Obligatorio: Inteligencia Artificial Generativa.

Proyecto: Animals with attributes CVAE

Leandro Cardoso - 166267
Felipe Schramm - 343028
Martin Rizzo - 343631

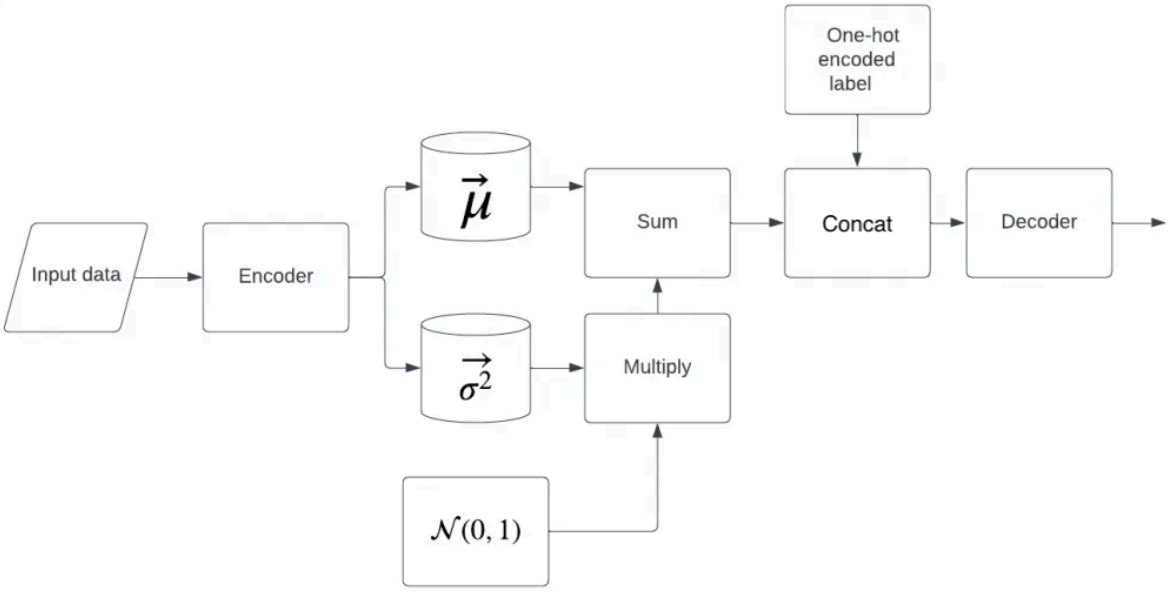
Introducción

Nuestro equipo planteó el desafío de conseguir generar imágenes de animales con una CVAE. Inicialmente propusimos utilizar el dataset Animals with attributes 2, dónde se encontraban más de 50 clases de animales y 85 atributos diferentes para cada uno de los animales.

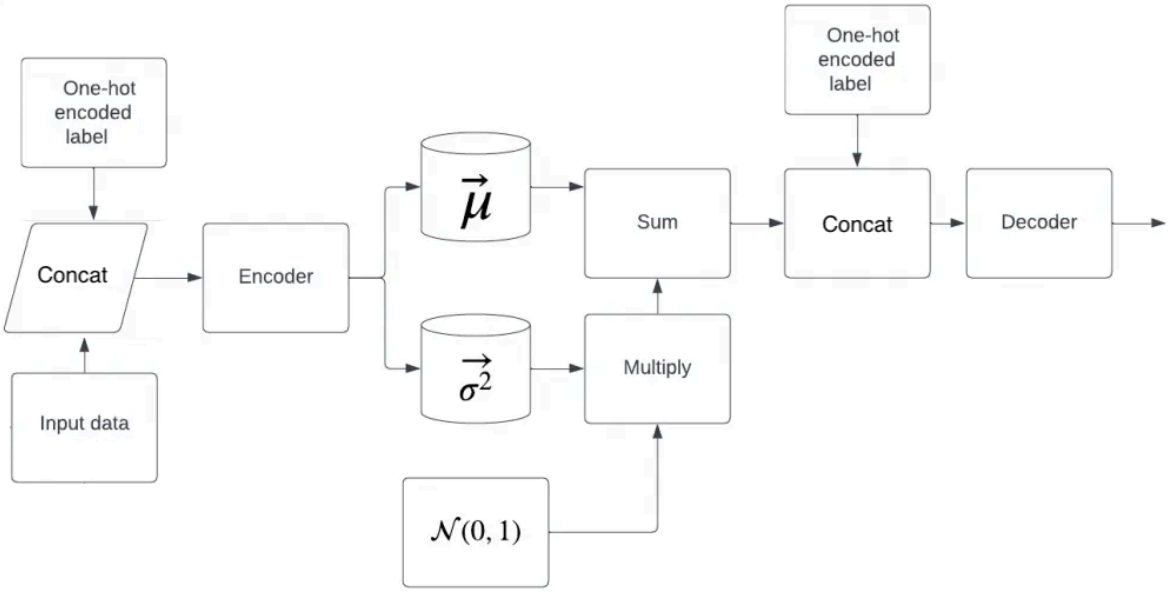
Cada clase tiene un subconjunto de los atributos asociados (Por ejemplo: De las 50 clases la única que tiene el atributo “flys” es “bat”, o “hops” los tiene “rabbit”, ”squirrel” y “hamster”. Por lo tanto, conociendo la clase del dataset se puede obtener un conjunto de atributos asociados a esa imagen. De esa forma se puede entrenar el modelo para después utilizar los atributos deseados y generar alguna de las clases con las que se corresponden estos atributos.

Un Conditional VAE (Variational Autoencoder condicional) es una variante del VAE que genera datos (como imágenes o texto) condicionados a una variable adicional, como una etiqueta o atributos, lo que permite controlar el tipo de datos generados.
Se manejaron 2 variantes:

Variante 1:



Variante 2:



Ejecución

Durante la ejecución de nuestro trabajo fuimos encontrando diferentes obstáculos que nos impidieron conseguir nuestro objetivo inicial planteado, pues nos encontramos con la dificultad de generar imágenes nítidas. Para ello, dividimos nuestro estudio en tres etapas:

1) Realizar pruebas con los lineamientos originales de nuestro planteamiento.
2) Reducción del problema a una expresión mínima.
3) Cambio de dataset con mejores resultados.

El código correspondiente a cada etapa del proyecto se encuentra en un notebook específico, el cual está nombrado siguiendo la siguiente nomenclatura: Obligatorio-iag-etapaN.

Etapa número 1

En esta etapa nos dedicamos a probar nuestro modelo de CVAE con diferentes hiperparametros y optimizaciones del modelo para procurar generar una variabilidad de imágenes basado en los atributos seleccionados.

Decidimos aprovechar esta estructura para utilizarla en el modelo, donde los atributos en un vector de tamaño n_attributes serían la entrada con los valores 0 o 1 en la posición que se encuentran en la matriz, que coinciden con sus IDs. De modo que se ingresa un 1 en las posiciones de los atributos que se querían utilizar como condición para generar y un 0 en las otras.

```
 1  antelope
 2  grizzly+bear
 3  killer+whale
 4  beaver
 5  dalmatian
 6  persian+cat
 7  horse
 8  german+shepherd
 9  blue+whale
10  siamese+cat
```

```
 1  black
 2  white
 3  blue
 4  brown
 5  gray
 6  orange
 7  red
 8  yellow
 9  patches
10  spots
```

```
Matriz de valores binarios:
[[0. 0. 0. ... 0. 0. 0.]
 [1. 0. 0. ... 1. 0. 0.]
 [1. 1. 0. ... 0. 0. 0.]
 ...
 [1. 1. 0. ... 1. 1. 0.]
 [1. 1. 0. ... 0. 0. 1.]
 [0. 1. 1. ... 0. 0. 1.]]
```

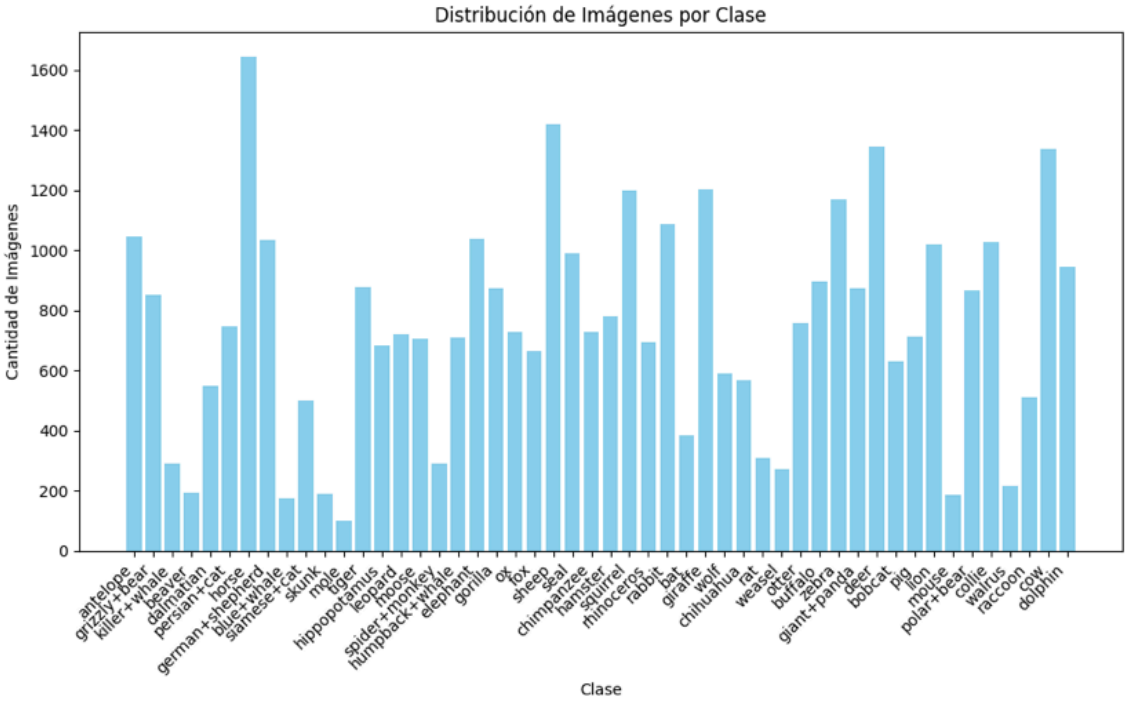
Dado que la cantidad de clases y atributos que tiene el dataset son muy extensas para el modelo que se podía implementar dentro del alcance del obligatorio, se decidió tomar un subconjunto de las clases y de los atributos para hacer el problema más sencillo.

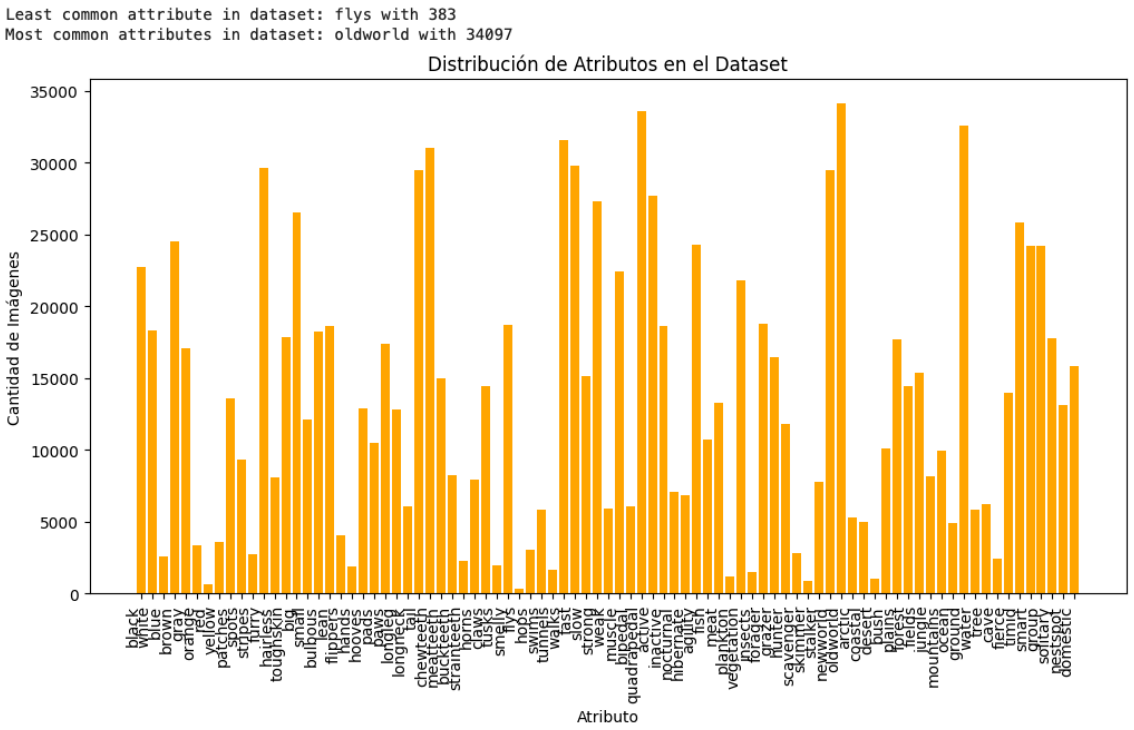
```
Clases seleccionadas: {7: 'horse', 11: 'skunk', 23: 'sheep', 30: 'bat'}
Atributos seleccionados: {1: 'black', 2: 'white', 16: 'small', 26: 'tail', 40: 'fast', 42: 'strong'}
Matriz filtrada:
 [[1 1 0 1 1 1]
 [1 1 1 1 1 0]
 [1 1 0 0 0 0]
 [1 0 1 0 1 0]]
Nuevos índices para clases: {7: 0, 11: 1, 23: 2, 30: 3}
Nuevos índices para atributos: {1: 0, 2: 1, 16: 2, 26: 3, 40: 4, 42: 5}

Clases seleccionadas (reindexadas): {0: 'horse', 1: 'skunk', 2: 'sheep', 3: 'bat'}
Atributos seleccionados (reindexados): {0: 'black', 1: 'white', 2: 'small', 3: 'tail', 4: 'fast', 5: 'strong'}
Matriz filtrada:
 [[1 1 0 1 1 1]
 [1 1 1 1 1 0]
 [1 1 0 0 0 0]
 [1 0 1 0 1 0]]
```

Se aplicaron algunas transformaciones para obtener una matriz con la misma estructura pero con las clases y atributos deseados de todos los que se disponen.

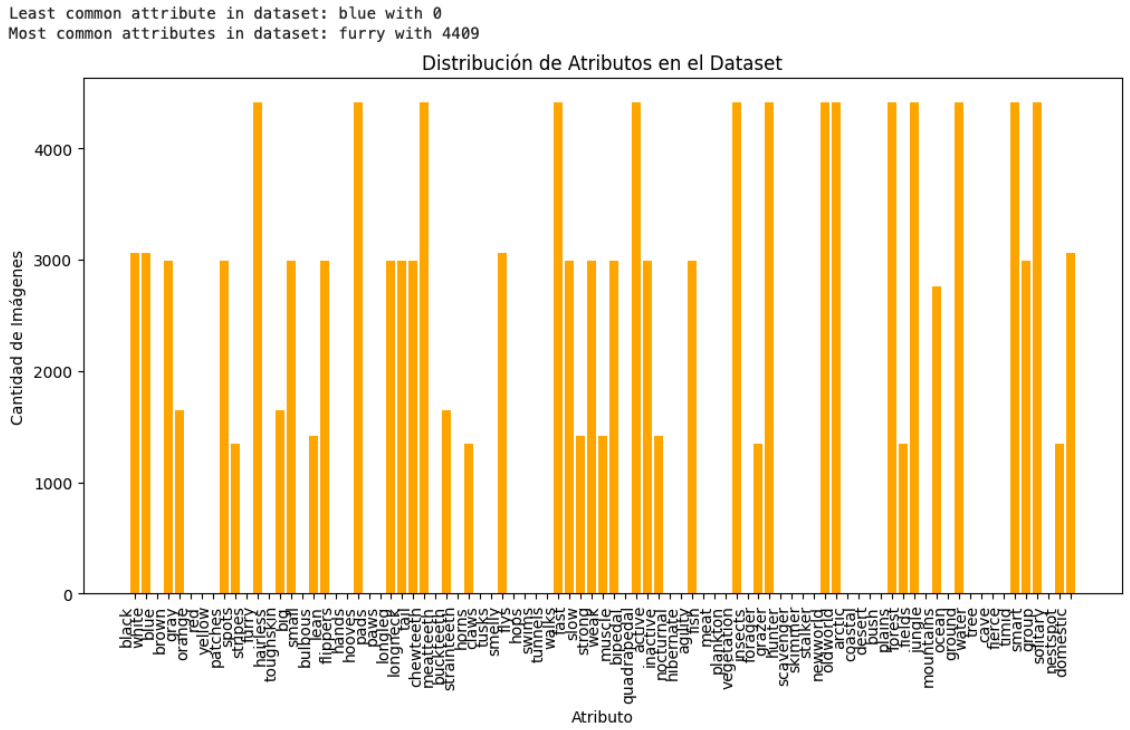
El criterio para seleccionar las clases y atributos para entrenar el modelo se basó en la disponibilidad de imágenes de cada clase y el balance de atributos y clases.

```
Least common class in dataset: mole with 100
Most common class in dataset: horse with 1645
```

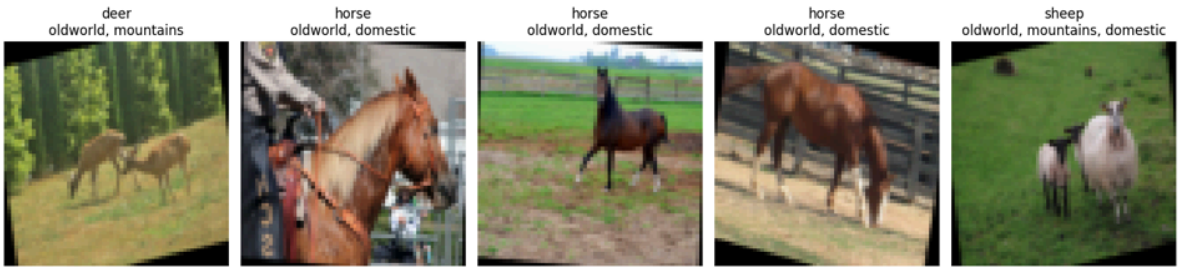




Dentro de las clases seleccionadas, elegir los atributos más frecuentes entre dichas clases.



Las transformaciones aplicadas a las imágenes fueron muy sencillas. Leves rotaciones de hasta 15ª, espejado y resizing de 64x64 junto con normalizado entre 0 y 1 fueron las únicas transformaciones aplicadas a las imágenes. Se evitaron transformaciones que pudieran contradecir las condiciones o características que identifican a cierta clase como el color.

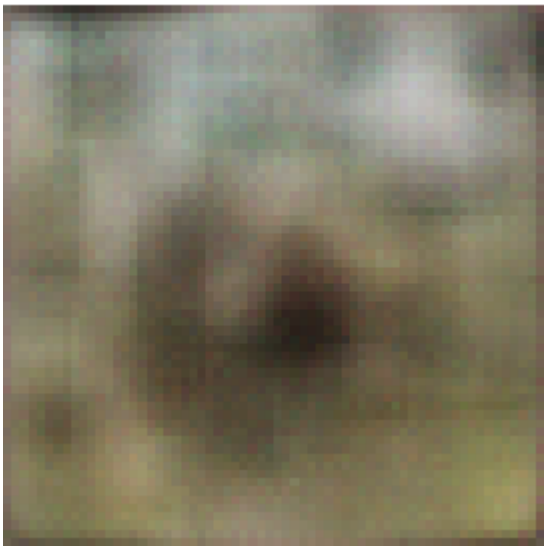


En esta etapa se utilizó únicamente la variante 1 (sin usar los atributos en la entrada del encoder)

Teniendo en cuenta que lo que se pretende generar son imágenes, el modelo utilizado fue implementado con convoluciones en el encoder y de convoluciones en el decoder. En la entrada del encoder se concatena la lista de atributos en forma de tensores fila al tensor z y se pasa por una capa densa antes de aplicarles la deconvolución.

Se variaron distintos hiperparametros como la cantidad de canales en cada convolución, la dimensión del espacio latente, la cantidad de capas convolucionales, distintas funciones de activación como leaky ReLU en lugar de ReLU o sigmoidea en lugar de tangente hiperbólica. Los resultados luego de entrenar 20 épocas no cambiaron. Se probó bajar la cantidad de clases y de atributos para simplificar aún más el problema y reducir la variabilidad en los datos pero los resultados siguieron siendo los mismos.

- **Dimensión del Espacio Latente**: 400. Se probaron otros tamaños de espacio latente, tanto mayores como menores, pero no variaron el resultado en inferencia.
- **Tamaño de Imagen**: 64x64 píxeles.
- **Épocas**: 20. Se evaluaron diferentes cantidades de épocas, pero no se observaron mejoras en la calidad de las imágenes generadas al aumentar este número.
- **Recon Loss**: MSE.
- **Optimizador**: Adam con una tasa de aprendizaje (lr) de 0.0001.

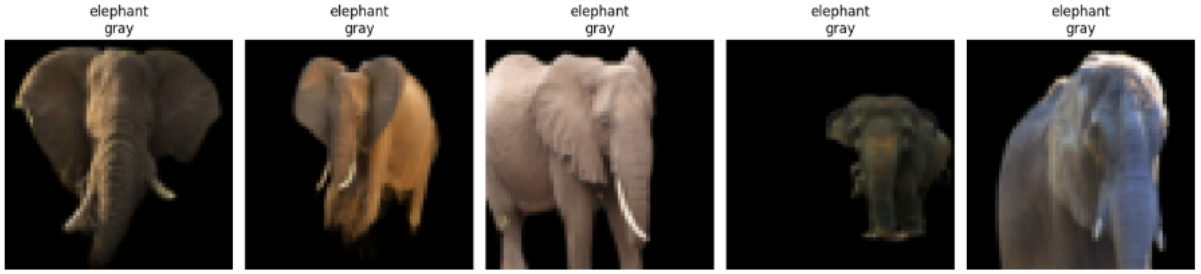


Etapa número 2

En esta etapa nos planteamos en reducir el problema a su mínima expresión. Era necesario verificar sí nuestro modelo era lo suficiente para poder procesar las imágenes de nuestro dataset.

En primer lugar, seleccionamos un único animal y una única clase para intentar generar una imagen de un elefante. Luego, eliminamos todos los fondos de las imágenes y seleccionamos de forma manual posiciones del animal que sean similares.

Nuestro dataset terminó con 205 imágenes sin fondo con el animal de frente como en las siguientes imágenes:



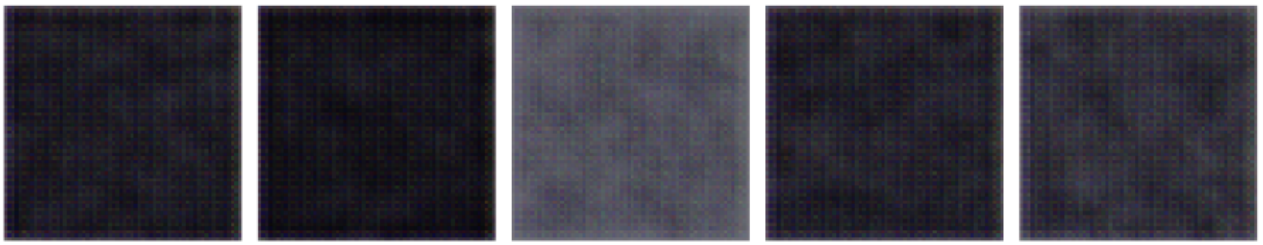
Con estas imágenes intentamos realizar diferentes ejecuciones con nuestro modelo contando con diferentes hiper parámetros y regularizaciones. Obteniendo resultados como los siguientes:

a) **CVAE vanila**

Hiperparámetros:

lr = 1e-4, img_size = 64, num_hidden = 512, num_classes = 1, epochs = 30

Resultados:

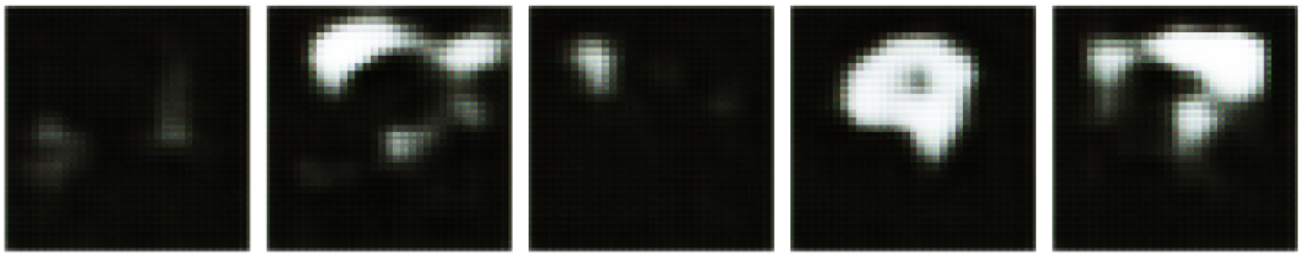


b) **CVAE w/improvements: conv2d + dropout + batchnorm**

Hiperparámetros:

lr = 1e-4, img_size = 64, latent_dim = 512, num_classes = 1, epochs = 30

Resultados:

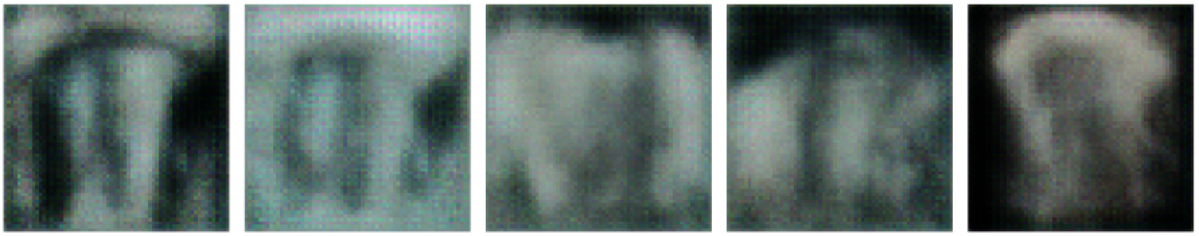


c) VAE w/improvements: conv2d + batchnorm

Hiperparámetros

lr = 1e-4, num_hidden = 512, z_dim = 15, epochs = 30

Resultados:



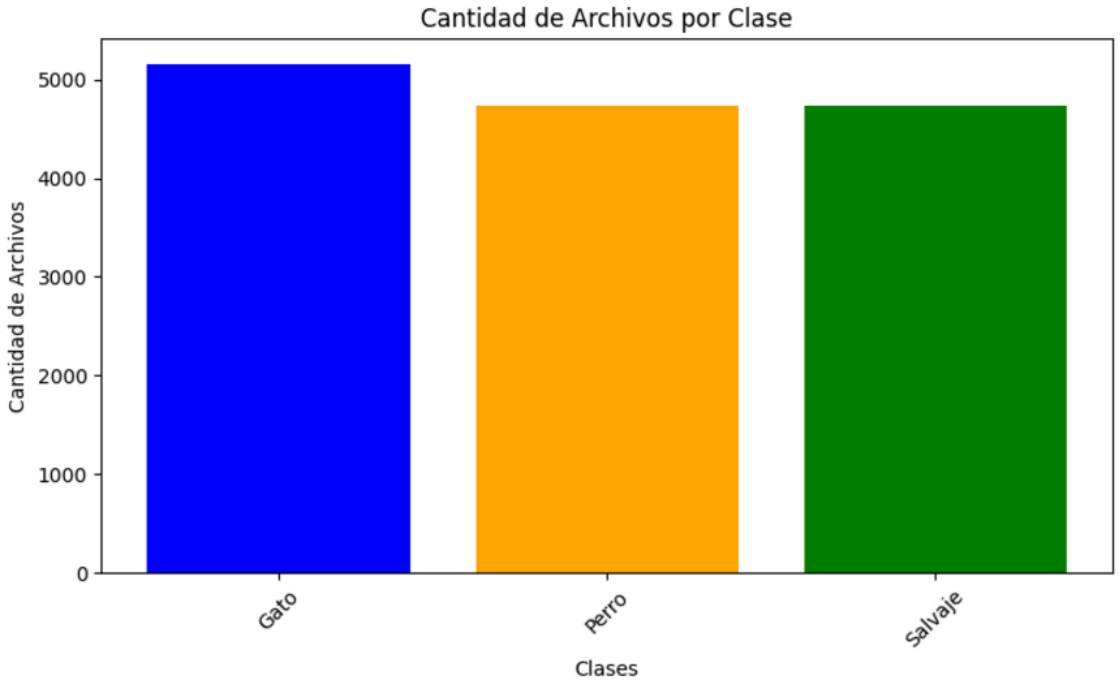
Los resultados no fueron suficientes y nada alentadores. Reducimos el problema a su mínima expresión e inclusive decidimos migrar hacia un modelo VAE para evitar sobre exigir a nuestro modelo con una matriz de atributos.

Por estos motivos, llegamos a la conclusión que nuestro dataset era demasiado complejo como para ser procesado por una CVAE. Las diferentes posiciones de los animales, más los fondos variables y la gran cantidad de variabilidad de clases y atributos, hacía imposible para nuestro modelo generar una imagen que se aproxime a la realidad.

Etapa número 3

Nuestro nuevo dataset, “animal faces”, que utilizamos en esta etapa contiene imágenes de tres clases distintas: gatos, perros y animales salvajes. Este conjunto de datos es ideal debido a que se trata de caras de animales que no tienen fondos que complican la generación. Lo que favorece a explorar la variabilidad en las características faciales de diferentes especies.

La distribución de clases en el dataset es equilibrada, lo que facilita la evaluación del rendimiento del modelo en cada categoría. A continuación, se presenta una imagen que ilustra la distribución de clases dentro del conjunto de datos.

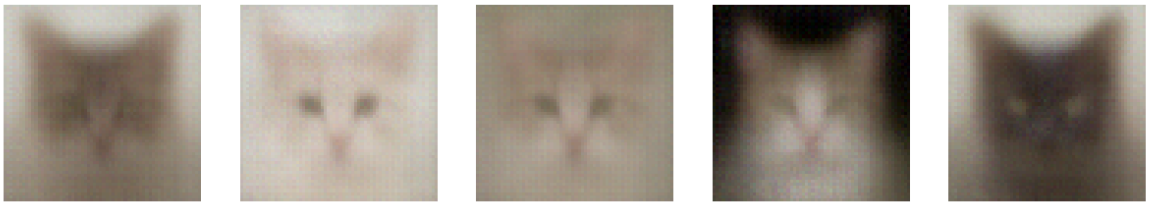


A continuación, se muestran ejemplos representativos de cada clase:

Ahora pasaremos a describir los casos que se probaron:

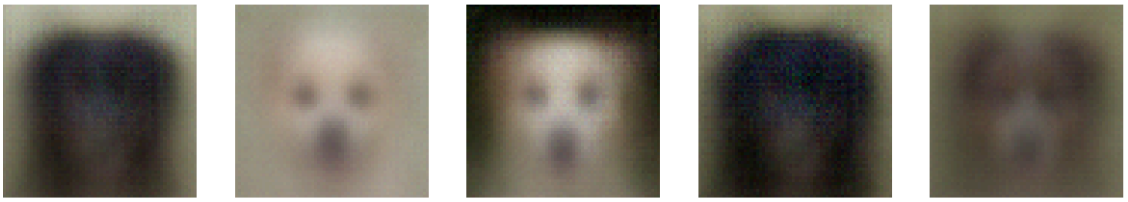
a) **Generación de Imágenes de gatos con VAE:**

Se comenzó utilizando una VAE entrenado exclusivamente con imágenes de gatos. Los resultados fueron prometedores, logrando generar imágenes que eran visualmente aceptables y representativas de la clase.



b) Prueba con Imágenes de Perros:

Posteriormente, se aplicó el mismo modelo VAE al dataset de imágenes de perros. Al igual que con los gatos, el modelo fue capaz de generar imágenes con calidad similar, lo que sugiere que la VAE es efectiva para ambas clases.



c) Implementación de CVAE:

A continuación, se probó un CVAE, que permite generar imágenes basadas en la condición de la clase (gato o perro). Este enfoque también funcionó bien, generando imágenes que correspondían a la clase especificada.

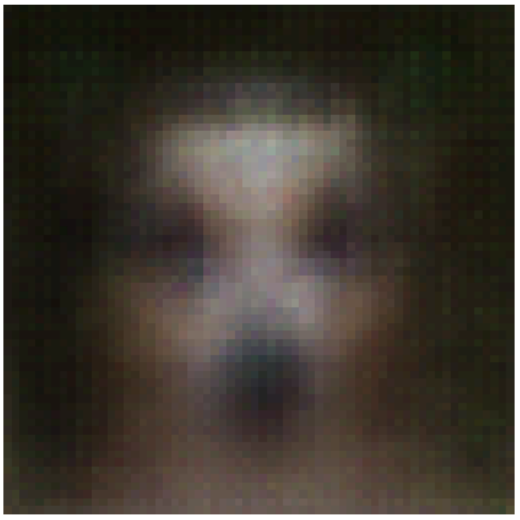
d) Optimización de la Calidad de las Imágenes:

Para mejorar la nitidez de las imágenes generadas, se experimentó con diferentes técnicas de regularización:

Dropout:

Imagen generada de un gato

Imagen generada de un perro



Dropout + Batch Normalization:

Imagen generada de un gato

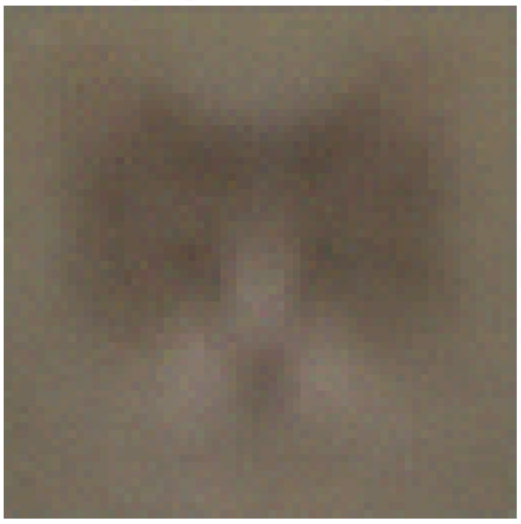
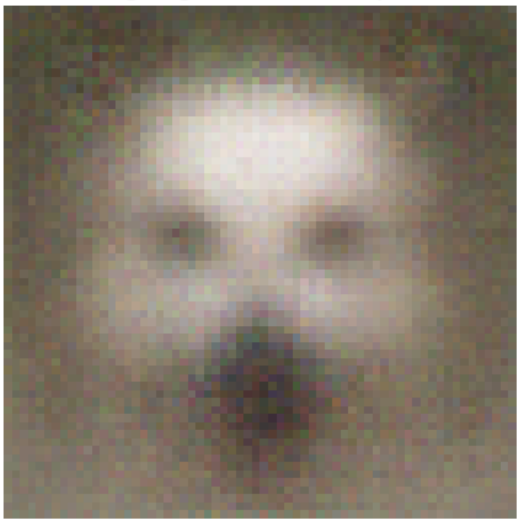


Imagen generada de un perro



Batch Normalization únicamente:

Sin embargo, a simple vista no se observaron mejoras significativas en la calidad de las imágenes generadas como para justificar su inclusión en un modelo final.

e) Expansión del Modelo a Tres Clases:

Finalmente, se decidió utilizar el modelo base que había funcionado bien para gatos y perros y agregar una tercera clase: animales salvajes. Este modelo con las tres clases demostró ser efectivo, generando imágenes que podían distinguirse claramente como un gato, un perro y un animal salvaje.

Para asegurar condiciones iguales en todas las pruebas y facilitar una comparación justa entre los modelos, se mantuvieron constantes los siguientes hiper parámetros, los cuales se seleccionaron basándose en pruebas previas con VAEs que mostraron los mejores resultados visuales:

- **Batch Normalization**: Se utilizó para estabilizar el aprendizaje.
- **Dimensión del Espacio Latente**: 128. Se probaron otros tamaños de espacio latente, tanto mayores como menores, pero estos resultaron en una disminución de la calidad visual de las imágenes generadas.
- **Tamaño de Imagen**: 64x64 píxeles.
- **Épocas**: 20. Se evaluaron diferentes cantidades de épocas, pero no se observaron mejoras significativas en la calidad de las imágenes generadas al aumentar o disminuir este número.
- **Optimizador**: Adam con una tasa de aprendizaje (lr) de 0.001.

Se probó con los mismos hiper parámetros, pero con la variante 1, quitando complejidad al encoder. Los resultados son similares a la implementación anterior, pudiendo distinguirse de forma clara entre los tres tipos de animales generados.

Conclusiones

Durante nuestro proyecto enfrentamos numerosos desafíos al intentar generar imágenes a partir de una CVAE. Experimentamos con diversas tácticas para superar los obstáculos y producir imágenes nítidas con nuestro conjunto de datos original, pero sin éxito.

Para llegar a esta conclusión, probamos diferentes modelos y modificamos las imágenes de varias maneras. A pesar de nuestros esfuerzos, no pudimos lograr los resultados deseados.

Esta limitación nos llevó a cambiar nuestro conjunto de datos original para buscar resultados más alentadores.

Las variantes manejadas durante las pruebas tuvieron un comportamiento similar. Posiblemente, el modelo maneja pocas condiciones o genera imágenes muy sencillas como para que estas variaciones lleguen a influir en su desempeño.

CVAE es un modelo interesante aunque limitado dada su arquitectura. Es un modelo complejo aunque no suficiente para problemas donde hay una variabilidad muy alta sobre la naturaleza de las imágenes.

Trabajo futuro

Además de abordar aspectos relacionados con la nitidez y la calidad de las imágenes, una adición futura podría ser la incorporación de un *prompt* y un mecanismo que permita decodificar y los atributos deseados en el texto para la entrada de la CVAE, obteniendo así un input más natural y flexible.

En cuanto a la capacidad de manejar múltiples atributos, se podría investigar otras formas de codificar el input con las condiciones para el modelo. En este caso se utilizó únicamente la versión que concatena los inputs como vector junto a los del vector de entrada (z) al decoder, formando un vector de largo z_dim + n_attributes para la entrada a la capa densa previa a las deconvoluciones. Otra posible forma de computar los atributos como entrada del modelo podría haber sido, por ejemplo, pasar el vector de atributos por una capa densa tal que a la salida se obtenga un vector de la misma dimensión que la entrada z_dim y sumarlo a esta.

Bibliografía

- Rrebirrth. (n.d.). *Animals with Attributes 2* [Dataset]. Kaggle. Recuperado de https://www.kaggle.com/datasets/rrebirrth/animals-with-attributes-2

- Mvd, A. (n.d.). *Animal Faces Data* [Dataset]. Kaggle. Recuperado de https://www.kaggle.com/datasets/andrewmvd/animal-faces/data

- Google Drive. (n.d.). *Elefantes_frente* [Dataset]. Recuperado de https://drive.google.com/file/d/11VheLuf9fkqS2aj5LHECQFYX6sEmAgG0/view?usp=sharing

- Mankar, V. (n.d.). *Asian vs African Elephant Image Classification* [Dataset]. Kaggle. Recuperado de https://www.kaggle.com/datasets/vivmankar/asian-vs-african-elephant-image-classification

- Sofeikov, K. (2022, abril 12). *Implementing Conditional Variational Autoencoders (CVAE) from Scratch*. Medium. Recuperado de https://medium.com/@sofeikov/implementing-conditional-variational-auto-encoders-cvae-from-scratch-29fcbb8cb08f