Universidad Peruana de Ciencias Aplicadas

Ciencias de la Computación

Machine Learning (CC72)

TA 1

Profesor: Luis Martín Canaval Sánchez

Integrantes:

- Bruno Tiglla Arrascue (U20181A686)
- Junior Alfredo Huerta(U201620846)
- Sebastián Fernando Peralta Ireijo (U201816030)

Ciclo: 2022-02

Septiembre, 2022

1). FORMATOS

En el artículo proponen utilizar una red adversarial generativa de reconstrucción de objetos más conocida en inglés como (ORGAN) donde usarán una red neuronal convolucional 3D (CNN) con conexiones salteadas. Con un generador de GAN condicional (CGAN). Donde, se contará con dos objetos de optimización: *a mean absolute error (MAE) and an Improved Wasserstein GAN (IWGAN)*

Red Neuronal Convolucional (*CNN*) son un tipo de redes neuronales artificiales donde las neuronas corresponden a campos receptivos de una manera muy similar a las neuronas en la corteza visual primaria de un cerebro humano.

Red Adversaria Generativa (*GAN*) es una máquina de aprendizaje que consta de dos redes neuronales que compiten para producir predicciones más precisas, es decir, la ganancia o la pérdida de una de las redes se compensa con la ganancia o pérdida de la opuesta.

Mean Absolute error (*MAE*) es una medida de la diferencia entre dos variables continuas

Improved Wasserstein GAN (*IWGAN*) es una extensión de la red adversarial generativa que mejora la estabilidad al entrenar el modelo y proporciona una función de pérdida que se correlaciona con la calidad de las imágenes generadas.

El formato OFF (Objective File Format) utilizado por el grupo del artículo en el que se desarrollará las siguientes partes del trabajo. Es un tipo de formato para la creación de figuras 2D y 3D, basado en el uso de polígonos

La composición del formato OFF, recibe en el primer parámetro las siglas del OFF para verificar que es un archivo de este formato. En la siguiente línea recibe 3 parámetros, el número de vértices, el número de caras, y el número de bordes(este último puede ser ignorado).
A partir de ahí recibe la lista de vértices dando las coordenadas X,Y,Z de cada punto.
Luego de los vértices se solicita ingresar los datos de las caras, donde el primer parámetro a ingresar es la cantidad de lados que el polígono tendrá para esa cara. A partir de este primer parámetro se tendrán que enviar la cantidad exacta de los índices de los vértices que se desean unir en dicho orden. De igual manera se pueden colocar los materiales y el RGB en esta parte de las caras.
Imagen 1: Cubo tridimensional en formato OFF.

```
OFF
8 6 0
0.0 0.0 1.0
1.0 0.0 1.0
0.0 1.0 1.0
1.0 1.0 1.0
0.0 1.0 0.0
1.0 1.0 0.0
0.0 0.0 0.0
1.0 0.0 0.0
4 0 1 3 2
4 2 3 5 4
4 4 5 7 6
4 6 7 1 0
4 1 7 5 3
4 0 2 4 6
```

2). STL FILE FORMAT

El formato de archivo .stl tiene como principal uso la impresión y diseño (CAD) en 3D. Las siglas stl tiene como significado *stereolithography*, tecnología popular para la impresión 3D. Consiste en el proceso de creación de objetos 3D usando un dispositivo que emite luz, iluminando y expulsando, en estado líquido, fotopolímero. Este material se expulsa formando capas de tal manera que se crea un objeto. Cada archivo .stl está conformado por una serie de triángulos interconectados que describen la geometría de un modelo 3D o objeto. Mientras más complejo el diseño; es decir, tenga una mejor resolución y nivel de detalle, mayor será la cantidad de triángulos.

El formato de archivo .stl tiene dos presentaciones: ASCII STL y Binary STL. Ambos pueden contener o representar el mismo modelo, sin embargo, los archivos binarios son más livianos en términos de almacenamiento. Asimismo, son más fáciles de procesar por Software. Por otro lado, los archivos ASCII son más fáciles de modificar de manera manual.

El formato de STL requiere de un primer parámetro solid/endsolid para especificar que estará en dicho sólido, luego se agrega la normal de a donde mira la cara de ese triángulo generado. facet normal/endfacet y por último el outer loop/endloop donde se colocan los 3 vértices que conforman dicho triángulo. A diferencia de OFF el orden en el que se colocan los triángulos afecta.

Imagen 2: Cubo tridimensional en formato STL ASCII, extraído de blender.

```
solid Exported from Blender-3.3.0
facet normal 0.000000 0.000000 0.000000
outer loop
vertex 0.000000 0.000000 1.000000
vertex 0.000000 1.000000 1.000000
vertex 1.000000 1.000000 1.000000
endloop
endfacet
facet normal 0.000000 0.000000 0.000000
outer loop
vertex 0.000000 0.000000 1.000000
vertex 1.000000 0.000000 1.000000
vertex 1.000000 1.000000 1.000000
endloop
endfacet
facet normal 0.000000 0.000000 0.000000
outer loop
vertex 1.000000 0.000000 0.000000
vertex 1.000000 0.000000 1.000000
vertex 0.000000 0.000000 1.000000
endloop
endfacet
facet normal 0.000000 0.000000 0.000000
outer loop
vertex 1.000000 0.000000 0.000000
vertex 0.000000 0.000000 1.000000
vertex 0.000000 0.000000 0.000000
endloop
```

3). CONVERSIONES

En vista de que en el trabajo guía de "3D Reconstruction of Incomplete Archaeological Objects Using a Generative Adversarial Network" desarrollado por estudiantes de la PUCP, utilizan el formato OFF para la reconstrucción de los modelos 3D, se optara por este tipo de formato para la futura generación de piezas faltantes usando machine learning. El objetivo de este próximo proyecto es generar un modelo 3D de las piezas faltantes de modelos 3D. Dado que la recomendación para poder utilizar de manera precisa las máquinas necesarias para la impresión 3D en el FabLab es utilizar archivos en el formato STL. Es necesario desarrollar una función que convierta los archivos de tipo OFF a STL para poder utilizar de mejor manera las impresoras.

4). ALGORITMO

Enlace al notebook donde se trabajó el algoritmo sin script.
https://colab.research.google.com/drive/1mY2mD_2AkhpS6XQtW5a7TU4CHvjj7JuS#scrollTo=y80DS_o6Jt_H

El algoritmo leerá un archivo OFF sin comentarios y sin RGB asignado a las caras para poder realizar la conversión a formato STL en formato ASCII.

```
import numpy as np

class off_2_stl:
    def __init__(self):
        self.stl = None
        self.facet = None

    def fit(self, filename):
        with open(filename+'.off') as f:
            x = f.readline()
            if x == "OFF\n":
                print("Es un archivo OFF")
            else:
                print("No es un archivo OFF")
                return
            # numero de vertices, caras y bordes
            a, b, c = [int(x) for x in f.readline().split()]
            # guarda los vertices del poligono
            vertices = np.zeros((a,3), dtype= float)
            self.stl = [] # guarda coordenadas que conforman un triangulo
            vertexlist = [] # guarda vertices
            self.facet = [] # almacena calculos de la normal
            for i in range(a):
                vertices[i] = f.readline().split()
```

```
            for line in f:
                t = ([int(x) for x in line.split()])
                vertexlist.append(t)
            for i in range(b):
                ix = vertexlist[i][0]
                ix = ix -2
                for j in range(ix):
                    self.stl.append([vertexlist[i][1], vertexlist[i][j+2],
                                     vertexlist[i][j+3]])
            for i in range(len(self.stl)):
                for j in range(3):
                    p = self.stl[i][j]
                    self.stl[i][j] = vertices[p]
            for i in range(len(self.stl)): # calculo normal: (p2-p1) x (p3-p1)
                p1 = np.array(self.stl[i][0])
                p2 = np.array(self.stl[i][1])
                p3 = np.array(self.stl[i][2])
                N = np.cross(p2-p1, p3-p1)
                self.facet.append(N)

    def gen_stl_file(self, newfilename):
        with open(str(newfilename)+'.stl','w') as stlscript:
            stlscript.write('solid\n Creado por el grupo de Machine Learning')
            for i in range(len(self.stl)):
                stlscript.write('facet normal {0} {1}{2}\n'
                                .format(self.facet[i][0],
                                        self.facet[i][1],
                                        self.facet[i][2]))
                stlscript.write('outer loop\n')
                for j in range(3):
                    stlscript.write("vertex {0} {1}{2}\n"
                                    .format(self.stl[i][j][0],
                                            self.stl[i][j][1],
                                            self.stl[i][j][2]))
                stlscript.write('endloop\n')
                stlscript.write('endfacet\n')
            stlscript.write('endsolid\n')


converter = off_2_stl()
print("Make sure the OFF file is in the same folder as the program")
origin_fl = input("Enter the name of the .off file:")
```

```
new_fl = input("Enter the name of the converted .stl:")
converter.fit(origin_fl)

converter.gen_stl_file(new_fl)
```

Instrucciones de uso:

Para correr el script se sigue el siguiente flujo:

1. Guardar el archivo .off en la misma ruta donde está el script *off2stl.py*.
2. Abrir cmd en la misma ruta e introducir python off2stl.py.
3. Paso seguido, pide ingresar el nombre del archivo .off a convertir. (Presionar ENTER al terminar de escribirlo)
4. Introducir el nombre que llevará el archivo .stl. (Nuevamente presionar ENTER).
5. Buscar el archivo .stl generado en la ruta en la que se está trabajando.

Resultados:

Al programa generado por el grupo se envió el cubo anteriormente mostrado(Figura 1). Dando como resultado lo siguiente en código y en imagen.

Imagen 3: Código de la figura en STL ASCII generado por el algoritmo.

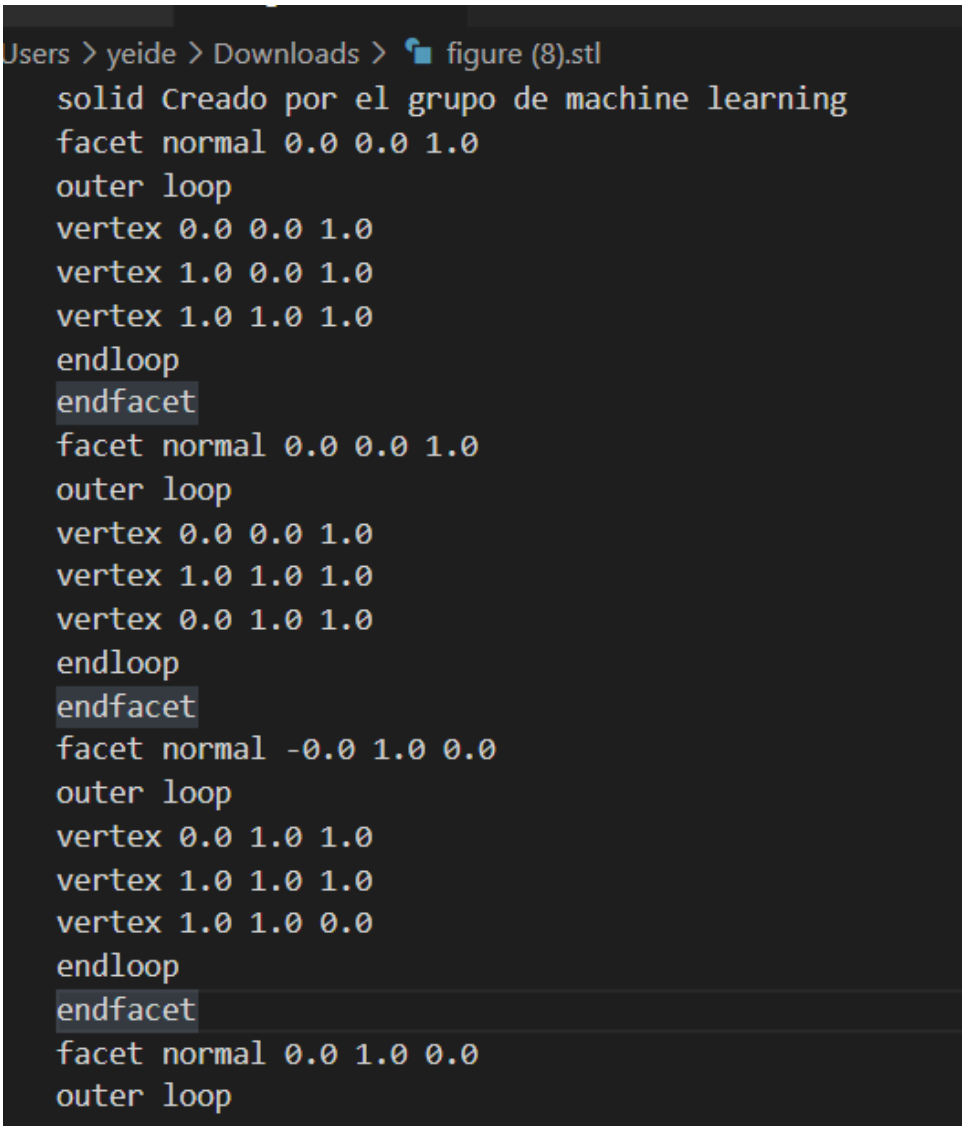
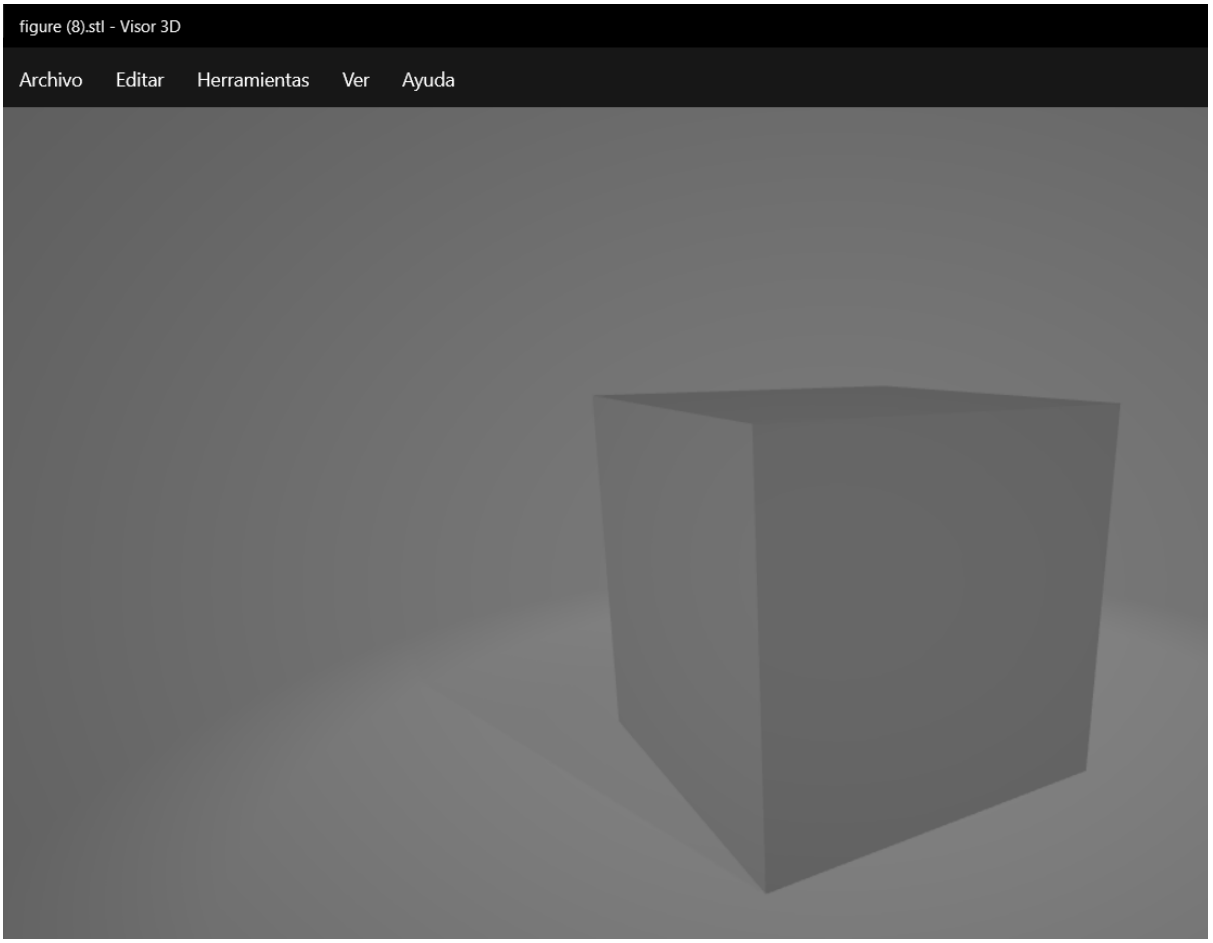


Imagen 4: Cubo en STL ASCII visualizado en un programa.



5). BIBLIOGRAFÍA

Hermoza, R., & Sipiran, I. (2018). 3D reconstruction of incomplete archaeological objects using a generative adversarial network. In *Proceedings of Computer Graphics International 2018* (pp. 5-11). https://arxiv.org/pdf/1711.06363.pdf

FileFormat (2022). Recueprado de https://docs.fileformat.com/cad/stl/

OFF Files Geomview Object File Format (2012). Recuperado de https://people.STL File

Adobe. (s.f). STL files.
https://www.adobe.com/creativecloud/file-types/image/vector/stl-file.html#stl

Cory Dickson (2020). Binary vs. ASCII STL: Choose Best File Format For 3D Printer. 3dprintingwiz.com/. Recuperado en el 2022 de la página.
https://3dprintingwiz.com/binary-vs-ascii-stl/

Joel C. Najmon, Sajjad Raeisi, Andres Tovar. (2019). Review of additive manufacturing technologies and applications in the aerospace industry. ***Additive Manufacturing for the Aerospace Industry.*** **https://doi.org/10.1016/B978-0-12-814062-8.00002-9.**
https://www.sciencedirect.com/topics/materials-science/stereolithography