Segmentación de órganos

Camilo Esteban Zambrano Pereira
czambrano@unal.edu.co
Técnicas de Inteligencia Artificial
Universidad Nacional de Colombia.
Bogotá. Colombia.

I. Resumen

EN el siguiente documento se realiza el análisis de un dataset tomado de una competencia de Kaggle y se explica pix2pix, la arquitectura a implementar para solucionar el problema.

II. Introducción

Al rededor de la mitad de los pacientes de cáncer gastro-intestinal son tratados con radio terapia, generalmente en sesiones de 10 a 15 minutos al día durante 1 a 6 semanas. Para ello, se aplican grandes dosis de rayos X direccionados al tumor evitando los intestinos y el estomago, debido a esto, los oncologos necesitan saber la posición de los mismos para no cometer errores, este proceso se realiza de forma manual, lo que puede prolongar el procedimiento entre 15 minutos y una hora al día, debido a que los órganos cambian de posición para cada sesión, es necesario repetir este procedimiento varias veces realentizando los tratamientos. A partir de esto, la Universidad de Winsconsin-Madison plateo una competencia en Kaggle, que busca un modelo de deep learning capaz de tomar imágenes de tomografías abdominales y segmentar los intestinos y el estomago. A continuación se hablara de la posible solución a este problema.

III. Marco Teórico

III-A. Autoencoders

Un autoencoder es un tipo de red neuronal utilizado para elaborar codificaciones de datos eficientes, por lo general se utilizan en aplicaciones no supervisadas para reducir la dimensionalidad. Esta arquitectura busca imponer un cuello de botella con el fin de obtener una representación compacta de los datos. El autoencoder se compone de dos redes neuronales, la primera (codificador) se encarga de transformar los datos a un espacio de menor dimensionalidad, tambien conocido como espacio latente; la segunda (decodificador) realiza el proceso inverso, toma los datos del espacio latente y los lleva a su espacio original. Para lograr esto, se utiliza la siguiente función de perdida:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} ||x_i - \widetilde{x}_i||^2 \tag{1}$$

Donde x_i es la observación original y $\widetilde{x}_i$ esta dado por:

$$\widetilde{x}_i = decorder(codificador(x_i)) \tag{2}$$

Es decir, la salida del sistema.

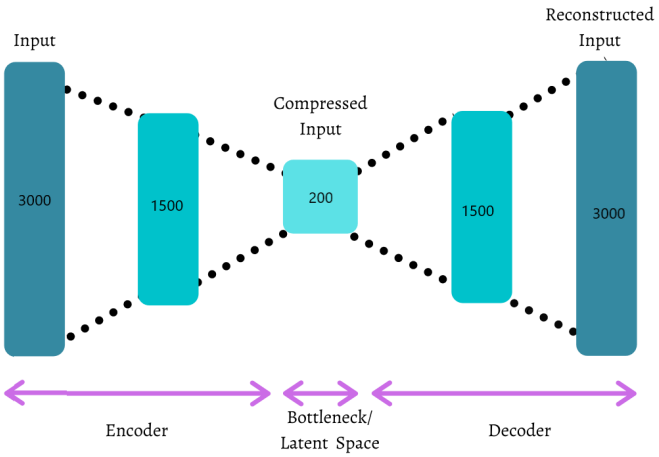

Figura 1: Autoencoder.

El concepto del autoencoder es sencillo y es ampliamente utilizado, algunas de sus aplicaciones son:

- Reducción de dimensionalidad.
- Extracción de características.
- Reducción de ruido en imágenes.
- Compresión de imágenes.
- Búsqueda de imágenes.
- Detección de anomalías (por ejemplo transacciones fraudulentas).
- Predicción de valores faltantes.

III-B. U-Net

U-Net es una red neuronal convolucional que se combina con un autoencoder, originalmente fue diseñada para propósitos de segmentación en imágenes biomédicas. Se compone por la aplicación repetida de dos capas de convolución 3x3 seguida por max pooling de 2x2 con una reducción de dos píxeles; para cada reducción se duplican el número de filtros aplicados por las capas convolucionales hasta pasar de imágenes de 570x570 píxeles a 32x32. Posteriormente, se realiza el proceso inverso, se aplica una capa deconvolucional que aumenta el tamaño de la imagen en dos píxeles que adicionalmente esta conectada a su capa equivalente en el codificador; luego se añaden dos capas de convolución de 3x3 y se repite el patrón hasta llegar nuevamente al tamaño original de la imagen:

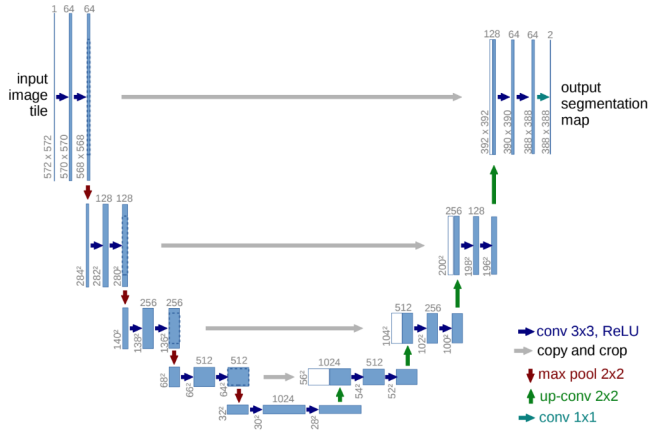

Figura 2: U-Net.

Algunas de sus aplicaciones son:

- Segmentación de imagenes.
- Predicción de pixeles faltantes.
- Predicción de estructura de proteínas.

III-C. Redes generativas adversarias (GAN)

Las GAN's son un tipo de red neuronal muy ingeniosa que se desarrollo en la ultima década, su principio de funcionamiento consiste en enfrentar dos redes neuronales que compiten en un juego de suma cero (las ganancias de uno se compensan con las perdidas del otro), de esta manera, la red generativa se encargará de producir muestras de algún tipo de dato, pueden ser imágenes, texto, sonidos, entre otros. Evidentemente los resultados serán malos, por lo que la segunda red neuronal (discriminador) se encargará de tomar dichos resultados y evaluarlos. Si la evaluación es incorrecta la red discriminatoria le comunicará a la generadora la cercanía a los resultados esperados para de esta forma, actualizar los pesos y volverlo a intentar. Para que la red generativa logre engañar a la red discriminatoria son necesarios demasiados intentos, dependiendo del problema pueden ser cientos, miles o millones, debido a esto, los tiempos de entrenamiento de este tipo de red neuronal son elevados.

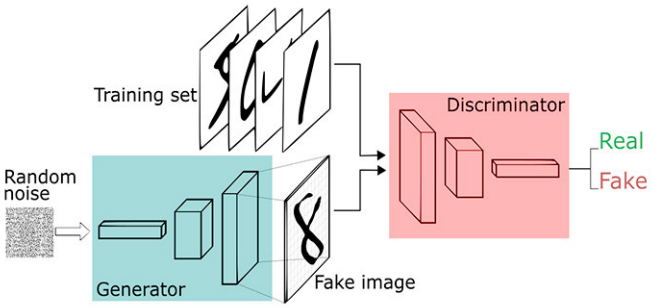

Figura 3: Arquitectura de una GAN.

Algunas de las aplicaciones de las GAN's son:

- Generación de caras humanas.
- Generación de personajes animados.
- Fotos de caras a emojis.
- Súper resolución.
- Texto a imagen (Ejemplo Dall-E 2)
- Transferencia de estilo.

Figura 4: Caras generadas por una GAN.

III-D. Pix2pix

Pix2pix combina todas las arquitecturas tratadas anteriormente para generar una cGAN (Red generativa adversaria condicional), este modelo tiene como finalidad realizar traducciones de imagen a imagen. Su arquitectura consiste en un generador basado en U-Net y un discriminador representado por una PatchGAN convolucional. En esta ultima no se va a entrar en detalle, pero básicamente consiste en una red convolucional que se encargará de comparar la imagen original con la entregada por la red tipo U-Net.

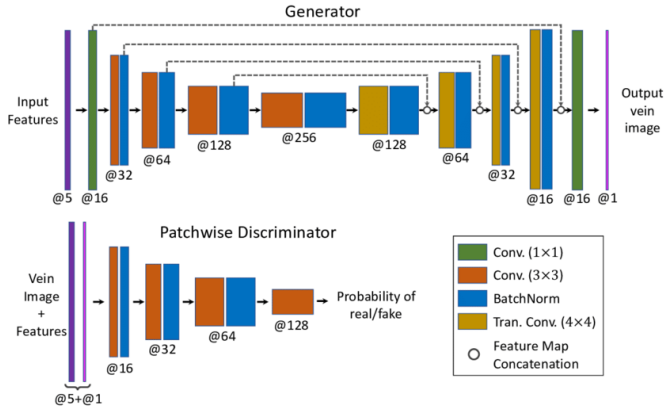

Figura 5: Arquitectura de Pix2pix.

IV. Descripción del problema y análisis de los datos

El problema fue tomado de una competencia de Kaggle llamada ÜW-Madison GI Tract Image Segmentation"que busca que busca segmentar el intestino grueso, intestino delgado, y el estomago en imágenes de tomografías abdominales, esto con el fin de ayudar a los médicos a determinar el tamaño de dichos órganos y facilitar la decisión del tamaño de las dosis en el tratamiento de cáncer gastro-intestinal. Para ello, se entrega un dataset que consta de 38.496 imágenes con sus respectivas mascaras para cada órgano con la siguiente proporción de datos, repartidas en 85 pacientes:

	Segmentadas	Sin segmentar
Estomago	8.627	29.869
Intestino grueso	14.085	24.411
Intestino delgado	11.201	27.295
Total	16.590	21.906

Las imágenes tienen 4 tamaños:

- 266px x 266px
- 360px x 310px
- 276px x 276px
- 432px x 432px

Y siguen las siguientes distribuciones:

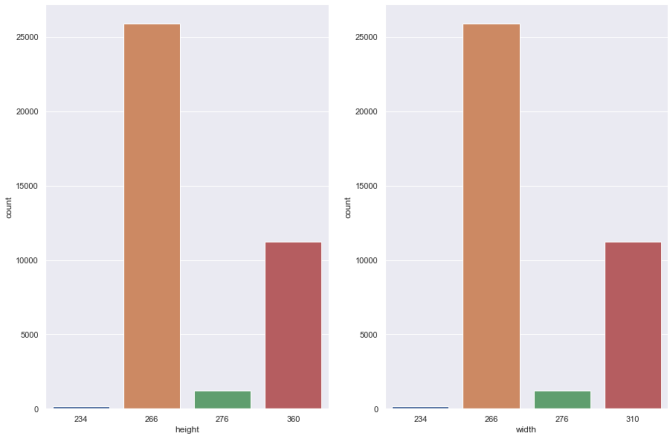

Figura 6: Tamaño de las imágenes.

Las máscaras se encuentran codificadas en RLE-encoded, codificación que indica la posición y longitud de los contornos de la figura. Realizando la codificación y destinando un canal de color para cada órgano (R = Intestino grueso, G = Intestino delgado, B = Estomago), de esta forma, se obtienen las siguientes mascaras:

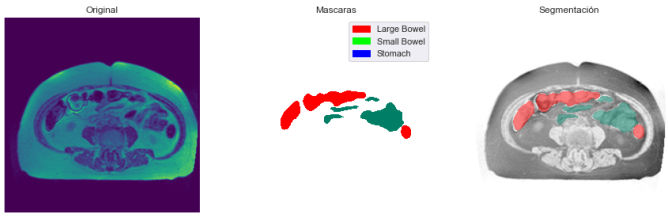

Figura 7: Imagen de ejemplo segmentada.

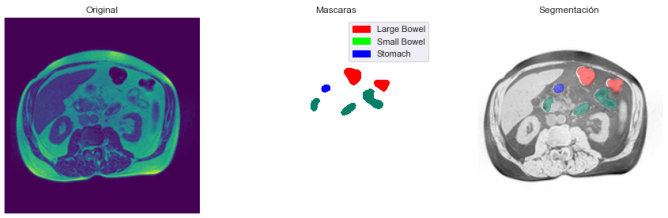

Figura 8: Imagen de ejemplo segmentada.

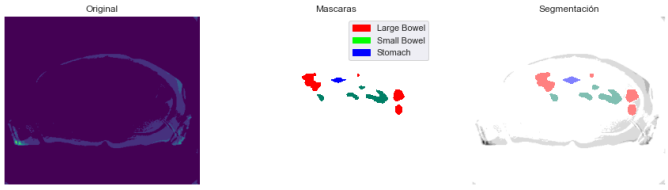

Figura 9: Imagen de ejemplo segmentada.

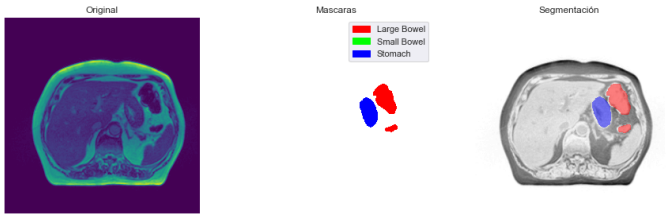

Figura 10: Imagen de ejemplo segmentada.

Como se puede ver, algunas imágenes no presentan ninguna tonalidad pero están segmentadas, esto presentara un problema para el modelo.

V. Propuesta de solución

Como propuesta de solución se plantea la arquitectura pix2pix, donde como imagen de entrada se recibe un arreglo de 256x256 px en escala de grises, este es el tamaño óptimo para pix2pix y no es lejano a los originales de las imágenes. A la salida se espera una imagen en RGB y la mascara de cada órgano representa cada canal de color. La red neuronal discriminadora tomará las mascaras originales y las comparara con las generadas para de esta manera actualizar los pesos.

VI. Entrenamiento

Para empezar, se parte el dataset en 80 % para entrenar y 20 % para validar, sin embargo, no se mezcla el dataset, esto se debe a que las imágenes de cada caso son muy parecidas entre si, por lo que mezclar el dataset puede generar que imágenes muy similares queden en entrenamiento y prueba, mostrando resultados erróneos. Por otro lado, las imágenes son normalizadas en valores entre -1 y 1, intervalo recomendado por los autores de la arquitectura.

Otro problema evidente es el sesgo que se presenta en el dataset, pues el 43.09 % de las imágenes están segmentadas, particularmente para cada órgano este porcentaje disminuye, de forma que el modelo tendrá una mayor tendencia a entregar como resultado imágenes en negro. Para solucionar esto se pueden tomar varios enfoques, el primero es entrenar únicamente con las imágenes segmentadas y posteriormente observar los resultados con el dataset de entrenamiento. Otra solución es eliminar la mitad de las imágenes sin segmentar, dado que son muy similares entre si, es posible generar un dataset balanceado sin afectar el rendimiento del modelo de forma negativa.

VII. Resultados

Obtener buenos resultados en este problema es complicado debido al alto volumen de datos para entrenar y las limitaciones computacionales (una sola RTX 3060ti trabajando) puede tardar días o semanas en completar el entrenamiento. Debido a esto, los resultados a continuación son extraídos en etapas iniciales (después de 24 horas de entrenamiento), por lo que las máscaras no son correctas aunque son ligeramente cercanas a las originales, además, se puede apreciar que el modelo entiende que únicamente se usan 3 colores en las imágenes resultantes.

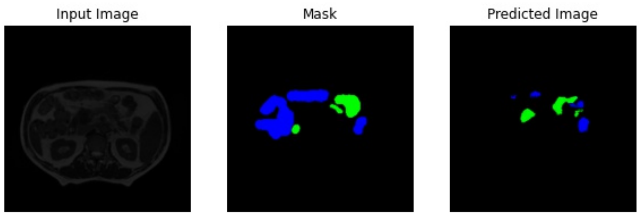

Figura 11: Predicción del modelo.

VIII. Conclusiones

Se pudo comprender e implementar la arquitectura pix2pix, observando resultados decentes en un tiempo de entrenamiento relativamente corto, además, se pudieron analizar diferentes modelos muy utilizados en la actualidad que pueden ser utilizados en una gran variedad de aplicaciones.