BIGCHAIR*

Julian Quevedo [1] Sidney Hough [1] Pino Cholsaipant [1]

[1]Department of Computer Science

Problem

Language models that encode structural information about 3D shapes may facilitate efficient and expressive retrieval of 3D information and aid humans in the model design process. Retrieval capabilities are helpful for querying databases of 3D shapes without human annotation. Generative capabilities could ease software usage and inspire new designs.

BIGCHAIR, or BI-modal Graph-CHAracter learnIng for Retrieval, is about learning a joint embedding space for natural language descriptions and 3D shapes which can be used for downstream tasks like retrieval and generation. In contrast to previous approaches [1], we use 3D shape meshes instead of voxelizations. Our model uses a **text encoder** (pre-trained transformer) to encode object descriptions and a **graph encoder** (Graph Attention Network [2]) to encode the meshes of 3D shapes. We also leverage descriptive contexts to capture important features of shape descriptions. Our learning goal is to make description embeddings similar to their corresponding mesh embeddings.

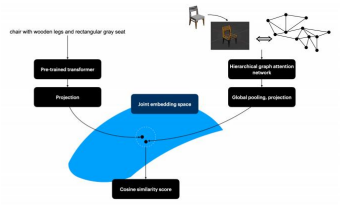

Figure 1. Basic architecture

Background

Text2Shape: Text2Shape [1] is a previous work that similarly learns a joint embedding space for text and 3D shapes. Instead of using mesh data, however, it encodes shapes as voxels and learns a CNN over them. Applying the same convolution operation over more or less dense voxels leads to information loss or wasted computation.

We use the same dataset as Text2Shape: ShapeNet plus crowdsourced textual descriptions of its chairs and tables. The novelty in our project is in converting 3D object into meshes, which we then represent as **graphs**. This allows us to make use of the versatile and powerful class of models called **graph neural networks (GNNs).**

Graph neural networks: GNNs update node representations over several layers. In our case, nodes encode mesh vertices. Each node defines its own computation graph, aggregating and transforming messages from its neighbors in convolutional layers. We base our approaches on a GNN variant known as a **Graph Attention Network** [2] (GAT). GAT layers computes attention embeddings over node pairs, allowing nodes to assign weight to more important neighbors.

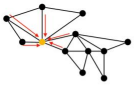

Figure 2. Visualization of a node (in gold) processing embeddings from its neighbors; this processing is done for every node in the graph

Methods

Graph encoder design. We experimented with a vanilla *Graph Attention Network* (GAT) and a more complex model, *Directional Mesh Encoder*. The Directional Mesh Encoder is a stack of GAT convolutional layers complemented with an edge convolutional operator, which learns an MLP over the difference in a node's features and its neighbors' to update that node. Because we encode vertex coordinates as node features, this module essentially computes and transforms distances between vertices. We also derive a graph-level representation by learning an MLP over the concatenation a global mean pool and global max pool of the final node embeddings.

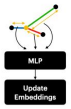

Figure 3. the directional mesh encoder essentially processes physical distances between mesh vertices, feeding them into an MLP and using the results to update node embeddings

Aspect-aware text encoding. In our task of 3D shape retrieval, the descriptive components of the query sentence are the most critical and should be exploited. Thus, we extracted adjective-noun pairs from the descriptions, running these key words separately from the text through a transformer; this gives us *aspect-aware* embeddings of the descriptions. These feature-level embeddings are concatenated with the global-level embeddings.

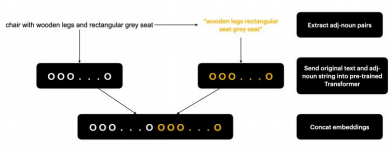

Figure 4. text encoder, with adjective-noun pair extractions

Experiments

Model	Train RR@5	Test RR@5
Random Retrieval	$\approx 1\%$	$\approx 1\%$
GAT	3.84%	2.67%
GAT + Aspect	**5.92%**	**3.92%**
Directional + Aspect	2.04%	1.72%

Table 1. Train and test recall rates in top-5.

To evaluate the performance of each model, we use the recall rate (RR@k) of text-to-shape retrieval. After contrastive pretraining, we compute embeddings for all of the meshes and natural language descriptions in our validation dataset. For each description embedding, we compute its dot product similarity with each mesh embedding. The recall rate at 5 is the is the percentage of descriptions whose corresponding mesh's similarity score is among the top 5 scores.

Analysis

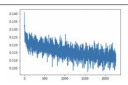

Figure 5. Loss: GAT + Aspect

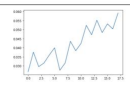

Figure 6. Train accuracy: GAT + Aspect

The relatively simple GAT trained jointly with a Transformer from CLIP, augmented with description embeddings, performed best on the retrieval task. The more complex Directional Mesh Encoder performed poorly in our initial experiments. This is likely due to a small dataset and a disproportionately deep model (3 convolutional layers + a 2-layer pooling MLP + a 2-layer edge convolutional layer). Considering the trajectories of the loss and accuracy plots, we would likely achieve better results with more compute resources and time.

Model	Train RR@5	xBetter than random
Text2Shape	2.37%	$\approx 8x$
GAT + Aspect	3.92%	$\approx 4x$

Table 2. Train and test recall rates in top-5.

While our model does worse relative to chance than Text2Shape, we are learning a more difficult task: meshes maintain hundreds of vertices and edges, as well as complex structural information. Given extra training time, a larger dataset (we used a smaller subset than Text2Shape) and optimized architectures, a mesh encoder could plausibly outperform Text2Shape on ShapeNet. With a dataset containing more intricate shapes (chairs and tables are relatively block-y), our mesh encoder would likely be superior.

Conclusions

We see that the graph is a powerful abstraction for 3D objects, and that GNNs produce shape embeddings compatible with language descriptions. When coupled with expressive pre-trained transformers and aspect-aware features, GNN-based mesh encoders perform reasonably on shape retrieval. Our quantitative and qualitative evaluations show evidence of low-hanging fruit that could strongly boost model performance.

References

[1] Kevin Chen, Christopher B. Choy, Manolis Savva, Angel X. Chang, Thomas A. Funkhouser, and Silvio Savarese. Text2shape: Generating shapes from natural language by learning joint embeddings. *CoRR*, abs/1803.08495, 2018.

[2] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2018.

[3] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics*, 38(5):1–12, Nov 2019.

julianq@stanford.edu shough@stanford.edu pschol@stanford.edu