShapeNet has hundreds of different types. For example, there are L-shape couches, double couches, etc under the class "sofa". Therefore, the goal of the fine-level classifier is to predict the 3D model that best represents the input object image provided by Mask-RCNN across all furniture objects of the same class. Thus, the problem is formulated as a fine-grained classification task. If a typical CNN or its variant (such as VGG net or ResNet) is used, the prediction accuracy is far from satisfactory, e.g., around 30~40% in classifying different types of "sofa". As a result, inspired by [3], we developed an attention driven fine-grained visual classifier where the architecture is shown in Fig. 3.

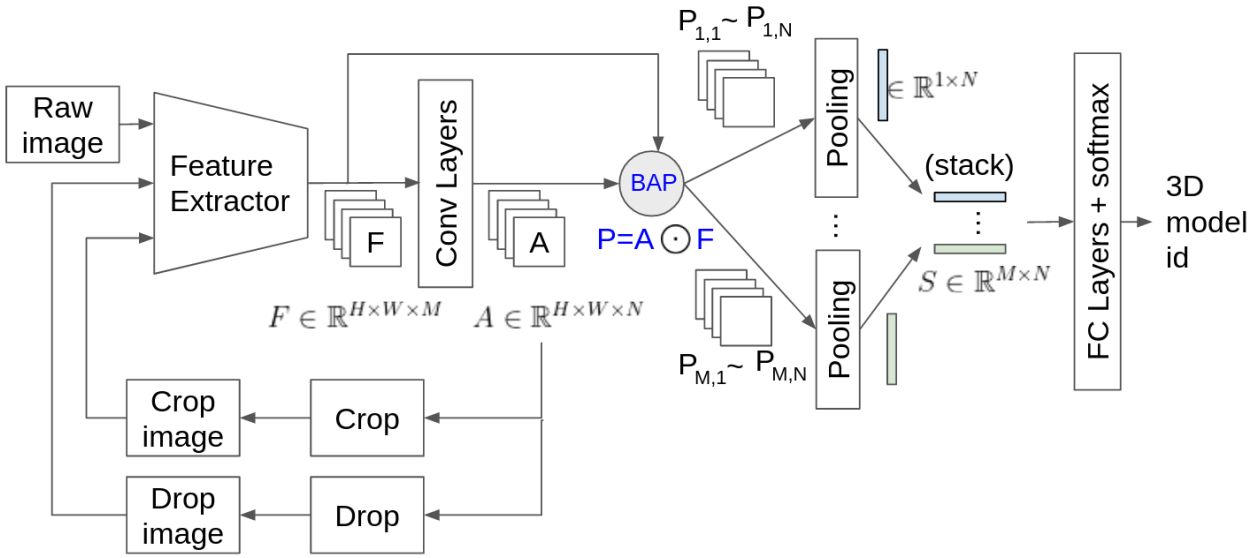


Fig. 3

The attention-driven fine-grained visual classifier works as follows. First, an input image (raw image) is fed to a feature extractor network which does transfer learning from a pretrained inception network. The output is M feature maps F, indicating "what features should the model look from the image". Then, convolution layers are used to predict N attention maps A, indicating "what part of the image should the model pay attention to". The "attention parts of the features", denoted as P, are subsequently produced by bilinear average pooling (BAP) [3] which performs Hadamard product (element-wise multiplication) between each F and all A. Thus, for each F, there are N numbers of P. To downsize P, global average pooling is employed to produce 1xN feature for each F which is stacked to create a MxN downsized attention feature map S. Finally, fully-connected layers with softmax activation are used to predict the category label.

Consider that we are trying to do intra-class classification, *i.e.*, classify different types of the same furniture, it would be helpful if the model *(i)* "looks closer" to the attention part of the features, and *(ii)* "drops common features" across all types of the same furniture so that discriminative local features can be extracted. In *(i)*, we randomly select one attention map, use the attention parts to crop the region of the raw image, and resize the cropped image before feeding into the model. As such, the model is able to look closer to the zoomed-in region, which might contain fine-grained features. As for *(ii)*, we also randomly select one attention map, but

we drop the attention part of the raw image before feeding it to the model. This forces the model to pay attention to other (local) parts of the features which might contain discriminative features.

The loss used for the training is defined by cross-entropy loss of the three images, *i.e.*, the raw image, the cropped image, and the dropped image. Plus, a center loss [4] is introduced to guide the attention convolutional network not to generate attention maps that are very much similar with each other.

To train the model, we additionally build a furniture dataset containing 8 furniture. The images come from the output of the Mask-RCNN running over a subset of ScanNet25k dataset. We manually label each category with the corresponding 3D model id, and the data is also augmented by horizontally flipping the images. For those low-qualified data, *e.g.*, blurry images, we also remove them manually. Below shows a list of all 8 furniture with the number of categories and total number of images.

- Beds: 7 categories, 274 images
- Bins: 9 categories, 1079 images
- Bookcases: 8 categories, 671 images
- Cabinets: 5 categories, 264 images
- Chairs: 10 categories, 2325 images
- Display: 6 categories, 496 images
- Sofa: 6 categories, 850 images
- Tables: 8 categories, 1056 images

(iii) Retrieve and Display: With the 3D model id predicted by the attention-driven fine-grained visual classifier, this block is in charge of retrieving the corresponding 3D model from the ShapeNet (or specifically, ShapeNetCore) dataset. The retrieved 3D models are represented by meshes, and the display is achieved via Open3D package. The placement of each 3D object, *i.e.*, object pose, is manually defined. This includes translation, rotation, and scale.

Results:

(i) Sample images of the database: Below shows some selected images of each furniture from our furniture dataset and the corresponding 3D model.

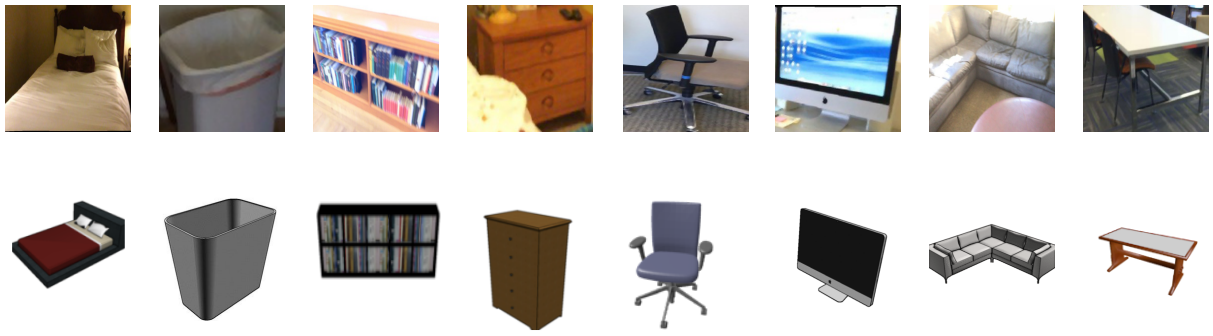


Fig. 4 (Top) Selected furniture dataset images. (Bottom) Corresponding 3D object models

(ii) Training & testing accuracy: For the coarse-level classifier, *i.e.*, Mask-RCNN, the objects are extracted if and only if the model predicts its class with more than 0.8 probability confidence, and the mask probability is over 0.7. This ensures that the object class is reliably predicted for the fine-level classifier, and it is also not occluded too much by other objects.

As for the fine-level classifier, the training accuracy with respect to epochs are shown in the following figure. 80% of the data is used for training, while both validation and testing use 10% of the data. From Fig. 5, it is evident that the model learns pretty well, reaching over 90% of accuracy for all furniture objects. This is the same for the test accuracy, where bed, bin, bookcase, cabinet, chair, display, sofa, and table has 98.1%, 97.3%, 95.77%, 96.55%, 97.05%, 96.15%, 95.45%, and 98.17% of test accuracy, respectively.

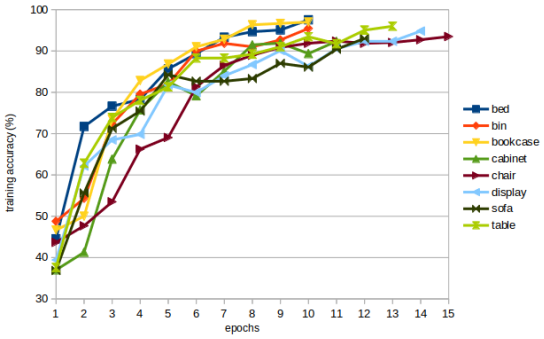
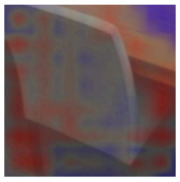
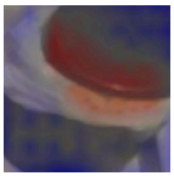
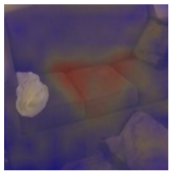


Fig. 5

(iii) Attention heat map: To interpret which region the attention-driven fine-grained visual classifier is paying attention to, we selected a number of images and the attention heat map as shown in Table 1. The red parts of the heat map represent the hot spots that the model relies on to make predictions. For example, the model puts more attention on the "legs" of the chair instead of the "shape" of the chair. Similarly, the heat map on the sofa shows that the number of seats is focused on. These discriminative features are indeed the features that distinguish one type of the furniture from another.

Table 1

raw image				
attention heat map				

(iv) Reconstruction results: Fig. 6 shows some reconstruction results. The pose of each object is carefully defined in such a way that they resemble very much alike from the 2D input image.

Fig. 6 Selected reconstruction results.

Challenges:

We faced a few conceptual and technical challenges. For the conceptual challenges, initially, we had a hard time understanding the conceptual framework of reconstructing a 3D object from a 2D image. But soon into this, we realized that the accuracy of reconstructing the 'geometry' of the 3D objects was not as crucial as accurately capturing the furniture altogether. This way, it was enough to just correctly classify the type and subcategory of the object instead of entirely replicating it. Hence, the idea of using a database assisted model to fetch the closest object was applied.

As for the technical challenges, at first, we were planning to generate 3D point clouds of the observed 2D outputs, but our results were not quite accurate. Following this, we decided to implement a simple pipeline with a basic convolutional neural network which however could only precisely classify the objects, not giving attention to the minute details of the object like the legs of the table. To tackle this problem, we decided to include an Attention-Driven Fine-Grained Classifier that was able to capture every inch of the object. We also observed a lot of inconsistencies in the dataset that impacted our accuracies. We had to manually re-label wrongly classified images and remove some non-qualified images from the dataset from each class. Given the size of the dataset, this was very time consuming as well.

Reflection and Discussion:

Although we faced several difficulties in our implementation, each part of the pipeline was a great learning experience for us. We were able to successfully reconstruct 3D objects that closely resemble the observed 2D object from the image and obtain high accuracies for each object category. Currently our model only uses eight classes with few broad categories in each. For example, the 'chair' class is being classified into 10 categories. In the future, we can include more classes and categories for each class to allow the model to distinguish more different types of furniture. Additionally, our current implementation requires manual change of position of the reconstructed 3D object to recreate the similar room scene. We could also look into incorporating an architecture that estimates and automatically moves the pose of the object as in the original room. Other future works could include extracting the room layout and embedding that with the 3D reconstructed objects to re-create the whole room. Overall, this experience was very fruitful for each of our team members.

Bibliography

[1] Gümeli, Can, Angela Dai, and Matthias Nießner. "ROCA: Robust CAD Model Retrieval and Alignment from a Single Image." *arXiv preprint arXiv:2112.01988* (2021). [url: https://arxiv.org/pdf/2112.01988.pdf]

[2] Kuo, Weicheng, Anelia Angelova, Tsung-Yi Lin, and Angela Dai. "Patch2CAD: Patchwise Embedding Learning for In-the-Wild Shape Retrieval from a Single Image." In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12589-12599. 2021. [url: https://arxiv.org/abs/2108.09368]

[3] Hu, Tao, Honggang Qi, Qingming Huang, and Yan Lu. "See better before looking closer: Weakly supervised data augmentation network for fine-grained visual classification." *arXiv preprint arXiv:1901.09891* (2019). [url: https://arxiv.org/abs/1901.09891]

[4] Wen, Yandong, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. "A discriminative feature learning approach for deep face recognition." In *European conference on computer vision*, pp. 499-515. Springer, Cham, 2016.