# Procesamiento de imágenes utilizando Deep learning y Carla Simulator

Gatica, Isais
email: isaiasgatica1@gmail.com
Saez, Lautaro Andres
email: lautaroandressaez@gmail.com

***Resumen*—En este informe se presenta la implementación de una red Pix2Pix capaz de pasar de una imagen RGB a una imagen segmentada donde cada color indica un objeto diferente. Se comenzara desde cero obteniendo el dataset mediante el simulador CARLA para luego poder entrenar la red con una gran variedad de imagenes con diferente clima y vegetación. Finalmente se probara con imagenes reales para analizar la viavilidad de entrenar una red con un simulador y luego ingresarla en el mundo real.**

## I. Introducción

En los ultimos años, debido a la necesidad de resolver cada vez mas complejos y la posibilidad de utilizar GPU's las redes neuronales han cobrado gran importancia en nuetras vidas, y ya son parte del dia a dia, cosas tan simples como los filtros de Instagram basan su reconocimiento facial en redes neuronales.

Las redes permiten resolver problemas que van desde la clasificación de imagenes hasta la conducción autonoma, claro esta que para esto existen diferentes arquitecturas las cuales son expecificas para determinadas tareas, como el procesamiento de imagenes (CNN, GAN, YOLO), procesamiento de texto (RNN), procesamiento de datos (NN).

La desventaja principal de estos modelos supervisados es la necesidad de una dataset muy bien estructurado y para el cual es necesario el trabajo humano, esta es la tarea mas importante y que si esta mal realizada el red no tendra un desempeño correcto.

## II. Marco Teoríco

En esta sección se hara un breve marco teoríco de los conceptos basicos que se utilizaran en el proyecto.

### *II-A. Deep Learning*

Las redes neuronal son muy utilizadas en el campo de la minería de datos, debido a su gran versatilidad. Dentro de este mundo existen 2 grandes tipos de entrenamientos para los supervisados que dada una entrada $X$ se conoce la salida deseada $Y$ y otros casos los algoritmos no supervisados como los modelos de Q-learning o los mapas autoorganizados (SOM). En este proyecto se opto por un entrenamiento supervisado.

Para el modelado de la red neuronal se utilizó un modelo llamado Pix2Pix [4]. El cual pertime dada una imagen de entrada $X$ obtener una imagen de salida $Y$.

Este modelo consta de 2 redes separadas, la primera se denominada el generador Fig.1 el cual deberá aprender a segmentar la imagen semanticamente. Y la segunda se llama el discriminador Fig.2 este se encarga de correguir al generador aprendiendo cuando una salida es realista". Llamaremos salida realista.[a] aquella imagen que cumpla el objetivo esperado, en este caso sería una imagen segmentada de coherente.


Figura 1. Esquema de la red generativa.

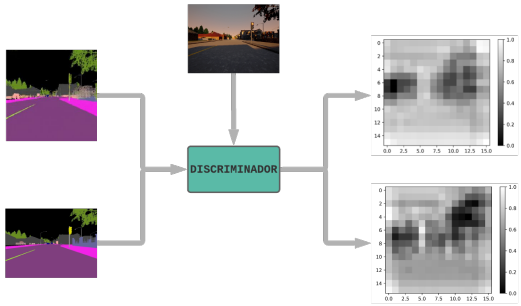


Figura 2. Esquema de la red discriminativa.

*II-A1. Generador:* El generador cuenta con una arquitetura denomina U-net [3], la cual tiene la cual posee una muy buena eficiencia debido a los bypass que existen entra las capas de convolución y las capas de deconvulición. Con la finalidad de comprender de forma mas sencilla se puede observar en la Fig.3 dicha arquitectura.

El error del generador es cuantizado por la red discriminativa.

*II-A2. Discriminador:* El discriminador posee una arquitectura mucho mas sencilla, la cual consta de 6 capas de convolución. Esto se debe a que la salida representa que tan realista es un segmento de la imagen de entrada. El error del discriminador se cuantiza mediante una entropía cruzada. Ya que se propone que siempre que le entre una imagen del generador la salida debe ser una matriz de 0, ya que esta imagen no es del dataset, mientras que cuando la entrada sea una imagen del dataset el resultado debe ser un 1.

Estodesemboca en una pelea entre en generador ya que debe

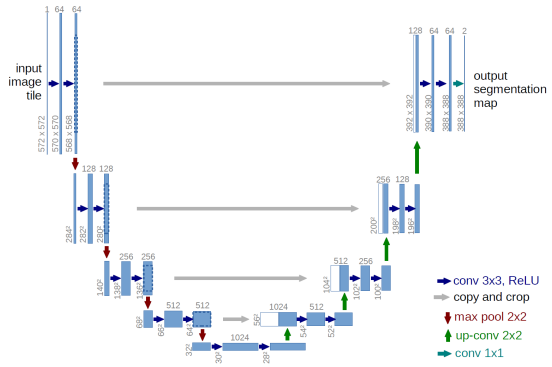


Figura 3. Arquitectura de una U-net.

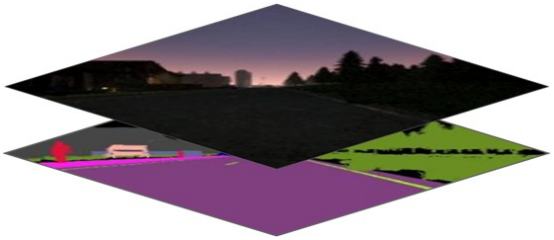

Figura 4. Visualización del stack previo al random jitter.

aprender a engañar al discriminador y el discriminador que debe lograr siempre reconocer que la imagen $\hat{Y}$ es totalmente falsa, ya que es una creación de la red generativa.

### *II-B. Preporcesado de los datos*

Un paso fundamental para trabajar con algoritmos de imagenes es pasar del dominio $[0; 255]$ a un dominio mas acotado, para este trabajo se utilizo el domino de $[-1; 1]$. De esta forma se logra una mayor eficiencia computacional.

Debido al tamaño de las imagenes es recomendable hacer un resize a dimensiones mas pequeñas para disminuir el uso de RAM cuando son cargadas en memoria y el uso de GPU al realizar las convoluciones. En este caso las imagenes de entrada y de salida son redimensionadas a $256x256$.

*II-B1. Random jitter:* El *random jitter* es un proceso por en el cual se busca realizar un aumentado del dataset, esto disminuye la posibilidad de experimentar **overfitting**. Este proceso cuenta de 2 pasos:

Crop Se realiza un resize a $286x286$ para luego realizar un recorte a $25x256$.

Flip De forma aletoria con probabilidad $0{,}5$ se realiza un rotación de $180°$.

Debido a que es utiliza en algoritmos supervisados es necesario aplicar el *random jitter* tanto a la imagen de entrada como a la de salida, para ello se hace un stack de las mismas como se observa en la Fig.4.

En la Fig.5 se observa el proceso de random jitter completo.

## III. Desarrollo

Se presentara de forma ordena todas las etapas del proyecto. Debido a la cantidad de imagenes utilizadas no es posible presentarlas a todas, por ello se relizaran analisís particulates para algunos casos de interes.

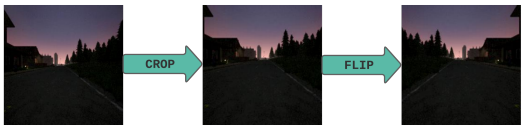


Figura 5. Visualización del random jitter.

Figura 6. Imagenes de muestra del dataset

### *III-A. Obtención del dataset*

Para obtener el dataset se colocaron una camara RGB y una camara de segmentación semántica [2], ubicadas en la misma posición. Se recolectaron los datos cada 5 segundos. Las condiciones climaticas utilizadas fueron aletorias, lo que permite obtener un dataset mucho mas variado Fig.6.

Para lograr un manejo autónomo del vehículo se utilizó el autopilot [1]. Con la finalidad de aumentar la variedad de imagenes se utilizaron todos los mapas posibles y se adicionaron otros vehiculos autónomos junto a personas. Aunque debido a limitaciones de hardware solo se pudieron ingresar 20 personas, lo cual provocó una deficiencia en la detección de personas de la red Pix2Pix.

### *III-B. Entrenamiento de la red*

Como fue mencinado en el marco teoríco el modelo implementado se denomina Pix2Pix [4]. Se utilizó como entrada la imagen de la camara RGB y la salida será una imagen segmentada, para ello se utilizó el dataset obtenido con el simulador de **CARLA** el cual posee 8000 imagenes.

*III-B1. Tratamiento de las imagenes:* Para evitar problemas en el entrenamiento de la red es muy usual llevar los valores de cada canal al intervalor $[-1; 1]$, esto evita problemas de continuidad en red. Luego se procede a aplicar un *random jitter*.

El dataset fue dividido en 2 partes una utilizada para el entrenamiento, aproximadamente el $80\,\%$ de las imagenes y otra parte para el testing de la red utilizando el $20\,\%$ restante, siguiendo el principio de Pareto.

*III-B2. Entrenamiento:* Para realizar el entremiento se utlizó el dataset de testin, aproximadamente 6400 imagenes, y se realizon 400 iteración a la red para lograr un mejor desenpeñó.

En la Fig.7 se muestra una comparación para 1 imagen del dataset de testing luego de la primera iteración y al finalizar el entrenamiento. Puede observarse que existieron grandes mejoras sobre todo en la zona de la palmera donde al finalizar el entrenamiento es posible observar los detalles de la misma. Aunque cabe destacar que debido a la poca cantidad de señales de trafico en el dataset se observa que si bien reconoce la estructura del semaforo no es capaz de utilizar el color adecuado.

Al analizar otro imagen del dataset la cual se observa en la Fig.?? COMPLETAR!!!

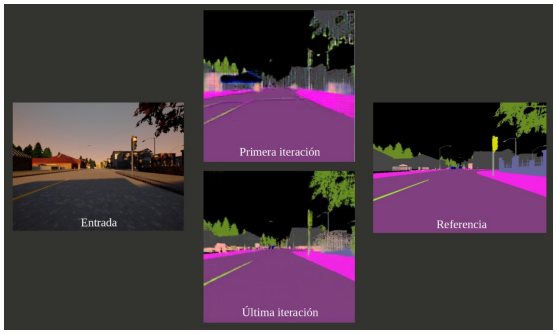


Figura 7. Imagen del dataset de testing.

## IV. Conclusiones

## Referencias

[1] "Carla api documentation," https://carla.readthedocs.io/en/latest/python_api/.
[2] "Carla sensors reference," https://carla.readthedocs.io/en/latest/ref_sensors/.
[3] "U-net architecture," https://en.wikipedia.org/wiki/U-Net.
[4] Phillip.I, J.Zhu, T.Zhou, and A.A.Efros, "Image-to-image with conditional adversarial networks," 2018.