Topic E - Exploring Unified 3D Point Cloud Representation

Insalaco Ugo
ugo.insalaco@hotmail.fr

Abstract

Uni3D [4] builds on top of CLIP [3] to create a foundation model for text, images and 3D models. This method aims at learning an encoder for 3D point clouds that maps a 3D model in the same space as the CLIP space for images and text while keeping semantic similarity between the latent representation of an object and its text description and well as the 2D image renders of this same object. In this project, I try to reproduce the results obtained in the base paper for zero shot classification on the objaverse LVIS dataset [1], and study more in depth this classification for color-less, and partial point clouds obtained from a single view. I also propose an implementation of the painting experiment from the Uni3D article [4] which goal is to produce a coloration of a point cloud using prompt or image indications.

1. Introduction

In order to process the complexity of the world, we humans, have access to a range of sensory captors from which we internally construct a representation of the world. It seems natural that information provided by different type of sources (e.g. a sound or a visual stimuli) can represent the same object (e.g. a meowing or the vision of a small quadruped animal with a tail and pointy ears for a cat). In the field of Artificial intelligence, the CLIP model [3] aimed at learning a shared latent space for text and images which made it one of the first multimodal algorithm and allowed to tackle a variety of challenging tasks such as open vocabulary computer vision or image generation. Uni3D [4] took a step forward and extended this concept to a joint representation between text, images and 3D models. This new foundation model can be used in a wide range of applications, from zero shot classification, to model painting or point cloud reconstruction.

2. Zero shot classification

Zero shot classification consists of taking the trained Uni3D model and using it as a classifier. This is possible by using text templates for each class (”a 3D model of a ... ”) and comparing the similarity between the completed template for each class, and the 3D model encodings, choosing the classification class as the one with the most similar encoding to the 3D model’s.

We compare the authors results (Authors) with the reproduced classification results obtained by encoding the entire colored point clouds (Repro), the uncolored point clouds (No color), and a reduced version of the point cloud obtained by removing hidden points from a reference camera position (Partial). A picture of the different processing is shown fig.1

2.1. Data and models

In this first task, the goal is to reproduce the experimental setup for zero shot classification on a subset of the Objaverse LVIS dataset. This dataset contains 1126 classes, and our subset is made of 9997 objects randomly sampled from the base dataset. The model used is Uni3d-g trained on an ensembled dataset of 3D objects without LVIS, and made available by the authors here. The CLIP encoder is ”EVA02-E-14-plus” available here.

2.2. Code and environment

The code used was based on the authors’ on github and a main file was created for each experiment. The results were saved in a separate repository with the modified version of the code. The experiments were conducted on google colab using a T4 GPU. In a first environment setup I used GPUs kindly provided by the teaching team on google cloud, but due to the computational needs of the project and the difficulty to find available virtual machines, I managed to do the experiments on the base colab setup. However, the RAM limit on google colab (13GB on CPU and 15GB on GPU) didn’t allow to load the full clip encoder (that requires 29 GB of RAM), which is why I used CLIP features computed by Mrs Chen (data/lvis_text_features.npy in my repository) and only ran the Uni3D point cloud encoder at inference.

The partial point clouds were computed with a hidden point removal algorithm by [2] implemented in the Open3D python library.

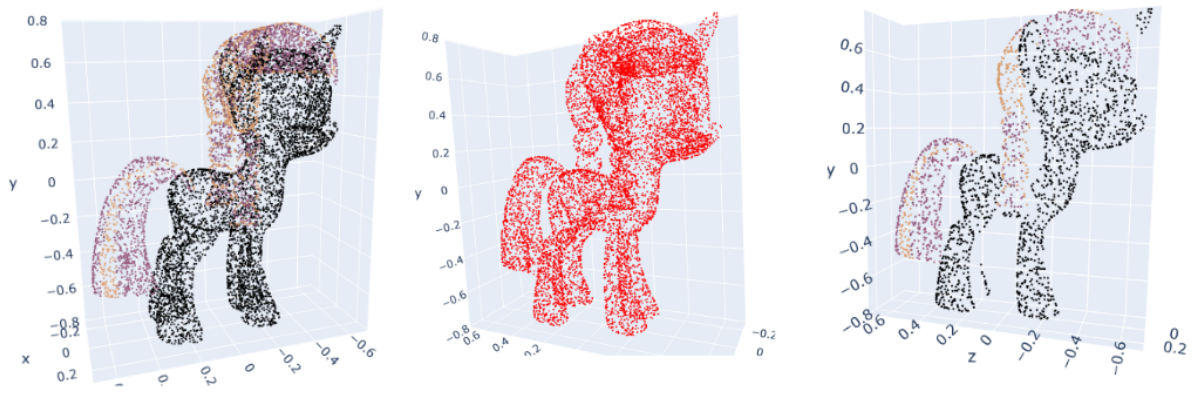


Figure 1. Base model (left), model w/o color (center), partial model (right). Note that the input color given to Uni3D in the "no color" model is actually black (0)

	Top 1(%)	Top 3(%)	Top 5(%)
authors	47.2	68.8	76.1
repro	46.9	68.4	75.7
no color	43.3	63.7	71.1
partial	20.1	32.9	39.4

Table 1. Results for zero shot classification. Authors: the authors' results [4], Repro: reproduced zero shot classification, No color: classification without the color data, Partial: classification by removing hidden points from certain angles.

2.3. Results

I compare the classification accuracy for Top 1, Top 3 and Top 5 prediction meaning that the target class has to be in either top 1, 3 or 5 in the classification ranking of the model's logits.

According to tab.1, the reproduced classification results fit almost perfectly the ones from the article, even considering that we took a subset of the LVIS dataset. As for the impact of the color on the prediction, we note that color reduces the accuracy by only a few percent. This could mean that the model doesn't rely that much on the color of the point cloud to create its internal representation. However the results for the partial point clouds show a great reduction after removing hidden points. The model therefore must use points from multiple angles to create a reliable encoding.

3. Class by class study

In this section, I conduct the same experiment than before, but grouping the results by class. The hypothesis are that the model can have worse accuracy for some classes, and that some class can be more impacted by others on the loss of color or the removal of occluded points. for this experiment I focused on three classes: cat, parrot and book, however the results can be computed easily on the Kaggle notebook

3.1. Results

tab. 2 shows the results for the three classes, we observe that according to the class, the behavior can be very different. The cat and book class don't seem impacted a lot by the loss of color (the cat class even improves its accuracy without color) whereas the parrot class suffers much more from it. This is possibly because parrots models are colorful objects that have specific patterns and that removing its color might cause the Uni3D to confuse it with other classes of birds like "parakeet" (fig.3) Oppositely, the cat and parrot accuracy diminish a lot when using partial point clouds while the book class has its score increasing. We may explain this by the fact that the book models are mainly 2-dimensional since they are flat and that it makes it robust to the reduction of points in our process.

4. Painting

4.1. Data and models

For this experiment I used 3D models of a cat, a unicorn and a butterfly. Different prompts were written in order to color specific parts of the model, for instance:

- A unicorn with a red horn
- A cat with a blue belly
- A cat with a purple tail

	Top 1 (cat)	Top 3 (cat)	Top 1 (parrot)	Top 3 (parrot)	Top 1 (book)	Top 3 (book)
zero shot	63.6	86.3	72.7	100	12.5	87.5
no color	77.2	90.9	36.4	63.6	12.5	75.0
partial	9.0	36.3	27.3	36.4	37.5	75.0

Figure 2. Class by class comparison of accuracy on zero shot classification using Uni3D

Figure 3. Exemple of a cat (left) parrot (center) parakeet and book (right) from the objajverse LVIS dataset

At the same time I used clip encodings of an image of a tiger, with the hypothesis that the encoding would allow to color the cat model according to the image colors. The model used were a frozen Uni3d-g and the EVA02-E-14-plus CLIP encoder. During "training", the optimizer changed the base uncolored 3D model's pixel values in order to increase the cosinus similarity between the point cloud encoding and the text encoding.

4.2. Code and environment

This experiment was run on a Kaggle notebook which allows for larger RAM capacity and makes it possible to load the CLIP model. First the text and image encodings were computed on CPU, and in a second time I ran the optimization loop on the 3D model's pixel values on GPU.

4.3. Results

During this experiment I had very poor results, and the optimization didn't seem to converge toward something convincing. Some times it happened that there was a localized change in the model color (the back of the cat, the horn of the unicorn) but by reproducing it multiple times it only seemed like it was the product of luck (see fig.**??**). Technically, when optimizing the input 3D model, the cosine similarity between the model and the target vector remains low at all time (never more than 0.05) and I don't manage to find a similar encoding by simply varying the pixel colors. This was also due to the difficulty of choosing hyper-parameters for this training, as I had to choose between different kind of optimization algorithm (SGD or Adam), a relevent learning rate (having a low learning induced a really slow training, far from the "second" mentioned by the authors and a large learning rate produced aberrant results), the right number of epochs, and choosing to start from a random, white or original version of the colored 3D model.

5. Conclusion

In this project I studied the performances of the Uni3D model and showed that the author's results on zero shot classification were reproducible and I studied more in depth how changing the input point cloud could impact the classification results. The qualitative study on painting was however not conclusive which is also a consequence of the lack of details provided by the authors on their experimental setup.

This project was technically very challenging, as the environment setup was not well documented by the authors on their github it had outdated packages (I had to locally modify the PointNet setup file to have it installing correctly) or packages only being available on linux (like deepspeed) that forced me to work on Colab or Kaggle notebook. The project was also very consuming in GPU resources and being able to connect to available GPU on google cloud was another challenge, finding the right place to have enough resources was another time consuming and not satisfying part of the project. Fortunately the pedagogical team of MVA was responsive and helped me with some issues, for which I am thankful.

Nevertheless I learned a lot on different state of the art implementation of models like CLIP, or Uni3D and manipulated point clouds through the use of Open3D, as well as set up an optimization loop for changing the input point cloud which for me was very interesting.

A. painting input images and prompts

Prompts:

- A cat with a blue belly

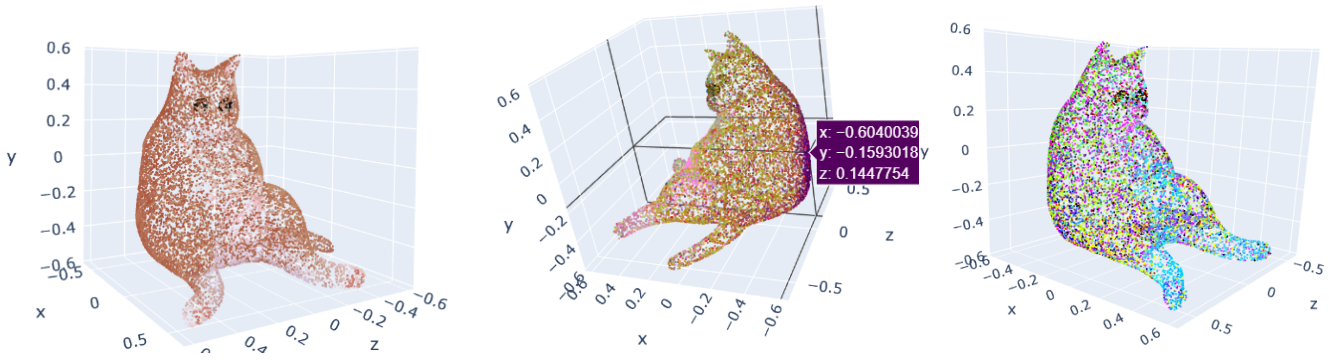


Figure 4. Base cat model (left) and results for the prompt: "A cat with a blue belly" (started from the base colors in the center and from random colors on the right)

Figure 5. Tiger image used as target for painting

- A cat with a purple tail
- A unicorn with a yellow horn
- A butterfly with a blue wing and a yellow wing (same as the initial paper)

Images: see fig5

B. Additional painting results

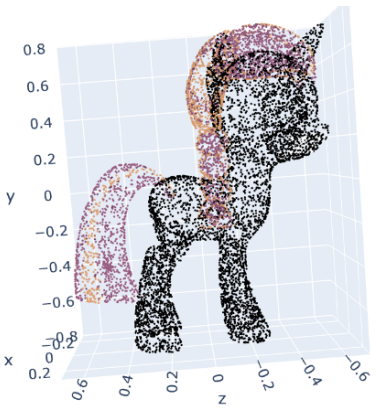
References

[1] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13142–13153, 2022. 1

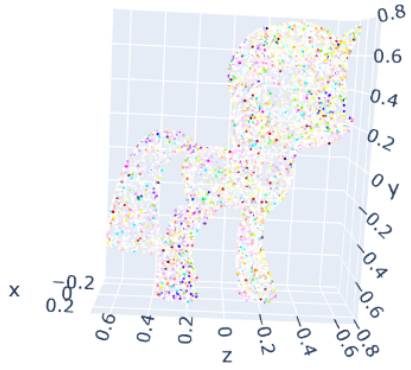
[2] S. Katz, Ayellet Tal, and Ronen Basri. Direct visibility of point sets. *ACM SIGGRAPH 2007 papers*, 2007. 1

[3] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, 2021. 1

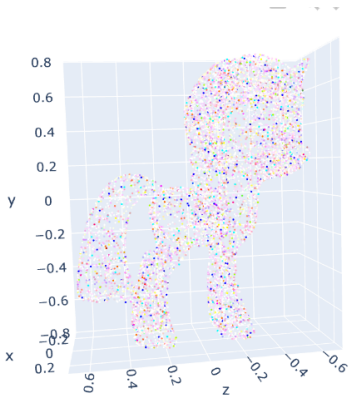
[4] Junsheng Zhou, Jinsheng Wang, Baorui Ma, Yu-Shen Liu, Tiejun Huang, and Xinlong Wang. Uni3d: Exploring unified 3d representation at scale. *ArXiv*, abs/2310.06773, 2023. 1, 2



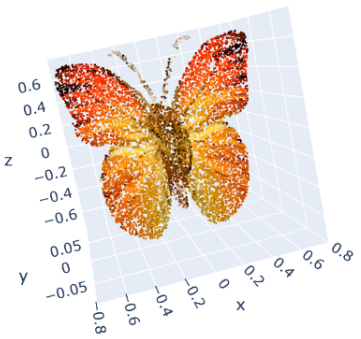
(a) base unicorn model



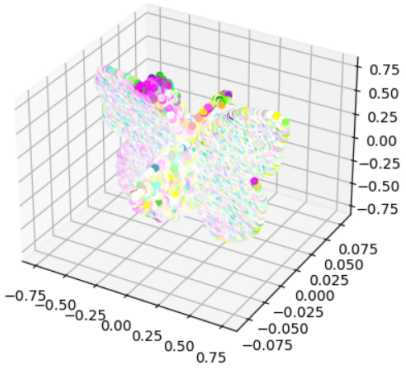
(b) painting the unicorn with the tiger image, started with a white model. We can assume that the yellow color comes from the tiger on the image being mostly orange and white



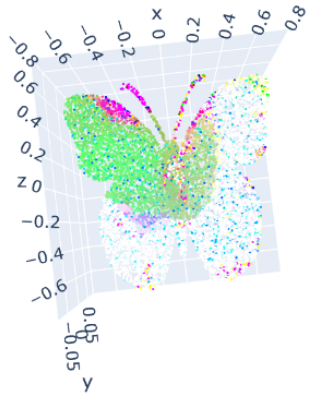
(c) painting the unicorn with the prompt: "A unicorn with a yellow horn", started with a white model.



(a) base butterfly model



(b) butterfly model painted with the prompt "A butterfly with a yellow wing and a blue wing". Here we can see appear the yellow color on one wing and the second coloring pink, both with cyan dots



(c) butterfly model painted with the prompt "A butterfly with a blue wing and a yellow wing"