Universidad Internacional de La Rioja (UNIR)

ESIT

Máster Universitario en Inteligencia Artificial

Planificación urbanística basada en GAN sensible a criterios ambientales

Trabajo Fin de Máster

Presentado por: Piñero Alquegui, Adán

Director/a: Dr. Infante Moro, Juan Carlos

Ciudad: Zaragoza
Fecha: 03/06/2021

Resumen

Las *Generative Adversarial Network* (GAN) son una arquitectura de redes neuronales de reciente desarrollo capaces de generar nuevas realidades a partir de un conjunto de datos de entrenamiento. Con el objeto de explorar el potencial de las redes generativas convolucionales en disciplinas que se expresan gráficamente, como el urbanismo, se desarrolla una herramienta de software denominada *eMapGAN*. Esta herramienta genera a partir de una imagen satélite del terreno, un mapa convencional, un mapa más sostenible e incluso una ortofoto de una realidad no existente, pero más sostenible que la original. Incluye también la posibilidad de editar mapas intermedios generando así nuevas realidades, todo ello mediante una interfaz gráfica de usuario. Se concibe como una herramienta experimental que explora las posibilidades de las redes GAN para inferir conocimientos complejos y constituir una herramienta de apoyo en las fases de concepción del diseño de las ciudades.

Palabras Clave: Redes generativas adversarias, sostenibilidad, planificación urbanística, cartografía, redes neuronales convolucionales, pix2pix.

Abstract

The *Generative Adversarial Network* (GAN) is a recently developed neural network architecture capable of generating new realities from a set of training data. In order to explore the potential of convolutional generative networks in graphically expressed disciplines, such as urbanism, a software tool called *eMapGAN* is developed. This tool generates from a satellite image of the terrain, a conventional map, a more sustainable map and even an orthophoto of a non-existent reality, but more sustainable than the original one. It also includes the possibility of editing intermediate maps, thus generating new realities, all through a graphical user interface. It is intended to be an experimental tool that explores the possibilities of GAN networks to infer complex knowledge and constitute a support tool in the conception phases of city design.

Keywords: Generative adversarial networks, sustainability, urban planning, cartography, convolutional neural networks, pix2pix

Índice de contenidos

Índice de tablas

Índice de figuras

1. Introducción

El crecimiento de la población y su concentración en ciudades supone un reto frente a los objetivos de desarrollo sostenible y de neutralidad climática fijados por los países desarrollados. En este sentido el urbanismo será una palanca del cambio hacia ciudades mejor planificadas, más sostenibles y eficientes ambientalmente. Como se observa en la Figura 1, diferentes conceptos urbanísticos desarrollan modelos de ciudades y por tanto distribuciones de población totalmente diferentes.

Figura 1. *Plano Diferentes urbanismos generan concentraciones de población diferentes.* (London School of Economics and Political Science, 2014)

Los algoritmos basados en Generative Adversarial Network (GAN) son descritos por primera vez en 2014 (Goodfellow, y otros, 2014) y están siendo utilizados para generar diseños industriales, reconstrucción de imágenes o generación de datos sintéticos entre otros. En el campo de la arquitectura han sido utilizadas para explorar nuevos diseños de viviendas (Nauata, Kai-Hung, Chin-Yi, Mori, & Furukawa, 2020) e incluso como herramienta experimental de apoyo en diseños urbanísticos (Fedorova, 2021).En todo caso, estas aproximaciones siempre se han realizado desde el punto de vista de la funcionalidad, la interconexión de espacios o incluso la valorización de la zona.

En el presente estudio se abordará el desarrollo de una herramienta de software, basado en GAN, capaz de interpretar la ortofoto de un área concreta y generar nuevos planos y ortofotos que tengan en cuenta criterios de diseño ambientales inferidos a partir de un conjunto de datos, residiendo en este aspecto la novedad.

No se pretende desarrollar un nuevo concepto de arquitectura GAN, sino que el desarrollo el código se apoyará fuertemente en la tipologías existentes y sus implementaciones como

pix2pix GAN, Cycle GAN o similares, combinados con dataset existentes y generadas ad hoc, con el objeto de generar las soluciones propuestas.

1.1 Motivación

La concentración de la población en las ciudades se ha acelerado en las últimas décadas, teniendo un impacto sobre el medio ambiente y sobre la calidad de vida de las personas. Los urbanistas planifican este crecimiento de forma desigual y atendiendo a diferentes intereses, en ocasiones económicos, estéticos o sostenibles. La planificación urbanística es un proceso en el que intervienen reglas, como la densidad de ocupación o la edificabilidad, pero también la creatividad y la intuición del urbanista. Es por tanto un proceso difícil de automatizar.

El presente trabajo propone una herramienta capaz de inferir conocimiento urbanístico a partir de un conjunto de planos, de tal forma que pueda ser capaz de complementar el proceso de planificación de forma automática. De esta forma servirá de herramienta de apoyo en fases previas de diseño a urbanistas, que podrán comprobar cómo puede influir la aplicación de diferentes modelos aplicados a diferentes zonas del territorio, incluso visualizando una ortofoto de cómo podría evolucionar el plano ordenado hacia una realidad futura.

1.2 Planteamiento del trabajo

Las inteligencias artificiales desarrolladas en las últimas décadas son más eficientes que las inteligencias humanas en problemas acotados y con reglas bien definidas. De esta forma existen algoritmos capaces de ganar al ajedrez o al go a los campeones humanos o pueden organizar un almacén de forma más eficaz que un experimentado responsable logístico.

En cambio, estas mismas inteligencias basadas en reglas, presentan serias dificultades para detectar una silla o un perro que no hayan visto con anterioridad, lo cual es capaz de hacer una mente humana ya en su edad más temprana.

El aumento exponencial en la capacidad de computación ha hecho surgir nuevas técnicas, basadas en redes neuronales, capaces de aprender de un conjunto de datos e inferir conocimientos de forma más sutil, más próxima a la forma de pensar humana. Ésta forma sutil de pensamiento posibilita abordar problemas más complejos, que incluyen incluso cierto grado de creatividad. Las arquitecturas GAN se han mostrado eficaces abordando este tipo de problemas.

El presente trabajo explora la forma en la que las arquitecturas GAN son capaces de inferir normas y estilos urbanísticos para aplicarlos de forma automatizada, de la misma forma que son capaces de simular el estilo pictórico de diferentes autores.

De esta forma un urbanista será capaz de organizar un nuevo dataset con las técnicas urbanísticas deseadas y generar un modelo capaz de convertir un mapa urbanístico actual en un posible mapa urbanístico futuro. Mediante la publicación de los diferentes modelos y de los diferentes dataset los profesionales del urbanismo podrán compartir esas diferentes formas de pensar el futuro, mediante las inferencias recogidas en las redes neuronales ya entrenadas, en definitiva, un conjunto de pesos estructurados.

Se propone un software que permita convertir una ortofoto actual en una ortofoto futura una vez aplicado un urbanismo sostenible inferido de un dataset creado ad hoc. Para ello se proponen los siguientes pasos:

Ortofoto actual > GAN > Mapa actual > GAN > Mapa futuro >GAN> Ortofoto futura.

El software desarrollado estará entrenado para un dataset creado por el autor, infiriendo la necesidad de añadir zonas verdes a las ciudades en zonas con alta densidad de población. En todo caso, será sencillo generar nuevos archivos .h5 a partir de dataset creados exprofeso, para generar nuevos conceptos urbanísticos.

El software y código desarrollado será código abierto, permitiendo así a la comunidad reutilizarlo y poder enriquecer y evolucionar el concepto planteado en este estudio.

1.3 Estructura de la memoria

La memoria se estructura en 7 capítulos, que pretender exponen de forma exhaustiva los objetivos, la metodología empleada, el trabajo desarrollado y las conclusiones obtenidas.

El capítulo 1 es una introducción donde se expone el contexto del problema a abordar, así como su utilidad. En líneas generales se reflexiona sobre el impacto del urbanismo en la calidad de vida y su dificultad para ser automatizado ya que incorpora componentes creativas y próximas a la intuición. Se proponen las arquitecturas GAN basadas en redes de neuronas, como un elemento eficaz para inferir conocimientos sutiles e incluso permitir cierto margen

creativo que pueda ser de utilidad para el planificador de las ciudades del futuro. Las posibilidades de convertir las fotografías aéreas a mapas y viceversa pueden ayudar al urbanista en su diseño de las ciudades del futuro.

El capítulo 2 consiste en un análisis exhaustivo del estado del arte de las redes GAN, especialmente aquellas capaces de operar entre dos campos de imágenes como las Conditional GAN. Se analiza cómo se han utilizado estas redes para automatizar procesos en campos similares, como el arte o la arquitectura, para analizar finalmente la aplicación directa de estas arquitecturas a la cartografía. Un aspecto importante en el desarrollo de este trabajo es el entrenamiento de las propias redes, por lo que se analizan los *dataset* disponibles, así como la forma de crear nuevos conjuntos de datos.

El capítulo 3 profundiza en el análisis de problema, identificando el objetivo principal y los objetivos específicos que se pretende alcanzar. De esta forma se identifica el conjunto de subtareas que permitirán elaborar los dataset necesarios, entrenar las redes y finalmente permitir mediante una secuencia de operaciones que una ortofoto presente sea transformada en una visión futura a la que se le ha aplicado un urbanismo más sostenible. En definitiva, un conjunto de subtareas que definirá un pipeline que permitirá que el software cumpla con el objetivo final descrito.

El capítulo 4 consiste en una identificación clara de los requisitos que debe aportar el software que se pretende desarrollar. Hay que asumir que se trata de una herramienta innovadora para los urbanistas por lo que resulta complicado que la comprendan sin mostrar prototipos previos. Para ello se desarrollan algunas imágenes a partir de los algoritmos entrenados y se mostrarán a expertos urbanistas. De esta forma se analizarán sus necesidades y propuestas y permitirá evolucionar tanto la parte de inteligencia de la herramienta como una interfaz para su uso. En todo caso, se asume que la herramienta será para uso conjunto entre un urbanista y un experto en inteligencia artificial, al tratarse de arquitecturas complejas y experimentales de las que se obtendrá su máximo potencial con un ajuste lo más fino y personalizado posible a cada caso.

A partir del análisis del estado del arte y de la identificación de requisitos durante el trabajo conjunto con los expertos urbanistas, se evolucionará el prototipo hacia una herramienta con mayor usabilidad. El capítulo 5 describe las funcionalidades de la herramienta desarrollada, de las redes entrenadas, de los data set utilizados y de la propia interfaz de uso.

El capítulo 6 pretende analizar el grado de consecución de los objetivos. Al tratarse de un concepto experimental se analizará su utilidad, con expertos en el dominio. De forma análoga

se expondrán los diferentes resultados obtenidos con diferentes tipologías, datasets y ajustes, de tal forma que queden documentados para futuras evoluciones del software.

Del conjunto de análisis se obtendrán conclusiones que se plasmarán en el capítulo 7. Estas conclusiones servirán como una referencia rápida sobre la utilidad real del concepto desarrollado y sobre el uso de arquitecturas GAN para generar urbanismos más sostenibles. Además, se propondrán líneas de trabajo que no hayan podido abordarse durante el desarrollo del presente estudio.

El código desarrollado será Open Source y será publicado juntamente con el presente documento, mediante un artículo, de tal forma que el concepto planteado pueda ser replicado y evolucionado por la comunidad interesada en continuar con la investigación.

El desarrollo de este trabajo no hubiera sido posible sin el aporte de numerosos investigadores, y durante todo el estudio se hará un esfuerzo en documentar y referenciar lo máximo posible todas aquellas ideas, conceptos e incluso código que se hayan incorporado o hayan podido inspirar fuertemente partes del presente documento. Por ello el Capítulo 8 se dedicará en su totalidad a recoger la bibliografía.

> *«Decía Bernardo de Chartres que somos como enanos aupados a hombros de gigantes, de manera que podemos ver más cosas y más lejanas que ellos, no por la agudeza de nuestra vista».* (Salisbury, 1159)

Por último, y en forma de anexos, se incluirá el acceso al código fuente desarrollado, listados, gráficos y logs de interés obtenidos de los diferentes entrenamientos y los dataset, o acceso a los mismos, tanto los generados ad hoc como los utilizados de otras fuentes.

El presente estudio se acompaña de un artículo de investigación, con el objeto de facilitar el acceso al trabajo realizado a los investigadores e interesados en el ámbito del trabajo.

2. Contexto y estado del arte

El aumento en la capacidad de computación ha posibilitado el desarrollo de la aplicación de las redes neuronales al tratamiento de imágenes. El principal soporte de información del urbanismo es gráfico, siendo su máxima expresión los mapas. Es por tanto que actualmente nos enfrentamos a una aplicación temprana y experimental de arquitecturas recientes, como las redes GAN, a problemas conocidos como la generación de mapas. En los siguientes apartados se contextualizará la evolución desde el urbanismo clásico hasta las recientes aplicaciones de redes neuronales generativas a los sistemas cartográficos.

2.1 El urbanismo como sistema integrador de información con alto impacto en el desarrollo del entorno

La planificación urbanística ha evolucionado desde los primeros constructores hasta nuestros días, consolidando una sólida base de conocimiento y un conjunto multidisciplinar de profesiones que participan en el proceso. Es habitual abordar la planificación de la mano de equipos integrados por arquitectos, ingenieros, geólogos, geógrafos, arqueólogos, naturalistas, demógrafos, abogados, artistas o economistas. El proceso de planificación urbanística consiste en coordinar y plasmar una serie de conocimientos de forma gráfica posibilitando el desarrollo de un territorio con un objetivo concreto, como la mejora en la calidad de vida o la sostenibilidad.

El concepto de ordenar los usos del suelo surge ya desde las primeras concentraciones humanas, alcanzando un grado de sofisticación importante ya en el desarrollo urbano de Roma como capital del Imperio. No obstante, el urbanismo, tal y como lo conocemos en la actualidad, surge durante las transformaciones industriales del siglo XIX. La rápida concentración de población en las ciudades y los nuevos avances tecnológicos requiere de eficaces y ordenados instrumentos para el desarrollo de las ciudades.

Uno de los máximos exponentes de la capacidad transformadora del urbanismo es el Proyecto de Reforma y Ensanche de Barcelona, cuyo facsímil es accesible gracias a estudios posteriores (Terán Troyano, 2009). Tras la demolición de las murallas de la ciudad y la necesidad de crecimiento de la ciudad por el aumento de la población, el plan que desarrollaría el Ensanche fue aprobado el 30 de mayo de 1860 y define organización de parte de la ciudad de Barcelona de los últimos 160 años.

En definitiva, el proceso consistió en el concurso de varios proyectos de ensanche, cada uno de ellos con diferentes teorías sobre el urbanismo. Las novedosas ideas de Cerdá sobre

"urbanizar lo rural y ruralizar lo urbano" (Magrinya, 2010) se ve reflejada en su concepción de calles y de manzanas y en definitiva se plasma, gráficamente, en un conjunto de planos, como observamos en la Figura 2.

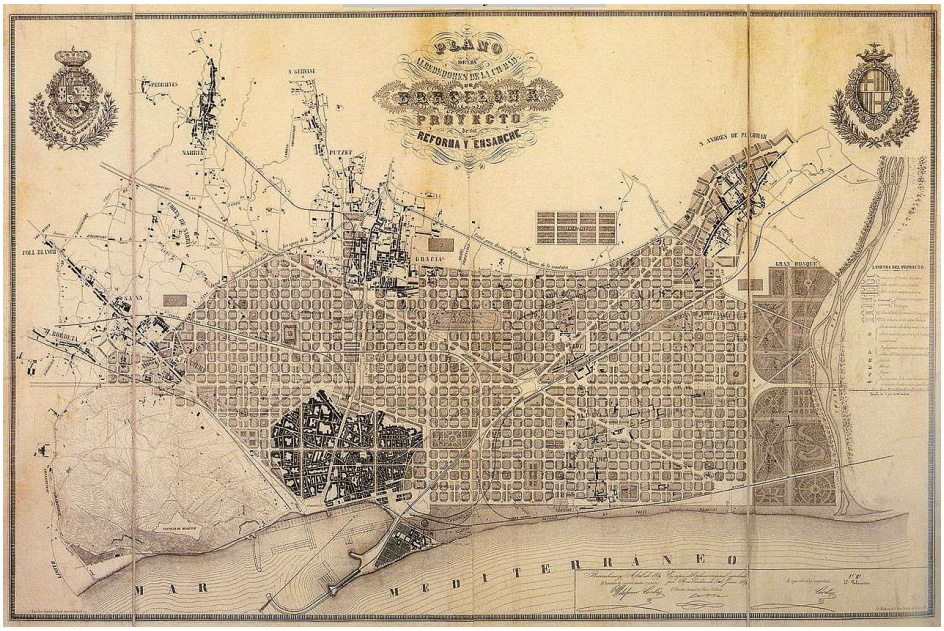

Figura 2. *Plano original del Ensanche de Barcelona.* (Terán Troyano, 2009)

Años después estos planos dan lugar a un desarrollo real, como observamos en la Figura 3. Este proceso, desde la inferencia de ideas, su plasmación gráfica y un resultado real es el que pretendemos capturar, mediante redes neuronales y de forma experimental, en el presente trabajo. Adicionalmente observamos que un conjunto de teorías, cuyos conocimientos son inferidos y plasmados de forma gráfica en forma de planos, trascienden más allá de la ciudad para la que fueron concebidos. En la figura 3, observamos la influencia del ensanche de Barcelona en las ciudades de Santiago de Chile y Montevideo (Malave & Aceves, 2019).

Figura 3. *Fotografías aéreas del parte de los ensanches de Barcelona (izquierda), Santiago de Chile (centro) y Montevideo (derecha).* (Malave & Aceves, 2019)

Por tanto, entendemos el urbanismo como un sistema para integrar la información de forma eficaz y con un alto impacto sobre el entorno donde se aplica.

2.2 La aplicación temprana de la inteligencia artificial a la arquitectura y el urbanismo

"El urbanismo, consiste en incentivar la disposición para aplicar la teoría a la práctica, en fomentar la integración de la información a la transformación coherente del entorno." (Francel & Uribe Kaffure, 2020). Es por ello por lo que las técnicas de inteligencia artificial que sean capaces de integrar información para su aplicación práctica se intuyen de utilidad en el proceso de planificación urbanística.

Un primer y significativo avance ha consistido en la digitalización de los planos, la utilización de fotografías aéreas y la incorporación del CAD al proceso de diseño. La flexibilidad en el desarrollo de planos y en la integración de la información por capas ha permitido, especialmente a partir de los años 90, diseños más complejos y con mayor detalle de los resultados.

Nuevas fuentes de datos abiertas, información procedente de redes sociales, datos geo posicionados y el uso de tecnologías globalizadas permiten planificar con más información, incluso entendiendo la planificación urbana como una evolución de las ciudades 2.0 hacia nuevas versiones 3.0. (Anttiroiko, 2012). Se observa esta integración de información en la Figura 4.

Figure 2. Technological trends relating to City 2.0 and City 3.0

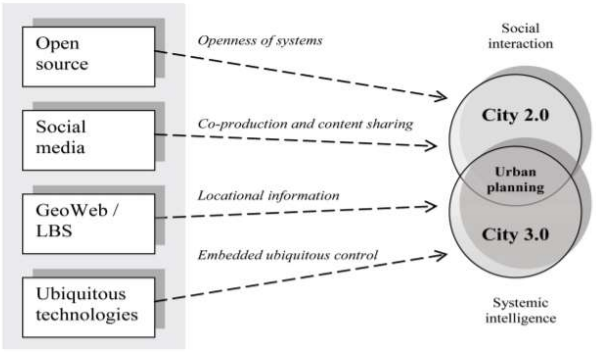

Figura 4. *De las ciudades 2.0 a las ciudades 3.0.* (Anttiroiko, 2012)

Los sistemas de ML son utilizados para generar mapas, incluso clasificando la tipología del suelo de acuerdo con el análisis de la textura de fotografías aéreas (Ruiz, 2018). Las técnicas más avanzadas en aprendizaje profundo nos permiten un enfoque más allá de la clasificación, que permita generar, o al menos inspirar, nuevas realidades.

En relación con el presente estudio existen aproximaciones inspiradoras a los ámbitos próximos a la arquitectura y el urbanismo y las redes de neuronas. En este sentido se supera la barrera definida por reglas fijas y bien definidas para abordar los procesos de diseño desde un enfoque más creativo e intuitivo. No obstante, cabe destacar que la aplicación temprana de tecnologías avanzadas de IA a la concepción gráfica de la arquitectura y el urbanismo es más experimental que útil. Las técnicas profesionales desarrolladas durante años están consolidadas y se demuestran eficaces, por lo que las nuevas técnicas exploradas, aún no son capaces de aportar soluciones mejores que las diseñadas de forma tradicional. En todo caso, no se pretende superar a las técnicas convencionales, sino explorar la aplicación de las nuevas tecnologías a caminos ya conocidos. Esto puede ser una buena definición de innovación.

Desde el punto de vista conceptual, son de especial interés las redes Archigan (Chaillou, 2019) basadas en topologías CycleGAN (Zhu, Taesung, Isola, & Efros, 2017) y Pix2Pix (Isola, Zhu, Zhou, & Efros, 2017). En sucesivas aproximaciones, el autor genera un pipeline capaz de utilizar redes neuronales para analizar una parcela, orientar y proponer un contorno para una edificación y generar un plan de necesidades, una distribución e incluso un amueblamiento (Chaillou, 2019). Es posible explorar incluso la aplicación de diferentes estilos arquitectónicos. Los resultados presentan importantes inexactitudes, pero no pretenden ser exhaustivos sino exploratorios. Es por ello por lo que permiten un análisis rápido e inspirador basado en redes entrenadas bajo conjuntos de datos diferenciados. Ésta es una buena inspiración para la aplicación de redes neuronales profundas en el planeamiento urbanístico. La figura 5 muestra algunos de estos resultados.

Figura 5. *Manchas de colores y distribución arquitectónica generada por red GAN.* (Chaillou, 2019)

La introducción de técnicas de inteligencia artificial en otros dominios bien consolidados, como la arquitectura y el urbanismo, requiere de expertos capaces de operar de forma transversal entre los dos campos. Muy recientes investigaciones abordan herramientas que buscan proporcionar propuestas de planificación que faciliten la labor de los profesionales. El algoritmo denominado LUCGAN (Pengyang, Fu, Dongjie, Bo, & Chang-Tien, 2020) define futuros usos del suelo entorno a un desarrollo existente, gracias a una eficaz extracción de características de mapas disponibles, rutas de transporte y precio de la vivienda, y al uso de redes GAN.

2.3 Las redes neuronales capaces de crear nuevas realidades o generativas.

Las redes neuronales artificiales son un modelo computacional consistente en un conjunto de unidades capaces de transmitirse señales entre sí. Existen diferentes funciones de activación de la neurona, diferentes tipologías de capa y arquitecturas diversas de interconexión, si bien tienen en común que se trata de sistemas conexionistas. Aunque existen referencias anteriores a 1958, diversos autores (Olazaran, 1993) identifican al perceptrón (Rosenblatt, 1958) y al algoritmo back-propagation (Linnainmaa, 1970), como los avances más significativos hacia las arquitecturas matemáticas capaces de inferir conocimiento a partir de un aprendizaje previo, es decir una red neuronal artificial.

El incremento exponencial de la capacidad de computación (Moore, 1975) y de almacenamiento de datos han permitido un resurgimiento de las aplicaciones y arquitecturas relacionadas con las redes neuronales en los últimos años.

Las redes neuronales discriminativas son aquellas capaces de inferir información a partir de un entrenamiento y clasificar nuevas entradas a la red dentro de un conjunto de categorías conocidas. En cierto modo son capaces de predecir la pertenencia de un registro a una clase u otra a partir de un entrenamiento con datos estructurados. Un ejemplo de esta tipología es la red neuronal InceptionResNetV2 (Google, 2015). Una red neuronal convolucional pre-entrenada por Google con más de un millón de imágenes y capaz de clasificar nuevas imágenes entre 1000 categorías de objetos predefinidas.

En 2014 se define una tipología de red neuronal conocida como Variational autoencoder (VAE) (Kingma & Welling, 2014), precursora de las redes generativas más actuales. Una red generativa es una red neuronal capaz de inferir información a partir de un entrenamiento. Una vez completado este proceso de entrenamiento será capaz de generar nuevos registros, similares a los del propio dataset de entrenamiento. En cierto modo es capaz de generar

nuevas realidades a partir de un conjunto conocido de realidades existentes. Este tipo de redes resultan de especial interés para el presente trabajo, ya que se pretende determinar el impacto de nuevos urbanismos incluso en ortofotos futuras.

Ahora bien, es necesario afrontar el reto de entrenar de forma eficaz estas redes generativas, para que sean capaces de crear nuevas realidades que no puedan diferenciarse de las originales. En 2014 se desarrolla el concepto de redes generativas antagónicas o GAN (Goodfellow, y otros, 2014). La figura 6 muestra, en amarillo, nuevas imágenes creadas por un algoritmo GAN similares a las del conjunto de entrenamiento.

Figura 6. *Nuevas imágenes generadas por una red neuronal GAN.* (Goodfellow, y otros, 2014)

Una red GAN entrena de forma adversaria a una red neuronal generativa y a una red neuronal discriminativa. Este enfrentamiento se produce en un contexto de juego de suma cero, es decir, aquello que un jugador gana lo pierde su oponente. De esta forma se enfrentan una red generativa que busca crear imágenes realistas y una red discriminativa que pretende determinar si la imagen generada por su oponente es falsa o perteneciente al dataset inicial. De esta forma, discriminador y generador, tras etapas de entrenamiento ajustan sus pesos consiguiéndose finalmente un generador capaz de crear imágenes nuevas no diferenciables del dataset original.

La arquitectura GAN parece por tanto una técnica generativa disruptiva que podría aplicarse a la generación de nuevos planos, desde el punto de vista de ser capaz de generar e inspirar nuevas realidades.

2.4 La aplicación de las GAN en la cartografía y las ortofotos

La planificación urbanística integra información y concentra su máximo potencial de expresión mediante planos. Gracias a técnicas capaces de convertir ortofotos en mapas y viceversa es posible además interpretar conjuntamente estos dos campos de información. Es por tanto que exploraremos a continuación el estado de la tecnología basada en arquitecturas GAN a la elaboración de mapas y a su relación con las ortofotos y con otros dominios de la imagen.

Recientes aproximaciones al diseño urbano centrado en la transferencia de estilos mediante redes neuronales definen los mapas resultantes como mapas imaginarios (del Campo & Manninger, 2020). Los autores profundizan en la idea de la toma de decisiones como una colaboración entre el humano y la máquina. En este sentido entrenan una arquitectura Style GAN (Karras, Laine, & Aila, 2018) con un dataset de mapas Nolli (University of Oregon, 2005), un estilo cartográfico del siglo XIX y lo aplican a cartografías actuales. De esta forma consiguen aplicar el conocimiento inferido de los mapas existentes para generar nuevos mapas en ciudades, por ejemplo, en la luna.

Los resultados, imaginarios o alucinatorios como los definen sus autores, pueden ser inspiradores para el urbanista. En otras investigaciones del mismo autor (del Campo, Carlson, & Manninger, Towards Hallucinating Machines - Designing with Computacional Vision., 2020), se divide un dataset en función de su estilo; Moderno o Barroco, y se infiere ese conocimiento sobre nuevos planos utilizando la arquitectura StyleGAN.

Más allá de la cartografía, la generación de imágenes a nivel de suelo a partir de imágenes aéreas también se ha abordado aplicando arquitecturas GAN. Un estudio del año 2018 (Xueqing, Yi, & Shawn, 2018) utiliza redes cGAN entrenadas a partir de un dataset de fotografías aéreas y a nivel de suelo, para generar imágenes fotorrealistas a nivel de suelo desde nuevas fotografías aéreas. Si bien el objetivo de esta investigación es conseguir las fotografías lo más próximas a la realidad, siempre existe cierta disimilitud, lo que en el caso del planeamiento no es un inconveniente sino una ventaja creativa.

Al inicio de este apartado se hace referencia a la cartografía como forma habitual de expresar la planificación urbana, aunque cabe reseñar que también es posible hacerlo mediante variables, como la demografía o la ocupación. Estas variables pueden interconectarse con grafos simulando las conexiones entre diferentes usos del suelo. Ésta es una línea de trabajo

alternativa, menos gráfica pero más concisa en variables cuantificables, abordada en 2021 por investigadores (Wang, Fu, Wang, Huang, & Lu, 2020) de la Universidad de Florida también mediante el entrenamiento de redes adversarias. En el presente trabajo nos centraremos en el carácter gráfico de los planos, procesados mediante redes GAN convolucionales, no optando por las variables discretas representadas mediante grafos. No obstante, la aplicación de arquitecturas GraphGAN (Wang, y otros, 2018)puede resultar una línea de investigación complementaria.

2.5 Arquitecturas GAN convolucionales con aplicación a la generación de planos.

En 2014 se propone una nueva arquitectura para modelos generativos basados en un proceso antagónico o adversario. Se denominaría Generative Adversarial Networks o GAN (Goodfellow, y otros, 2014). Consiste en entrenar dos modelos de forma simultánea. Un modelo generativo (G) crea una distribución de datos y un modelo discriminativo (D) estima la probabilidad de los datos creados por G pertenezcan al dataset de entrenamiento. Durante el entrenamiento G intenta maximizar los errores de D, y D por su parte intenta diferenciar datos propios del dataset y datos generados por G.

Inicialmente el entrenamiento de estos algoritmos resultó complicado y se realizaban soluciones a medida del dataset o del problema planteado. Presentaban problemas de convergencia en el entrenamiento y funciones de pérdida definidas a medida de cada problema.

No obstante, hay problemas con características comunes que pueden ser afrontados desde una arquitectura predefinida y probada con éxito. En 2018 el laboratorio para la investigación en AI de la UC Berkeley “investiga las GAN como una solución de propósito general a problemas de traslación imagen a imagen” (Isola, Zhu, Zhou, & Efros, 2017). Las conclusiones de la investigación concluyen en una arquitectura denominada pix2pix capaz no sólo de generar imágenes de salida a partir de imágenes de entrada, sino que ajusta de forma adecuada la función de pérdida durante el entrenamiento. De esta forma es aplicable a problemas genéricos como la traslación entre fotos a mapas o desde imágenes con contornos a fotos. Los autores consiguen resultados aceptables con dataset de 400 imágenes y 2 horas de entrenamiento en una GPU Pascal Titan X.

Cabe destacar que las métricas de evaluación no se basan en pretender que la imagen resultante coincida con la realidad, sino en que la imagen resultante sea percibida como real. Es por ello por lo que se plantean métricas relacionadas con la percepción estadística

consultada a observadores humanos. Algunos autores se refieren a esta circunstancia como “clasificación semi-supervisada donde la meta principal no es entrenar un modelo que consiga mucha similitud al dato de entrenamiento […] buscando imágenes que los humanos encuentren realistas” (Salimans, y otros, 2016). Esta arquitectura pix2pix evoluciona hacia imágenes en alta resolución, denominándose pix2pixHD (Wang, y otros, Hight-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs, 2018), con el apoyo de NVIDIA y con importantes requisitos de computación.

Hay ocasiones donde las imágenes entre los dos dominios no están emparejadas entre sí. En 2020 una investigación (Zhu, Taesung, Isola, & Efros, 2017) de la UC Berkeley propone un algoritmo capaz de trasladar imágenes de un dominio a otro, con una arquitectura GAN entrenada con dataset no emparejados imagen a imagen. De esta forma es capaz de convertir una fotografía con caballos en cebras, o del invierno al verano, como se observa en la figura 7.

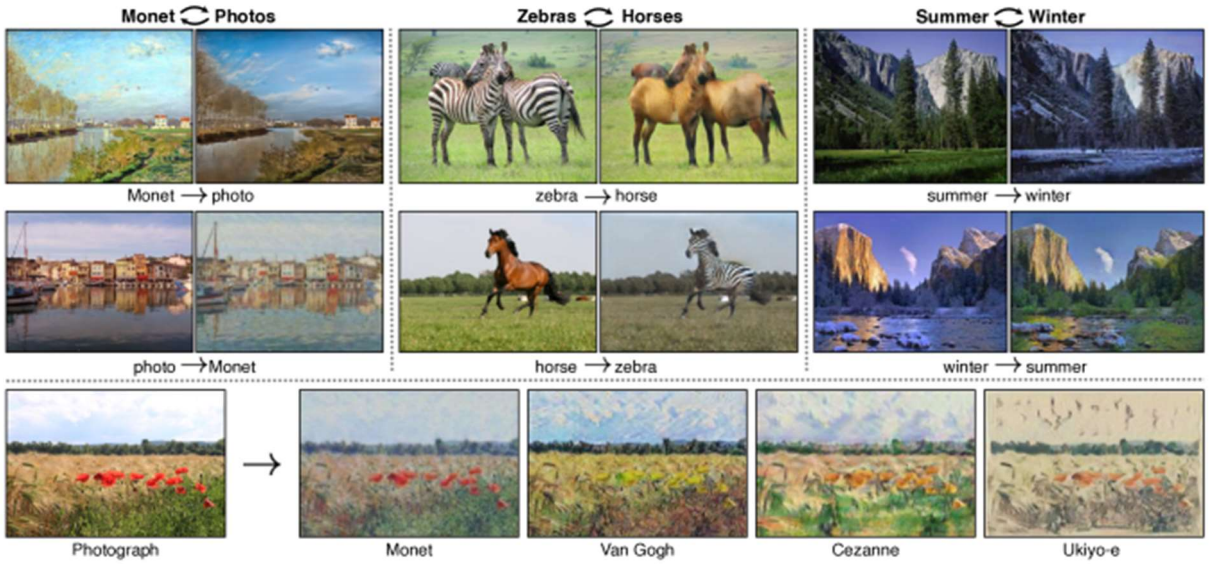

Figura 7. *Resultados obtenidos entrenando GANs entre dos conjuntos de imágenes desemparejadas entre sí.* (Zhu, Taesung, Isola, & Efros, 2017)

Sobre la arquitectura pix2pix se han realizado numerosas propuestas con el objeto de optimizar la conversión de ortofotos a mapas. De esta forma surgen arquitecturas específicas con este fin, en todo caso a costa de perder la flexibilidad que presenta el algoritmo pix2pix original. Estas arquitecturas, entre otras son MapGAN que utiliza grafos (Li, Chen, Zhao, & Shao, 2020), StarGAN (Choi, y otros, 2018) con capacidad multidominio , BicycleGAN (Zhu, y otros, 2017) que limita la variabilidad en los posibles outputs para un único input y MSGAN (Tran, Tran, Nguyen, & Cheung, 2019) con la incorporación de aprendizaje auto supervisado

para datos no etiquetados. La figura 8 muestra un comparativo de resultados obtenidos con diferentes redes GAN.

Figura 8. *Comparativo entre varias arquitecturas photo-to-map* (Li, Chen, Zhao, & Shao, 2020).

Cabe destacar que no se localizan en el mercado soluciones comerciales de software que utilicen GAN para apoyar al urbanista en su diseño mediante la generación automática de planos, con un componente creativo y que además puedan incorporar criterios ambientales.

2.6 Herramientas genéricas de conversión imagen a imagen

El presente estudio utilizará de forma sistemática la arquitectura basada en la tipología Pix2Pix (Isola, Zhu, Zhou, & Efros, 2017). Esta arquitectura cosiste en una red GAN, cuyo generador es capaz de traducir imágenes entre dos campos distintos. De la misma forma que es posible traducir de idioma español al idioma inglés, o de una tipología de imágenes a un etiquetado semántico de partes de estas, es posible traducir entre dos conjuntos de imágenes. Por ejemplo, de imágenes por satélite a mapas, de fotografías a cuadros, de fotografías diurnas a nocturnas o de imágenes de primavera a imágenes de invierno.

El concepto de traducción imagen a imagen se ha abordado tradicionalmente con herramientas específicas para cada uso. En cambio, el algoritmo pix2pix supone una red razonablemente capaz de trabajar entre conjuntos de imágenes de forma no especificada para un uso concreto. Es decir, es un algoritmo más genérico, por lo que puede adaptarse a las necesidades del desarrollo de software objeto del presente trabajo.

Hasta la aparición de las GAN el enfoque generativo se ha centrado en la resolución de herramientas a medida para problemas muy acotados. Existen técnicas de composición o *acolchado* (Efros & Freeman, 2001), que, combinando pequeñas piezas de un conjunto de imágenes en otra, generando una nueva que adquiere eficazmente texturas del dataset original.

La metodología denominada Images Analogies (Hertzmann, Jacobs, Oliver, Curless, & Salesin, 2001) enfoca la generación de nuevas imágenes como si de la aplicación de filtros se tratara. De esta forma a partir de un conjunto de imágenes de entrenamiento A: A' es capaz de inferir un filtro que aplica a una imagen B generado la nueva B' como si de la aplicación de un filtro se tratase. Se trata de una metodología generativa, aunque no enfrenta redes adversarias.

En todo caso observamos que estas aproximaciones se asocian a la resolución de problemas muy concretos. En cierto modo esto tiene su lógica, ya que se basan en reglas más bien fijas de procesamiento, por lo que intentar realizar soluciones versátiles complica el algoritmo en gran medida. De ahí la especialización en soluciones a medida.

Es la propuesta de la arquitectura GAN (Goodfellow, y otros, 2014) la que permite un significativo avance en el entrenamiento de redes neuronales generativas. De esta forma soy capaz de entrenar la red generativa con la ayuda de un discriminador que en cierto modo va enseñado al generador el camino correcto. De esta forma es posible que el generador infiera las características necesarias del dataset de forma supervisada. Es decir, mediante dataset donde las imágenes del dominio A y el B se emparejan entre sí, es posible entrenar una red Pix2Pix que es capaz de inferir el conocimiento necesario para generar nuevas imágenes B a partir de A. Esta versatilidad tiene la limitación importante de la necesidad de que las imágenes del dataset estén emparejadas, pero por otro lado ha sido la base del desarrollo de desarrollos posteriores.

En un estudio pormenorizado de las redes GAN más significativas desarrolladas en base a la traducción imagen a imagen supervisada (Yongxin, 2020) resalta CGAN, BicycleGAN, SPA-GAN.

CGAN (Isola, Zhu, Zhou, & Efros, 2017) o también conocida como pix2pix, se refiere a una arquitectura GAN con un generador encoder-decoder de arquitectura U-Net, capaz de converger razonablemente para una variedad de problemas.

BicycleGAN (Zhu, y otros, 2017) es capaz de generar relaciones entre el espacio latente generado y las imágenes de salida de forma bidireccional. Presenta mejoras que evitan que diferentes inputs puedan dar lugar a un mismo output lo que podría suponer un colapso modal durante el entrenamiento.

SPA-GAN (Emami, Aliabadi, Dong, & Chinnam, 2020) integra la capacidad de generar mapas de atención en el discriminador. De esta forma, el discriminador no sólo se pronuncia respecto a la clasificación real o fake de la imagen generada, sino que puede focalizar su trabajo en determinadas regiones de la imagen.

Como hemos comentado con anterioridad este conjunto de redes GAN supervisadas requieren de un conjunto de imágenes pareadas. Esto supone una limitación ya que no siempre es fácil disponer de dataset con esta tipología de emparejamientos. Por ejemplo, es posible disponer de un conjunto amplio de imágenes diurnas y nocturnas, pero puede ser difícil conseguir que estén emparejadas entre si correspondiendo al mismo paisaje. Es por ello por lo que se desarrollan métodos no supervisados, donde no es necesario el emparejamiento de imágenes.

El mismo estudio de análisis de las redes GAN en base a traducción imagen a imagen (Yongxin, 2020), en este caso de forma supervisada destaca:

CycleGAN (Zhu, Taesung, Isola, & Efros, 2017). Esta tipología introduce un tipo de función de coste relacionada con la relación entre los dominios de la imagen, de tal forma que es capaz de inferir las características que permiten pasar de un dominio a otro sin la necesidad de disponer de las imágenes emparejadas. En general funciona mejor para traducir texturas o imágenes, en vez de formas.

DualGAN (Yi, Zhang, Tan, & Gong, 2017). Esta tipología de red entrena un generador capaz de traducir imágenes del dominio A al dominio B, y otro que realiza la función inversa. De esta forma es capaz de traducir A al dominio B y a partir de B reconstruir A’ y compararlo con la A original. De optimizar este proceso de reconstrucción es capaz de entrenar generadores razonablemente eficaces.

Como hemos indicado, el utilizar una arquitectura no supervisada, capaz de aprender a generar nuevas imágenes a partir de un dataset de dos dominios no pareados entre sí, supone la ventaja de una mayor facilidad para conseguir el conjunto de datos. No obstante, en

contrapartida, este tipo de redes requiere un número muy superior de registros. Por este motivo se desecha para este trabajo este tipo de arquitecturas GAN, si bien pueden ser una línea de trabajo futura.

En definitiva, dada la versatilidad de la arquitectura cGAN o pix2pix y pudiéndose entrenar de forma adecuada a partir de 400 imágenes pareadas, parece adecuado el uso de esta tipología para el desarrollo de las diferentes redes neuronales que componen la herramienta objeto del presente trabajo.

2.7 Fuentes de datos de cartografía y fotografías aéreas

Si bien el software que se desarrolla en el presente estudio puede ser utilizado con imágenes aéreas y cartografía procedente de diferentes fuentes, siempre que sea adecuadamente preprocesadas, se procederá a un análisis de las fuentes de datos disponibles y a la selección de las más adecuadas como base para el presente estudio.

Los satélites proporcionan un flujo constante y actualizado de fotografías aéreas de diversas tipologías. Generalmente, al ser sistemas desarrollados por organismos públicos una parte de estos datos es obtenible a partir de los portales de datos abiertos.

A modo de ejemplo, el Instituto Geográfico Nacional, dependiente del Ministerio de Transportes, Movilidad y Agenda Urbana de España, facilita los siguientes datos de forma abierta:

- Plano topográfico nacional en escala 1/25.000 y 1/50.000 tanto en formato imagen como en formato vectorial.
- Bases cartográficas provinciales y regionales.
- Modelos digitales del terreno, en diferentes resoluciones.
- Mapas históricos a diferentes escalas.
- Mosaicos de ortofotos en alta resolución como parte del programa PNOA (Ministerio de Transportes, Movilidad y Agenda Urbana. Gobierno de España., 2015).
- Otros datos georefenciables, como las unidades administrativas, las mallas cartográficas y geodésicas, los límites municipales, o los censos de población.

Adicionalmente las comunidades autónomas ofrecen datos en abierto, haciendo un especial énfasis en los usos del suelo y en la protección del medio ambiente. En general si bien existen sistemas de información geográfica (GIS) capaces de aglutinar, exportar e importar datos de diferentes fuentes, no siempre esta integración es directa y es necesario preprocesar las imágenes o las capas cartográficas.

Más allá de los datos facilitados por los organismos públicos existen proveedores de cartografía y ortofotos satélite, generalmente bajo licencia de pago. Estos datos están claramente estructurados y actualizados, presentan una mayor resolución que los proporcionados por los organismos públicos y acostumbran a distribuirse mediante una API eficaz. Un ejemplo de este tipo de proveedores es Mapbox.com o la propia API de pago de Google Maps. La ventaja de este tipo de datos es la calidad de estos, la cobertura internacional y la API de acceso bien desarrollada. En contrapartida requieren el desarrollo de un acceso a su API, así como una licencia de pago por lo que se desecha esta opción para el trabajo desarrollado, si bien se consideran una línea de trabajo futura.

Existen proveedores que cumplen con las especificaciones de los proveedores anteriores y con licencias de uso libre razonablemente amplias como es el caso de OpenStreetMap (Open Street Map, 2021) utilizado en algunas aplicaciones de geolocalización como Ushahidi (Ushahidi Inc., 2021). Pueden ser una opción razonable bajo una hipótesis de hacer la herramienta de software distribuible para un uso masivo y gratuito.

Por último, existen conjuntos de datos, que combinan ortofotos y planos, específicamente desarrollados para el entrenamiento o testeo con desarrollos de inteligencia artificial. Habitualmente son desarrollados por universidades y constituyen una buena opción para el software experimental desarrollado. La ventaja de estos dataset es que permiten comparar soluciones o reutilizar de forma sencilla parte del código ya desarrollado sirviendo como soporte para nuevas versiones, combinando incluso soluciones de varios investigadores. Similar a los conocidos dataset MNIST utilizado para testear soluciones de redes neuronales sobre un conjunto de cifras escritas a mano existen dataset para el trabajo con ortofotos y cartografía.

En este sentido se selecciona el dataset recomendado por el propio autor del algoritmo Pix2Pix (Isola, Zhu, Zhou, & Efros, 2017) denominado Maps Dataset. Se compone de ortofotos de Nueva York y su correspondiente mapa grafiado de acuerdo con la simbología de Google Maps. Consiste en dos conjuntos de 1097 y 1099 pares de imágenes y su mapa correspondiente con una resolución de 600x600. Las imágenes son en color RGB, mostrándose una de ellas en la Figura 9.

Figura 9. *Ejemplo del dataset utilizado y utilizado en la arquitectura Pix2Pix.* (Isola, Zhu, Zhou, & Efros, 2017)

Se ha optado por el tratamiento de imágenes en color de tamaño 256x256 ya que son suficientes para apreciar el funcionamiento del software, aunque al límite de la capacidad de computación de la que se dispone, especialmente para entrenar los modelos. Podrían utilizarse resoluciones mayores si bien, disponiendo de una importante capacidad de entrenamiento consistente en un conjunto de unidades GPU en paralelo. Ésta puede ser una línea de trabajo futura, adecuando el software a mayores resoluciones, siembre bajo sistemas computacionales de muy altas prestaciones.

Por último, cabe destacar que si bien se ha seleccionado un dataset consistente en ortofotos y su correspondiente cartografía existen otras representaciones más complejas de la superficie terrestre que podrían resultar en líneas de trabajo de interés. Cabe destacar la tecnología LIDAR, capaz de captar ortofotos más realistas gracias a introducir sensores capaces de medir la distancia real entre los objetos (Gharibi & Habib, 2018). Parece evidente la evolución de la fotografía hacia las tecnologías capaces de escanear en las tres dimensiones espaciales, recogiendo así de forma exacta la forma de los objetos, además de su apariencia.

2.8 Soluciones comerciales para la aplicación de GAN a la cartografía

Las arquitecturas basadas en redes neuronales adversarias surgen en 2014, siendo las primeras referencias de algoritmos eficaces con el tratamiento de imágenes próximas a los años 2017 y 2018. Es por tanto una tecnología experimental que se está adoptando de forma

temprana en diversos ámbitos profesionales como el urbanismo y la arquitectura. No obstante, estas primeras aproximaciones son aún teóricas y no tienen una aplicación práctica directa que haya sido explotada comercialmente de forma generalizada.

No se han localizado por tanto herramientas comerciales de uso genérico que utilicen las arquitecturas GAN para generar nuevas imágenes satélite. Existen usos especialmente relacionados con los proveedores de cartografía, que utilizan arquitecturas de este tipo, aunque lo hacen de forma interna y no publican abiertamente sus algoritmos.

Algunos autores estiman las ventajas de este tipo de algoritmos llegando a cuantificar el desarrollo de un mapa a escala 1:2000 de 10 km2 a mano puede suponer unos 20 días, mientras que un algoritmo bien entrenado puede hacerlo en 39 segundos (Song, Li, Chen, & Wu, 2021).

2.9 Conclusiones con aplicación directa

Del análisis del estado del arte podemos concluir que puede entenderse el urbanismo como un proceso integrador de muchas fuentes de información y que se expresa de forma gráfica, mediante planos. Además, causa un impacto duradero y de gran escala en su entorno por lo que tiene sentido trabajar la interacción entre los planos y las ortofotos, en los dos sentidos. El sentido ortofoto hacia plano, está bien documentado y supone la observación de la realidad existente y su representación gráfica. En cambio, el sentido plano hacia ortofoto es contraintuitivo, ya que debemos recurrir a la imaginación para abstraer cómo un plano puede impactar en una fotografía, que entendemos como expresión de la realidad. En este sentido cobran importancia los avances en redes neuronales generativas, que presentar algunas capacidades creativas de interés en esta transformación.

Las diferentes soluciones basadas en GAN son recientes, tratándose de algoritmos desarrollados desde 2014 y que empiezan a ser ampliamente utilizados por los investigadores especializados a partir del 2017. El incremento en la potencia de los procesadores gráficos ha permitido la investigación relacionada con redes neuronales convolucionales que pueden funcionar como traductor entre dos dominios de imágenes.

En este sentido, la traducción o transferencia de estilos entre imágenes es de especial interés en aquellas ciencias cuyos resultados se plasman gráficamente, como el urbanismo o la cartografía. Es por ello que se han desarrollado recientemente arquitecturas específicas para convertir las fotografías aéreas en mapas como MapGAN (Li, Chen, Zhao, & Shao, 2020), StarGAN (Choi, y otros, 2018), BicycleGAN (Zhu, y otros, 2017) y MSGAN (Tran, Tran, Nguyen, & Cheung, 2019).

Por otro lado, se ha detectado interés en el uso de las arquitecturas GAN como apoyo al proceso creativo en la arquitectura e incluso en el urbanismo, de la misma forma que se había introducido en el arte o la música. En este sentido cabe destacar cómo se asume de forma generalizada que el objetivo no es una predicción exacta de la realidad, sino la generación de una realidad posible. Este aspecto será clave a la hora de evaluar los resultados de los algoritmos entrenados en apartados sucesivos.

Por último, se ha identificado la arquitectura pix2pix como idónea para generar planos a partir de imágenes y de forma inversa imágenes a partir de contornos o bocetos. Además, adapta su función de pérdida de forma automatizada y permite resultados razonables con dataset de 400 imágenes. Dados los recursos limitados de computación de los que se dispone será la opción preseleccionada como core del software. Se localizarán asimismo implementaciones de código pix2pix en Keras para la implementación de esta arquitectura en el software desarrollado.

Analizado el estado del arte podemos por tanto concluir que el desarrollo de una herramienta de software capaz de servir de apoyo e inspiración en la toma de decisiones a un técnico urbanista a la hora de imaginar ciudades más sostenibles presenta interés. Este interés es reciente y se centra en investigadores capaces de trabajar tanto en el campo de la arquitectura y del urbanismo como en el desarrollo de inteligencia artificial. Las redes GAN se consolidan como candidatas a dar este soporte, ya que no es objetivo una representación exacta de una realidad conocida sino crear una aproximación a una nueva realidad.

3. Objetivos y metodología de trabajo

A continuación, se describirán los objetivos que se pretenden alcanzar con el presente desarrollo. En primer lugar, se concretará un objetivo general, que definirá claramente la funcionalidad de la herramienta de software desarrollada.

En segundo lugar, se definirán una serie de objetivos específicos, un conjunto de subtareas que convergerán en la consecución del objetivo general. De esta forma, adicionalmente, es posible medir el grado de consecución de cada una de las partes abordadas en el desarrollo.

Por último, se planteará una metodología de trabajo y una secuencia lógica de tareas y acciones que permitan completar el desarrollo previsto en el objetivo general.

3.1. Objetivo general

El objetivo general es desarrollar una herramienta de software capaz procesar una ortofoto aérea de un área de una ciudad y generar una nueva realidad, más sostenible, mostrándola tanto en mapa como en ortofoto. La sostenibilidad se define como un incremento sustancial de zonas verdes, que un algoritmo GAN pre entrenado determina. La herramienta se circunscribe al dataset disponible de parte de la ciudad de Nueva York y será testeado por arquitectos urbanistas, entendiendo que se trata de una herramienta experimental como inspiración para los procesos previos de diseño. Adicionalmente es posible retocar etapas intermedias del proceso, como los planos o las representaciones denominadas eDraw, de tal forma que sea posible ajustar los resultados obtenidos.

A continuación, en la figura 10, se muestran dos de las imágenes utilizadas en los entrenamientos previos. La primera de ellas es una imagen aérea de la ciudad de Nueva York. La segunda ha sido generada por un pipeline basado en arquitecturas GAN pix2pix que han sido entrenada para generar nuevas imágenes más sostenibles. Se aprecia que se trata de un urbanismo ingenuo pero que se plantea como herramienta complementaria al urbanista en sus fases previas de tomas de decisiones. En cierto modo podría incluso entenderse como una ventana al futuro.

Figura 10. *Imagen original (izquierda) e imagen generada con zonas verdes (derecha)* (Elaboración propia prototipos y testeos iniciales)

3.2. Objetivos específicos

El objetivo principal puede descomponerse en cinco objetivos específicos bien diferenciados. Consisten en tareas relacionadas entre sí, pero que por la tipología de los trabajos a realizar y las tecnologías utilizadas son diferenciables. De esta forma es posible agrupar las dificultades encontradas, las soluciones planteadas y las líneas de trabajo futuras en cada uno de estos subconjuntos.

O. 1.- Identificar la arquitectura GAN adecuada para realizar la traslación entre dos dominios de imágenes, con dataset limitados a 1000 imágenes en color de baja resolución y apto para los recursos de computación consistentes en menos de 50 horas de GPU en la nube. Incorporará una función de pérdida capaz de converger entre dominios de foto a foto y de foto a contornos.

O. 2.- Definir un pipeline entre varias redes GAN capaz de generar una ortofoto con urbanismo sostenible a partir de una ortofoto de la ciudad de Nueva York. Además, cada una de esas redes GAN entre dos dominios se entrenará adecuadamente con las siguientes tipologías:

- Foto aérea a mapa

- Mapa a imagen de machas según usos (agua, zona verde, vivienda o carretera)

- Imagen de manchas a imagen de manchas más sostenible

- Imagen de manchas a mapa

- Mapa a foto aérea.

La capa imagen de manchas puede ser prescindible. Se testeará su eficacia y se optará por una solución de conversión mapa a mapa si procede.

O. 3.- Identificar un conjunto de datos utilizable para el entrenamiento consistente en imágenes aéreas emparejadas con mapas. Se deberá abordar la forma de pretratar estas imágenes para su uso en la red GAN.

O. 4.- Diseñar e implementar una mínima interfaz de usuario que permita un uso razonable del pipeline. De esta forma será capaz de visualizarse la ortofoto original y la final mejorada, además de estados intermedios, como el mapa. Se trata de una interfaz básica y limitada al área geográfica de entrenamiento del dataset, en ningún caso incorporará la posibilidad de entrenar nuevos datasets para otras áreas geográficas, siendo esto una labor destinada al ingeniero en inteligencia artificial si procediera realizarse de nuevo.

O. 5.- Testear con expertos en el dominio; un arquitecto urbanista, un ingeniero en planificación y un artista conversando sobre la funcionalidad experimental que aporta la herramienta e identificar posibles líneas de trabajo futuras que pudieran converger en utilidades comerciales.

No es un objetivo el desarrollo completo del código en Python de una red GAN, por lo que se utilizan implementaciones ya existentes modificadas para adecuar su uso al problema tratado. Por tanto, el código generado está inspirado fuertemente en implementaciones Pix2Pix en Python basado en Keras (Brownlee, 2019).

3.3. Metodología del trabajo

Para consecución de los objetivos planteados en los subapartados anteriores se procederá a plantear una metodología de trabajo. Como se ha indicado con anterioridad se han definido los subobjetivos de tal forma que atienden a partes sustancialmente diferentes del proyecto, con características y tecnologías diferentes, por tanto, se abordarán con metodologías específicas. En general el software se desarrollará bajo una filosofía Agile, específicamente con dinámicas Scrum.

O. 1.- Identificar la arquitectura GAN adecuada para realizar la traslación entre dos dominios de imágenes, con dataset limitados a 1000 imágenes en color de baja resolución y apto para los recursos de computación consistentes en menos de 50 horas de GPU en la nube.

Para completar este objetivo se analizará el estado del arte de las redes GAN, en sus diferentes arquitecturas, especialmente aquellas que se utilizan para la traslación entre dos

dominios de imágenes. Una vez identificado un algoritmo viable se realizará un entrenamiento de este con fotografías aéreas y en sentido inverso. El entrenamiento se realizará con los recursos disponibles durante el proyecto, previsiblemente Colab o AWS.

Se medirá el tiempo de entrenamiento y se verificará si los resultados obtenidos son aceptables. En este sentido se recuerda que las métricas recomendadas no se basan en obtener representaciones fidedignas de la realidad sino aceptables a la percepción humana.

Se hará un primer test con usuarios expertos, mostrándoles imágenes en baja resolución reales y generadas por las GAN y determinarán si son aceptables. Si adicionalmente el tiempo de entrenamiento y los recursos consumidos son razonables se seleccionará la arquitectura testeada.

No es necesario probar varias arquitecturas y compararlas entre sí, sino que se optará por aquella recomendada del análisis profundo del estado del arte y en caso de no ser adecuada se continuará testeando más opciones.

O. 2.- Definir un pipeline entre varias redes GAN capaz de generar una ortofoto con urbanismo sostenible a partir de una ortofoto de la ciudad de Nueva York. Además, cada una de esas redes GAN entre dos dominios se entrenará adecuadamente.

Se entrenará una red GAN capaz de realizar una traslación entre el conjunto de fotos aéreas a mapas. Se guardarán los datos de la función de pérdida y de error durante el entrenamiento y se computará el tiempo de entrenamiento. Se visualizarán resultados cada 10 epoch decidiendo así cuando se alcanza un entrenamiento suficiente.

Se realizará el mismo proceso invirtiendo los dominios y obteniendo una transposición de mapa a foto aérea.

Se creará un algoritmo que cree a partir de mapas, manchas de colores, identificando solamente las zonas de agua, verdes, carreteras y urbanizadas, con 4 colores. Se creará un conjunto a partir de estas imágenes con más zonas verdes y se inferirá ese conocimiento a una red GAN entrenándola entre los conjuntos de “manchas” y de “machas mejorado”. La figura 11 muestra un esquema del pipeline de trabajo, que se explicará en profundidad en sucesivos apartados.

Se entrenará una red GAN capaz de realizar una traslación entre el conjunto de mapas de manchas a mapa o directamente a fotos aéreas, según se obtengan mejores resultados. Se guardarán los datos de la función de pérdida durante el entrenamiento y se computará el

tiempo de entrenamiento. Se visualizarán resultados cada 10 epoch decidiendo así cuando se alcanza un entrenamiento suficiente.

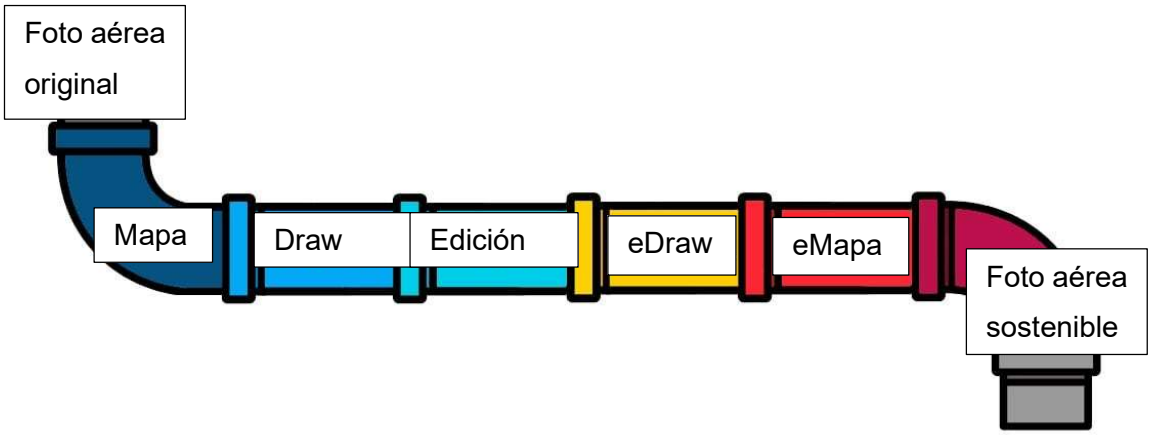

Figura 11. *Flujo de trabajo o pipeline de las imágenes* (Elaboración propia)

O. 3.- Identificar un conjunto de datos utilizable para el entrenamiento consistente en imágenes aéreas emparejadas con mapas. Se deberá abordar la forma de pretratar estas imágenes para su uso en la red GAN.

La metodología del objetivo O.3 y del objetivo O.2. deben está interrelacionadas ya que se procede a entrenar redes GAN con dataset identificadas o creadas a tal fin.

Se realizarán una búsqueda exhaustiva de dataset de fotografías aéreas y sus mapas, bien desde las API de los sistemas de geoposicionamiento como de las propias utilizadas en los artículos de investigación analizados en el estado del arte.

Una vez identificada una muestra suficiente, se testeará con los algoritmos definidos en los anteriores apartados. Se valorará la posibilidad de ampliar el dataset, bien con nuevos datos o bien con técnicas de rotación y variación de escala.

O. 4.- Diseñar e implementar una mínima interfaz de usuario que permita un uso razonable del pipeline. De esta forma será capaz de visualizarse la ortofoto original y la final mejorada, además de estados intermedios, como el mapa. Se trata de una interfaz básica y limitada al área geográfica de entrenamiento del dataset, en ningún caso incorporará la posibilidad de entrenar nuevos datasets para otras áreas geográficas, siendo esto una labor destinada al ingeniero en inteligencia artificial si procediera realizarse de nuevo.

El objetivo del software es experimental y en ningún caso se pretende una interfaz compleja ni comercial. Una vez que se haya conseguido ejecutar en Python un pipeline que sea capaz de cumplir los objetivos O1, O2 y O3 será considerado un producto mínimo viable. Este desarrollo tendrá el inconveniente de que sólo puede ser utilizado por un técnico con conocimientos de programación y capaz de entender el código utilizado.

Se identificará un framework de Python capaz de crear una ventana que muestre una imagen, o permita precargar una imagen del conjunto de datos y muestre la imagen final mejorada e incluso las capas intermedias.

Se testeará con 3 usuarios, en todo caso teniendo en cuenta que se trata de un software experimental con el objeto de analizar la potencialidad de las redes GAN y de inspirar en fases previas de diseño urbanístico.

O. 5.- Testear con expertos en el dominio; un arquitecto urbanista, un ingeniero en planificación y un artista conversando sobre la funcionalidad experimental que aporta la herramienta e identificar posibles líneas de trabajo futuras que pudieran converger en utilidades comerciales.

Una vez obtenida la primera versión del software operativo se testeará con tres profesionales de otros campos que pudieran utilizar la herramienta en búsqueda de utilidades que futuros desarrollos pudieran implementar.

Se prevé realizar esta acción con un arquitecto urbanista, siendo el técnico que finalmente puede hacer un uso de la inspiración aportada por la herramienta en fases previas de diseño.

Se prevé realizar la prueba con un ingeniero de planificación, con el objeto de que explore las capas intermedias y las herramientas basadas en visualización de realidades a partir de mapas.

Por último, se analizarán los resultados con un artista que pueda plantear conceptos como la alucinación de ciudades o la creación de ciudades distorsionadas, analizando capas intermedias y reflexionando sobre estos conceptos, de la misma forma que se ha podido explorar en artículos de investigación descritos en el estado del arte.

En general el software se desarrollará bajo una filosofía Agile, específicamente con dinámicas Scrum (Schwaber, 1997). Si bien el presente estudio no se desarrolla por un equipo de personas, parece adecuado adoptar varias de las técnicas Scrum, maximizando así las probabilidades de éxito del desarrollo de una aceptable herramienta de software.

Para el seguimiento de esta metodología en primer lugar se identifican los elementos del Product Backlog a completar, tal y como se ha realizado al inicio del presente apartado. Estos objetivos se desglosarán a su vez en tareas más concretas y abarcables de tal forma que puedan ser abordadas de forma individual. Se plantean sprint periódicos con los que desarrollar estos objetivos. Se han identificado los siguientes sprint durante el desarrollo del software objeto de este estudio:

- Prueba de diferentes algoritmos GAN, verificando su funcionamiento y las posibilidades de entrenamiento dentro de las capacidades de computación disponibles.
- Exploración de las capacidades de computación disponibles en la nube y selección de la más adecuada.
- Selección de un dataset o proveedor de ortofotos adecuado.
- Entrenamiento del dataset seleccionado, en el entorno seleccionado, con la arquitectura gan seleccionada.
- Ajuste de parámetros y de la programación de la arquitectura
- Definición del pipeline desde la Ortofoto hasta la nueva ortofoto mejorada, pasando por los mapas y los edraw.
- Creación de un data set para el entrenamiento desde map hasta emap.
- Desarrollo y recopilación de métricas de error y de entrenamiento.
- Mejoras posibilitando la captación de capas intermedias del generador de tal forma que pueda observarse la inferencia de características por las diferentes capas convolucionales.
- Pruebas de mejora mediante dropout y data augmentation.
- Selección de un entorno gráfico GUI
- Creación de un entorno gráfico GUI
- Integración de los modelos entrenados y funciones desarrolladas en la nube en el entorno gráfico GUI creado.
- Testeo y pruebas del software.

- Mejoras especialmente relacionadas con la posibilidad de retoques de mapas en pasos intermedios.
- Empaquetado del software para su uso en otros ordenadores.
- Testeo con usuarios profesionales.
- Documentación del software.
- Documentación de los algoritmos y soluciones utilizadas. Artículo científico.

Tras cada iteración se hace una reflexión de los logros obtenidos y se reajustan nuevas tareas. La filosofía de trabajo se ha mostrada acertada, siendo un desarrollo de softare experimental del que no se disponía de procesos y requerimientos rígidamente definidos, sino que se han podido completar paralelamente al desarrollo del propio proyecto. De esta forma, como se preveía de acuerdo con la figura 12, se han maximizado las probabilidades de éxito del proyecto.

	Waterfall	**Spiral**	**Iterative**	**SCRUM**
Defined processes	Required	Required	Required	Planning & Closure only
Final product	Determined during planning	Determined during planning	Set during project	Set during project
Project cost	Determined during planning	Partially variable	Set during project	Set during project
Completion date	Determined during planning	Partially variable	Set during project	Set during project
Responsiveness to environment	Planning only	Planning primarily	At end of each iteration	**Throughout**
Team flexibility, creativity	Limited - cookbook approach	Limited - cookbook approach	Limited - cookbook approach	**Unlimited during iterations**
Knowledge transfer	Training prior to project	Training prior to project	Training prior to project	**Teamwork during project**
Probability of success	Low	Medium low	Medium	**High**

Figura 12. *Comparativa de metodologías de desarrollo de proyectos (Schwaber, 1997)*

4. Identificación de requisitos

A lo largo del presente capítulo se expondrá el trabajo realizado para adecuar el software a las necesidades detectadas en el apartado anterior. El proceso de diseño de la herramienta de software se ha realizado de forma iterativa, de tal forma que se ha podido mantener un estrecho contacto con expertos en urbanismo, arte y arquitectura de tal forma que el software ha podido evolucionar de acuerdo con sus requerimientos.

El prototipado es uno de los seis vectores que aceleran el desarrollo de procesos innovadores (Oliván, 2020). Mediante el prototipado es posible generar una conversación eficaz sobre el proyecto. De esta forma, siguiendo una metodología Design Thinking se ha optado por el desarrollo iterativo de prototipos que han podido ser comentados con posibles usuarios finales.

4.1. El proceso Design Thinking. Iterando prototipos.

La forma eficaz de incorporar los requisitos aportados por posibles usuarios a la herramienta de software se ha realizado mediante un proceso iterativo, con el desarrollo de diferentes prototipos y con la puesta en común de los mismos.

El proceso denominado Design Thinking Process (Hasso Plattner Institute of Design at Stanford, 2019), como se observa en la Figura 13, propone 5 pasos desde la primera toma de contacto con el usuario hasta la prueba de la herramienta.

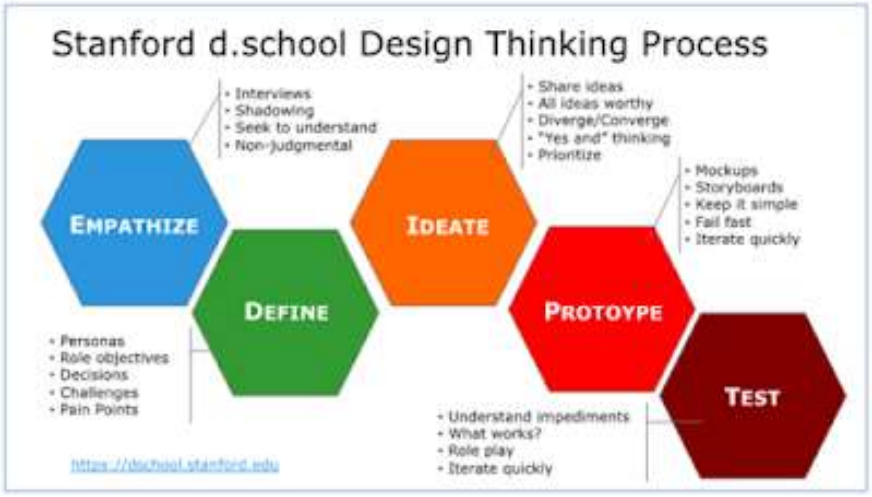

Figura 13. *Design Thinking Process en 5 pasos (Hasso Plattner Institute of Design at Stanford, 2019)*

Siguiendo la filosofía de aproximación iterativa se definen cuatro fases, aplicando los 5 pasos del método en cada una de ellas.

4.2. Fase 0. Prospección previa y muestra de interés

Atraído por la capacidad de las redes GAN para generar nuevas realidades se exponen estas ventajas a tres perfiles seleccionados, un doctor en geografía, un doctor urbanista y una artista profesional. El objeto de esta fase es captar ideas y constatar el interés de los profesionales en la materia.

Fase 0. Empatizar.

En primer lugar, se expone la complejidad del urbanismo, que abarca conocimientos complejos y sutiles que difícilmente pueden ser sustituidos por una máquina. El proceso de urbanismo, aunque se plasme de forma gráfica en gran medida incorpora sensibilidades sociales, artísticas y de compresión de la realidad más allá de las evidentemente técnicas. De esta forma se empatiza con los profesionales para que entiendan la herramienta como una exploración de las redes GAN en su campo y no como una utilidad directa para generar urbanismos.

Fase 0. Definir.

Se trabaja sobre similares proyectos, definidos en el estado del arte del presente documento, y que fundamentalmente abarcan la creación de mapas alucinatorios, transición de estilos entre mapas y las plantas y fachadas generadas mediante redes generativas.

Fase 0. Idear.

Se concreta que parece razonable trabajar sobre la conversión entre ortofotos y mapas, mapas y mapas mejorados y entre mapas y ortofotos.

Fase 0. Prototipar.

Se trata de una serie de conversaciones previas de las que no se genera prototipo alguno, sino que simplemente se concretan las ideas.

Fase 0. Test.

Se trata de una serie de conversaciones previas de las que no se genera prototipo alguno, sino que simplemente se concretan las ideas.

Se concluye la fase 0 de prospección previa constatando el interés de los profesionales en que se puedan desarrollar herramientas experimentales que exploren la traducción entre imágenes de diferentes conjuntos, como foto a mapa o mapa a foto. También muestran interés por conocer la forma de generar esta información de las redes neuronales.

4.3. Fase 1. Prototipados manuales y en papel

El objetivo de esta fase es generar en papel, o de forma manual, algunos prototipos que puedan ser testeados y permitan avanzar en la identificación de requisitos.

Fase 1. Empatizar.

En primer lugar, se conversa acerca de las necesidades de sus profesiones. Cabe destacar que los arquitectos presentan su falta de libertad en el diseño urbanístico ya que en ocasiones las limitaciones normativas son excesivas o no permiten expresar de forma adecuada las soluciones del arquitecto. Respecto a los geógrafos indican las ventajas de los portales SIG que integran la cartografía, suponiendo un avance y una normalización en la materia. Los expertos en arte se interesan por la forma en la que una red neuronal es capaz de inferir información y sus aplicaciones en diferentes campos sensoriales, como la música o la escritura, además de la imagen.

Fase 1. Definir.

Se concreta que un pipeline de trabajo adecuado para facilitar al cartógrafo y al urbanista visualizar el impacto de sus decisiones sobre el territorio, en forma de fotografía puede ser:

Ortofoto -> mapa -> Mapa modificado -> Ortofoto

Se discute sobre si la conversión de mapa a mapa modificado que debe de ser automática se debe programar mediante normas rígidas o dejar libertad a una red neuronal. Parece haber consenso en que dado el carácter experimental y exploratorio del software se opte por la red neuronal, si bien sus resultados pudieran ser menos predecibles y exactos. En contrapartida podrían resultar más inspiradores, lo que si resulta útil a los profesionales.

Fase 1. Idear.

Se concreta que la forma adecuada de hacerlo es mediante una herramienta de software que deberá tener una interfaz gráfica para poder trabajar con ella. Son conscientes de que la funcionalidad será limitada, pero les parece suficiente para conocer las capacidades. En caso de estudios más avanzados o grandes proyectos entienden que el desarrollo se haría

juntamente con un ingeniero experto en inteligencia artificial. Ponen como ejemplo utilizar el urbanismo de un zoco árabe para aplicarlo a la ciudad de Las Vegas y mostrar realidades paralelas y reflexionar así si una ciudad es la mejor opción posible de sí misma.

Fase 1. Prototipar.

Se editan fotografías a mano para mostrar resultados posibles. Algunas incluso se generan a partir de redes gan especialmente para la conversión map2sat. En la figura 14 se muestra un ejemplo de algunas de las imágenes prototipadas a mano. En la imagen izquierda se observa la fotografía original y en la derecha se han substituido manzanas enteras por zonas verdes.

Figura 14. *Prototipo editado sobre mapa a mano e imagen generada con map2sat* (Elaboración propia)

Durante esta fase se realizan además varios bocetos en papel de cómo debería ser una GUI. Parece interesante que las imágenes puedan descargarse directamente desde mapas de internet, si bien es una funcionalidad que se valorará su implementación más adelante.

Fase 1. Test.

El test impacta sobre los profesionales, si bien inciden en que la herramienta no capta la sutileza del urbanismo, sino que es bastante primitiva. Es decir, elimina casas para poner parques. En todo caso entienden que se trata de un software exploratorio y experimental y que su carácter es más inspirador que productivo.

4.4. Fase 2. Prototipo desde cuadernos Jupyter

Durante esta fase se testean los primeros resultados en serie de las redes neuronales ya entrenadas, despertando el interés de los profesionales por poder utilizarlo por sí mismos, sin necesidad de tener conocimientos de programación.

Fase 2. Empatizar.

Se prepara un sencillo dossier con algunos de los resultados obtenidos de tal forma que los expertos puedan verificarlos con anterioridad. Esto favorece la comprensión del trabajo

desarrollado, y permite al experto reflexionar sobre las posibilidades de aplicación experimentales. Se realizan videoconferencias por separado con los expertos para tratar los avances en el proyecto.

Fase 2. Definir.

Si bien se trabaja sobre los trabajos ya generados con cuadernos Jupyter se insiste mucho por parte de los profesionales en disponer de una interfaz gráfica. Incluso, especialmente los relacionados con el mundo le arte, les gustaría poder entrenar sus propios modelos para utilizarlos de forma divergente.

Fase 2. Idear.

Se concreta que las imágenes de 256x256 son suficientes para hacerse una idea de las posibilidades del sistema, entendiendo que atienden a un compromiso con la capacidad de computación en el entrenamiento de los modelos. Se concreta la interfaz gráfica que será sencilla e intutitiva.

Fase 2. Prototipar.

Se aportan a los profesionales las imágenes que se consiguen mediante los cuadernos Python. Les parece adecuado el espacio draw de manchas de color, si bien un editor gráfico podría ser de utilidad. Pueden observarse algunas de estas imágenes en esta fase en la figura 15.

```
import numpy as np
train_draw_images=[]
for image in train_map_images:
  train_draw_images.append(deteccion_de_calles(image))
train_draw_images=np.array(train_draw_images)
plot_images([train_sat_images[40],train_map_images[40],train_draw_images[40]])
```

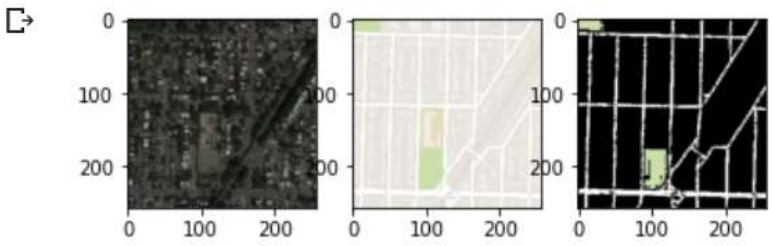

Figura 15. *Imágenes generadas desde los cuadernos Python* (Elaboración propia)

Fase 2. Test

Las imágenes son adecuadas si bien no es práctico al necesitarse a un programador para generar cada una de las imágenes. Se insiste en el desarrollo de la GUI.

4.5. Fase 3. Producto mínimo viable con GUI

Durante esta iteración se detecta la necesidad de una interfaz gráfica, además del interés por las capas intermedias del modelo.

Fase 3. Empatizar.

Se muestra un primer producto viable con su propia GUI, si bien aún se ejecuta sobre Python y no se ha generado un paquete de instalación para Windows. Se muestra en la Figura 16.

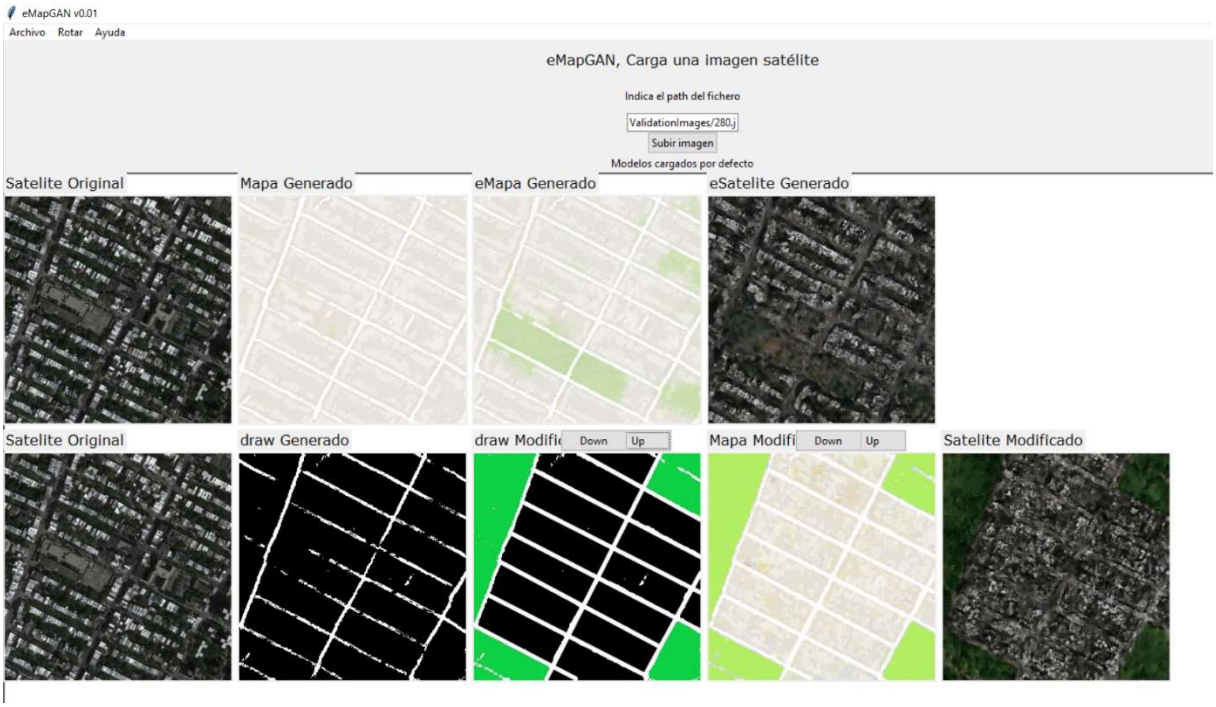

Figura 16 Interfaz GUI del software desarrollado (Elaboración propia)

Fase 3. Definir.

Insisten en ventaja de editar a su gusto las imágenes y los mapas. Se solicita poder ver las capas intermedias de la red neuronal. Se trabaja sobre ellas y se muestran en la figura 17.

Fase 3. Idear.

Se concreta un método sencillo para editar las imágenes, mediante un botón download y upload y editando la imagen con el editor de Windows. En todo caso parece que un editor personalizado, por ejemplo, solo con los colores del mapa facilitaría la edición.

Fase 3. Prototipar.

Se adapta la GUI y se presenta una edición funcional viable del programa, si bien aún no está empaquetada para Windows.

Fase 3. Test

Se testea el software tal y como se presenta en la figura 17, concluyendo que puede considerarse como una primera herramienta mínima viable. Se detecta la necesidad de probar la herramienta en diferentes ordenadores, con diferentes tamaños de pantalla y resoluciones, así como de generar documentación de ayuda.

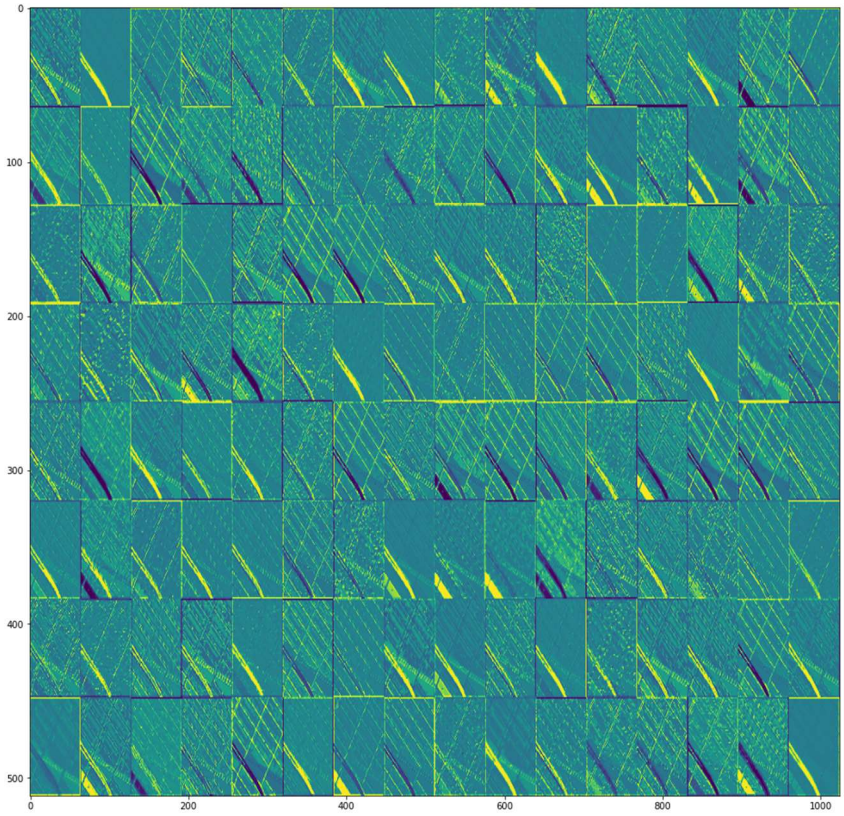

Figura 17. *Capas intermedias del encoder* (Elaboración propia)

4.6. Conclusiones y resumen de requisitos identificados

El proceso iterativo se ha mostrado eficaz, con especial ayuda de los prototipos. De esta forma las conversaciones con los expertos han podido realizarse sobre herramientas concretas y gracias a ello converger y detectar con más facilidad requisitos directamente relacionados con el software desarrollado. Asimismo, el proceso iterativo mejora las posibilidades de éxito, pudiendo incorporarse o reconducirse características de forma temprana.

En la Tabla 1 se expone un resumen de los requisitos planteados y su incorporación al proyecto.

Tabla 1. Resumen de requisitos a partir de las fases de diseño e iteración del prototipo.

Fase	Requisito	Profesional	Adoptado	Línea Futura
0	Foto->mapa->emapa->Foto	Urbanismo	Si	-
0	Foto->mapa->Foto	Geografía	Si	-
0	Versátil entre campos imagen	Arte	Parcial	Si
1	Disponer de interfaz gráfica	Urbanismo	Si	-
1	Disponer de interfaz gráfica	Geografía	Si	-
1	Seleccionar/Cambiar red h.5	Arte	Si	-
2	Crear ciudades en espacios vacíos	Urbanismo	No	Si
2	Compatible con SIG	Geografía	No	No
2	Guardar fotografías	Arte	Si	-
3	Crear/Editar mapas ad hoc	Urbanismo	Si	-
3	Imagen directa de Maps	Geografía	No	Si
3	Ver capas intermedias r. neuronal	Arte	Si	-

Elaboración Propia

Aquellas características de interés para el software, pero no abarcables durante esta fase de desarrollo se incorporan como líneas de trabajo futuras. Se explorarán brevemente en el capítulo 7.

5. Descripción de la herramienta software desarrollada

En el presente apartado se describirá con detalle el proceso del desarrollo, de acuerdo con los diferentes hitos definidos en el apartado 3. Se describirá en primer lugar el funcionamiento de la herramienta obtenida facilitando así una visión de conjunto. A partir de aquí se describirá cada proceso desarrollado con detalle con el objeto de exponer no sólo la función de cada elemento sino su funcionamiento interno. Cabe destacar que al tratarse de una herramienta experimental el principal interés reside en el funcionamiento interno del programa, ya que puede resultar inspirador o determinante para otros desarrollos posteriores.

La herramienta se ha denominado eMapGAN, queriendo ser un nombre que facilite la comprensión intuitiva de la funcionalidad que desarrolla. En definitiva, el objeto de la herramienta es generar mapas, o sus ortofotos, más ecológicas de ahí la denominación eMap. Por otro lado, la arquitectura utilizada es la tipología de redes neuronales adversarias, o su acrónimo en inglés GAN *(Generative Adversarial Network)*.

5.1. Descripción funcional de la herramienta desarrollada

La herramienta experimental de software desarrollado permite cargar una imagen satélite de en formato jpg ubicada en cualquier directorio del sistema operativo. Una vez indicado el path a la imagen y pulsado el botón cargar imagen, ésta se visualizará en la parte izquierda de la pantalla, bajo la denominación Satélite Original.

A partir de aquí la herramienta generará dos flujos de imágenes con características diferentes, como puede observarse en la captura del programa representada en la figura 13. Si bien se expondrá su funcionamiento y características en apartados sucesivos a continuación se describe de funcionalidad desde el punto de vista del usuario.

El primer flujo corresponde a la fila superior de imágenes y aparece denominado de la siguiente forma:

Satélite Original > Mapa Generado > eMapa Generado > Satélite Generado

Consiste en un *mapa generado* a partir de la *imagen satélite*, mediante una red neuronal entrenada para este efecto, denominada *Sat2Map.h5*. A partir de la imagen *mapa generado* se generará una nueva imagen *eMapa generado*, mediante la red neuronal *Map2eMap.h5*, y

consistirá en un mapa con algunas mejoras ambientales. Básicamente el algoritmo reconoce zonas de costa y determina que deberían ser verdes y además no permite grandes espacios urbanos sin ninguna zona verde. A partir de este *eMapa* más sostenible se generará, mediante la red neuronal ya entrenada *eMap2Sat* una *Imagen Satélite* nueva mostrando cómo sería la ortofoto de la zona de terreno original si hubiéramos aplicado en su urbanismo los criterios ambientales inferidos por la red neuronal *map2emap.h5*. En la figura 18 se muestra una captura del software desarrollado que integra los flujos de trabajo descritos.

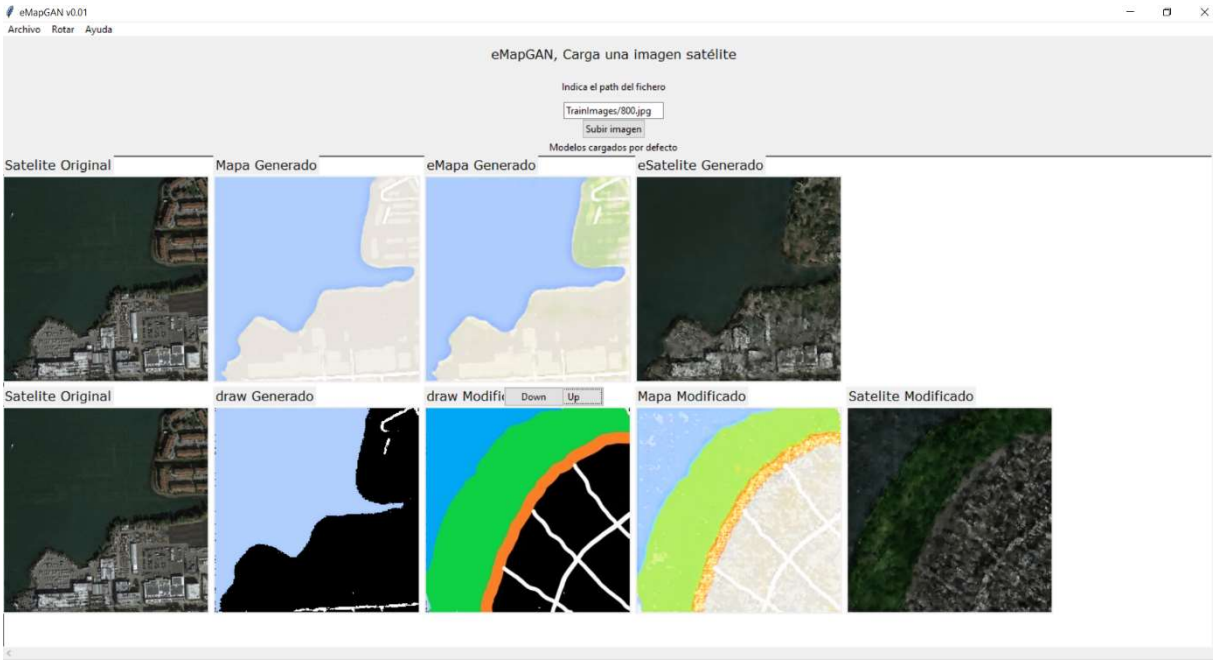

Figura 18. *Captura del software desarrollado denominado eMapGAN* (Elaboración propia)

El segundo flujo de imágenes se representa en la fila inferior de la pantalla del programa y pretende que el usuario pueda modificar el mapa a su voluntad. Para ello se ha definido un espacio denominado draw, que es una representación sencilla del mapa en manchas de color, lo que facilita la generación de nuevos mapas. También es posible modificar el espacio mapa, consiguiéndose añadir algún detalle más fino. De esta forma el flujo de este segundo pipeline de imágenes es:

Satélite Original > draw Generado > draw Modificado > Mapa Modificado > Satélite Modificado

Si bien se explicará con detalle en apartados sucesivos se expone a continuación el funcionamiento de este flujo de imágenes. Se parte de la imagen original cargada del satélite para aplicar la red neuronal preentrenada Sat2Map.h5, generando de esta forma un mapa. A partir de este mapa se aplicará una función de procesamiento píxel a pixel que reconoce los colores del mapa y los aglutina en sencillas manchas de color, más fácilmente editables por el usuario. De esta forma se genera el denominado draw. Es posible descargar, editar y cargar

de nuevo este archivo draw, que aparece en la carpeta por defecto del programa como drawMap.jpg. Una vez el archivo modificado se actualiza en el programa se actualizan todas las vistas que puedan verse modificadas por los cambios introducidos. A partir de este drawModificado, se genera un nuevo mapa modificado y una imagen satélite resultante.

Adicionalmente se dispone de un menú en la parte superior que permite modificaciones sencillas sobre la imagen de satélite original y que se verán actualizadas en el resto de las imágenes generadas. Los menús permiten guardar imágenes, rotar las imágenes a diferentes ángulos y realizar una simetría en espejo, tanto sobre el eje vertical como sobre el eje horizontal de la imagen. Además, se incluye un menú de ayuda con instrucciones sobre el programa.

En los sucesivos apartados se expondrá con detalle cada una de las decisiones adoptadas en el desarrollo del software, así como el funcionamiento interno de cada funcionalidad, exponiendo su arquitectura, sus puntos fuertes, sus debilidades y las líneas de trabajo futuras posibles.

5.2. Entorno de programación y exploración de los servicios de computación en la nube. Uso de GPU

Una vez determinada la arquitectura de red GAN que se va a utilizar y conociendo que adicionalmente se requiere una interfaz gráfica (GUI) para el desarrollo de la herramienta de software, se analizan los entornos de programación disponibles.

Atendiendo a los análisis sobre los lenguajes de programación realizados por diferentes investigadores podemos observar unas recomendaciones genéricas iniciales. Investigadores de la Universidad de Quebec (Alomari, El Halimi, Sivaprasad, & Chitrang, 2015), tras un análisis exhaustivo recomienda C# como el lenguaje más adecuado para el desarrollo de interfaces gráficas. Java presenta una mejor adaptabilidad a entornos de desarrollo web o de aplicaciones para dispositivos móviles gracias a su máquina virtual. C++ es especialmente adecuado para las conexiones con bases datos y consigue los mejores tiempos de ejecución del código. VB se posiciona como un lenguaje adecuado para el desarrollo de aplicaciones con interfaz gráfica y donde los procesos son guiados por eventos que se producen en dicha GUI. Finalmente, Python resulta adecuado para el prototipado rápido de aplicaciones al ser un lenguaje de alto nivel que permite un rápido desarrollo del código, si bien en contrapartida sus interfaces gráficas no serás especialmente sofisticadas ni el tiempo de ejecución será rápido.

La arquitectura Pix2Pix puede ser desarrollada en varios lenguajes de programación, si bien tal y como está descrita originalmente por sus autores (Isola, Zhu, Zhou, & Efros, 2017) se desarrolla en Python, bajo un entorno basado en *Keras* y por tanto en *TensorFlow*. Esta parece una opción adecuada, atendiendo a los recursos disponibles y al rendimiento de los algoritmos que utilizará la arquitectura GAN seleccionada.

Los primeros intentos de entrenamiento de los algoritmos se realizan en un entorno Python 3, bajo el paquete *Anaconda*, con un editor *PyCharm* y en un ordenador portátil con 4 gb de RAM y un procesador Intel i3 10th generación. Se constató que el proceso de entrenamiento se iniciaba si bien de forma extraordinariamente lenta. A partir de las primeras epoch entrenadas se estima que el entrenamiento de 2000 *epoch* superaría las 48 horas de entrenamiento. En la práctica se observa que el entorno de programación no es capaz de llegar hasta esas 2000 *epoch*, generando siempre errores de forma prematura, habitualmente asociados a la falta de memoria RAM. Podría ser posible depurar el algoritmo optimizando el uso de RAM al máximo e intentar este tipo de entrenamiento, si bien se optó por buscar un equipo más adecuado gracias a los servicios disponibles en la nube.

Dentro de los servicios disponibles en la nube se analizan dos opciones, por un lado, las máquinas virtuales *EC2* del servicio *AWS* y por otro lado los cuadernos basados en *Jupyter* de *Google Colab*.

El servicio EC2 de AWS permite la ejecución en la nube de un entorno Linux con prestaciones seleccionables bajo demanda. Cabe destacar que la potencia máxima disponible es muy superior a la de cualquier otro proveedor de servicios de computación en la nube de los consultados. En este caso es posible la puesta en servicio de una instancia p4d.24xlarge con 96 CPUs virtuales, 1152 GB de memoria RAM, 8 unidades GPU Tesla v100 con 600 Gb/s de transferencia de datos entre ellas, y un acceso a internet de 19 Gbps. El coste de esta instancia es de 32.77 $ / h. Si bien es una opción muy adecuada para el entrenamiento intensivo de algoritmos para herramientas comerciales se optó por explorar otras opciones con un coste menor, aún a costa de prestaciones más bajas.

Google Colab permite la computación en la nube a través de cuadernos Jupyter bajo un lenguaje de programación Python. Bajo una suscripción a Google Colab Pro, con un coste de 9.90$ mensuales permite la ejecución en la nube de cuadernos Jupyter en equipos Linux con 16 Gb de memoria RAM y un procesador GPU v100 Tesla de NVidia. Las limitaciones consisten en un máximo de 24 horas por instancia y un uso razonable de los recursos, siendo además que la disponibilidad no queda garantizada. Por otro lado, permite el acceso a Google drive, pudiendo disponerse bajo suscripción de una capacidad de almacenamiento de hasta

2 TB. Es posible adicionalmente y de forma puntual acceder a una capacidad RAM ampliada, según disponibilidad y bajo demanda. Los entrenamientos en Google Colab PRO han resultado satisfactorios, con tiempos de ejecución del orden de las 8 horas, siempre con la precaución de mantener las necesidades de RAM, que han alcanzados los 14.8 GB, siempre por debajo del límite de 16GB del sistema.

El desarrollo del presente software tiene carácter experimental y puede servir para futuros desarrollos. Se prevé su publicación en código abierto bajo licencia Creative Commons, debiendo por tanto estar bien documentado y ser accesible para la comunidad científica. De acuerdo con los estudios que realiza periódicamente la consultora tecnológica RedMonk (RedMonk analyst firm, 2021), para el primer cuarto de año de 2021, los lenguajes de programación más populares, tanto en los repositorios como en los foros especializados de internet son Javascript, Python y Java. Respecto a Javascript se centra especialmente en soluciones web, por lo que Python se posiciona como el lenguaje de referencia en desarrollos genéricos. Como bien se comentó al principio de este apartado es un lenguaje muy adecuado para desarrollar con éxito un prototipo. Este ranking puede observarse en la Figura 19.

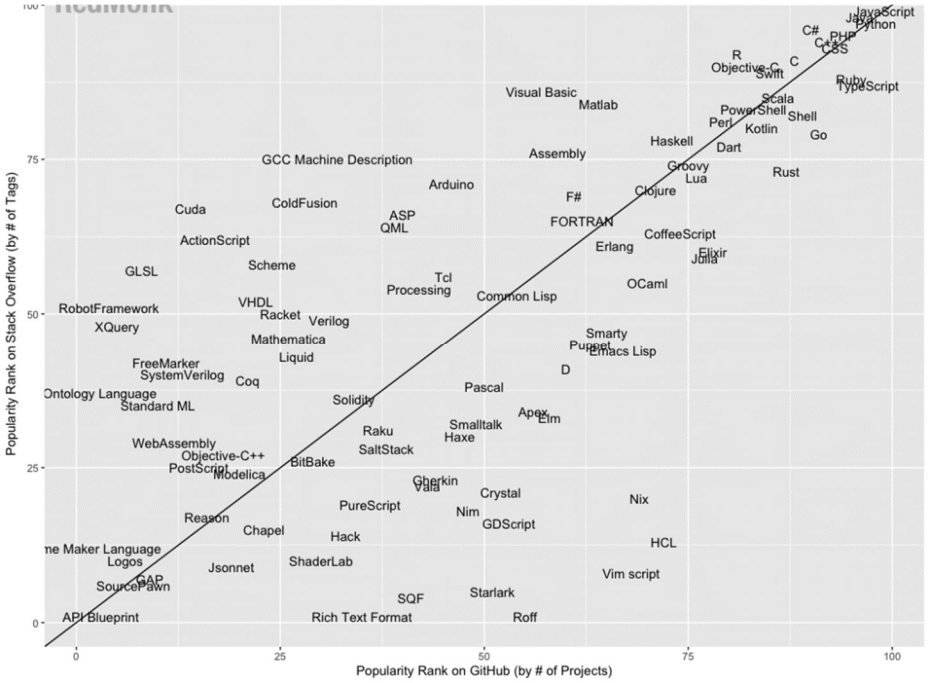

Figura 19. *Popularidad de los lenguajes de programación en GitHub y StackOverflow* (RedMonk analyst firm, 2021)

Por último, cabe destacar la importancia de las preferencias y conocimiento del autor en la selección del lenguaje de programación. En este sentido, se ha optado por Python, siendo un lenguaje de programación conocido por el desarrollador del presente estudio. Además, se ha valorado la gran variedad de código publicado, así como la disponibilidad de documentación,

de librerías y de foros de resolución de dudas. Tanto la documentación como los foros han sido claves en el desarrollo del software, pudiendo de esta forma resolverse funciones que a priori hubieran supuesto un esfuerzo no relacionado con el objeto del trabajo. Un ejemplo de esto ha sido la posibilidad del uso de librerías numpy para trabajar con arrays, Keras para el trabajo con TensorFlow y OpenCV para el tratamiento de imágenes.

De acuerdo con todos los criterios anteriormente expuestos se ha realizado un comparativo expuesto en la tabla 2 siguiente.

Tabla 2. Selección del lenguaje de programación

Recomendación expertos	Lenguaje	Importancia
- Eficacia de computación	C++	Media
- Mejor interfaz GUI	C#	Media
- Portabilidad Móvil	Java	Baja
- Desarrollo Prototipo funcional	Python	Alta
Otros criterios		
- Pix2Pix y Keras / TF	Python	Alta
- Computación en la nube	Python	Alta
- Conocimiento del entorno programador	Python	Alta
- Distribución Académica	Python	Alta
Lenguaje seleccionado	Python	

Elaboración propia.

Se ha optado por tanto por el desarrollo de la aplicación en Python 3, con el uso de librerías numpy, Keras, TensorFlow, Tkinter y OpenCV entre otras. Los entrenamientos de las redes neuronales se han realizado en una máquina virtual del servicio Google Colab Pro, con 16 GB de RAM y una GPU Tesla v100, siendo la capacidad límite, pero aceptable. El desarrollo de la aplicación se ha realizado en Python 3, utilizando un editor PyCharm y un entorno Anaconda en un ordenador portátil con 4 GB de RAM y un procesador i3 10th. Se recomienda este entorno para líneas futuras de trabajo, si bien para una herramienta comercial donde pudieran entrenarse redes neuronales con muy altos requerimientos de computación debería explorarse la mayor capacidad de las máquinas EC2 de AWS con hasta 8 GPU en paralelo.

5.3. Dataset y preprocesado de imágenes

Como se ha indicado en el análisis realizado en el apartado 2, se ha seleccionado un dataset formado por dos conjuntos de 1096 imágenes donde cada uno de los dos conjuntos incorpora imágenes pareadas con su correspondiente mapa y fotografía aérea. Se trata de imágenes de 1200 x 600 píxeles en color RGB, consistiendo por tanto en arrays bidimensionales de dimensiones 1200 x 600 con tres canales de profundidad tomando valores entre 0 y 255.

Para el entrenamiento hemos optado por cargar todas las imágenes al entorno de programación, siendo suficiente el RAM disponible y acelerándose de esta forma el proceso de entrenamiento. Cabe destacar que Google Colab se ha mostrado muy eficiente en el tratamiento de tensores, pero en cambio es lento en la carga de ficheros, imágenes en este caso, desde Google drive. En la Figura 20 se observa la función de carga de imágenes.

```
def load_2images(path, size=(256,512)):
  sat_list, map_list = list(), list()
   # enumerate filenames in directory, assume all are images
  i=0
  for filename in listdir(path):
    if(i%100==0):
      print(i,"/loaded")
    i+=1
    # load and resize the image
    pixels = load_img(path + filename, target_size=size)
    # convert to numpy array
    pixels = img_to_array(pixels)
    # split into satellite and map
    sat_img, map_img = pixels[:, 0:256],pixels[:, 256:512]
    sat_list.append(sat_img)
    map_list.append(map_img)
  return [asarray(sat_list), asarray(map_list)]
```

Figura 20. *Función de carga de imágenes* (Elaboración propia)

El proceso consiste en la carga de la imagen 1200x600 pixeles, en un array 512x256 y a partir de aquí la división de la imagen en un array 256x256 con la imagen satélite y otro 256x256 con la imagen mapa. Estos arrays tendrán tres canales de color con valores entre 0 y 255.

Las redes neuronales y específicamente las arquitecturas GAN pueden tender a inestabilidades durante su entrenamiento, siendo una buena práctica normalizar los parámetros de entrada a valores entre -1 y 1. En este caso, se diseñan las funciones indicadas en la Figura 16 para normalizar las imágenes de entrada desde el rango [0-255] hasta el rango [-1,1]. Cabe destacar además el uso de funciones de activación Relu, LeakRelu y Tanh como se profundizará en el siguiente apartado. Las funciones de normalización utilizadas se muestran en la figura 21.

```
def norm1_to_image255(X1):
    X1 = (X1 * 127.5) + 127.5
    return X1

def image_to_norm1(X1):
    X1 = (X1 - 127.5) / 127.5
    return X1
```

Figura 21. *Funciones de normalización* (Elaboración propia)

La arquitectura pix2pix es una arquitectura genérica de traducción entre dos campos de imágenes, por lo que las funciones de normalización descritas anteriormente se utilizarán en todas las conversiones realizadas en el presente trabajo. Cada conversión se describe de forma pormenorizada en el apartado correspondiente, si bien abarcan las siguientes tipologías:

- Foto satélite a mapa -> denominada Sat2Map
- Mapa a foto satélite -> denominada Map2Sat
- Mapa a mapa sostenible -> denominada map2emap
- Mapa a espacio de manchas de colores -> denominada map2draw
- Espacio de manchas de colores a mapa -> denominada draw2map

5.4. La red neuronal generativa. Encoder – Decoder

Una arquitectura GAN se basa en una red neuronal generativa y en una red neuronal discriminativa enfrentadas entre sí en un juego de suma cero. Mientras que la red neuronal discriminativa intentará determinar eficazmente si la imagen que se le muestra es real o generada, la red neuronal generativa intentará crear imágenes que difícilmente puedan distinguirse de las reales. En el presente apartado describiremos detalladamente la arquitectura de la red generativa utilizada, para en apartados posteriores profundizar en su antagónica red discriminativa y su entrenamiento conjunto configurando la GAN completa.

Una red neuronal generativa es aquella capaz de generar registros nuevos, no pertenecientes al dataset de entrenamiento. Para explicar este concepto y su aplicación al presente software haremos una breve introducción desde las primeras redes generativas o autoencoders hasta las tipologías U-Net (Ronneberger, Fischer, & Brox, 2015) como las utilizadas en el presente desarrollo.

Imaginemos que queremos entrenar una red neuronal generativa para crear nuevos cilindros. Disponemos de un dataset de 8 cilindros que los representamos en azul en la figura 22. Vemos no obstante que es posible generar otros cilindros a partir de los 8 indicados. En el espacio de dos dimensiones representado, es posible localizar nuevos puntos, con valores de altura y radio del cilindro, que generarán de forma eficaz nuevos cilindros que no pertenecían al dataset original.

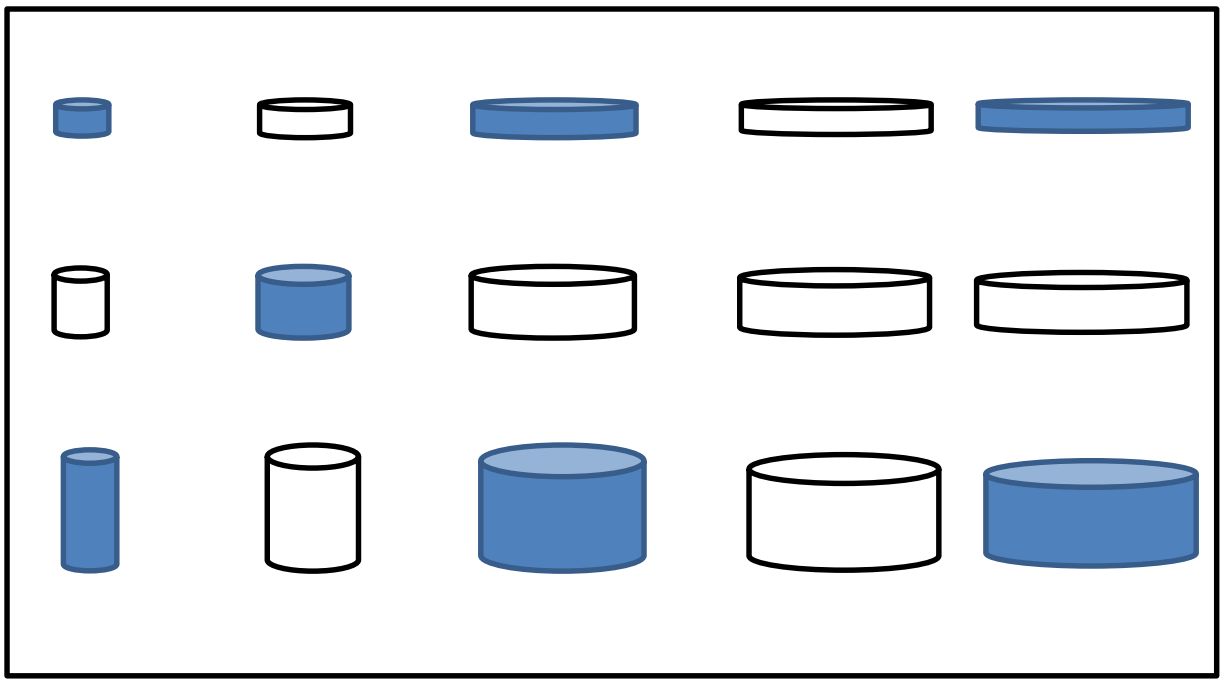

Figura 22. *Funciones de normalización* (Elaboración propia)

El objetivo de esta sencilla red neuronal generativa será la de constituirse como una función capaz de traducir un punto del espacio latente (radio, altura) y generar un cilindro f (radio, altura) que sea indistinguible del dataset original.

En el caso de dataset más complejos, como un conjunto de imágenes, este planteamiento se vuelve más complejo. Es necesario que el modelo sea capaz de inferir información de una forma más compleja, teniendo en cuenta no sólo el color o el brillo sino también la posición de los píxeles respecto a los próximos. Es decir, será necesario extraer características. Para la extracción de características se utilizará el concepto de red convolucional, capaz de extraer características entre conjuntos de píxeles. La figura 23 ilustra el funcionamiento de una red convolucional, donde la aplicación de un filtro de dimensión nxn es capaz de generar diferentes canales sobre los que pueden extraerse diferentes características. Estas convoluciones se aplican de forma sucesiva, de tal manera que la red neuronal puede inferir en las primeras capas información más genérica y en las capas más profundas es capaz de detectar patrones más sutiles o complejos.

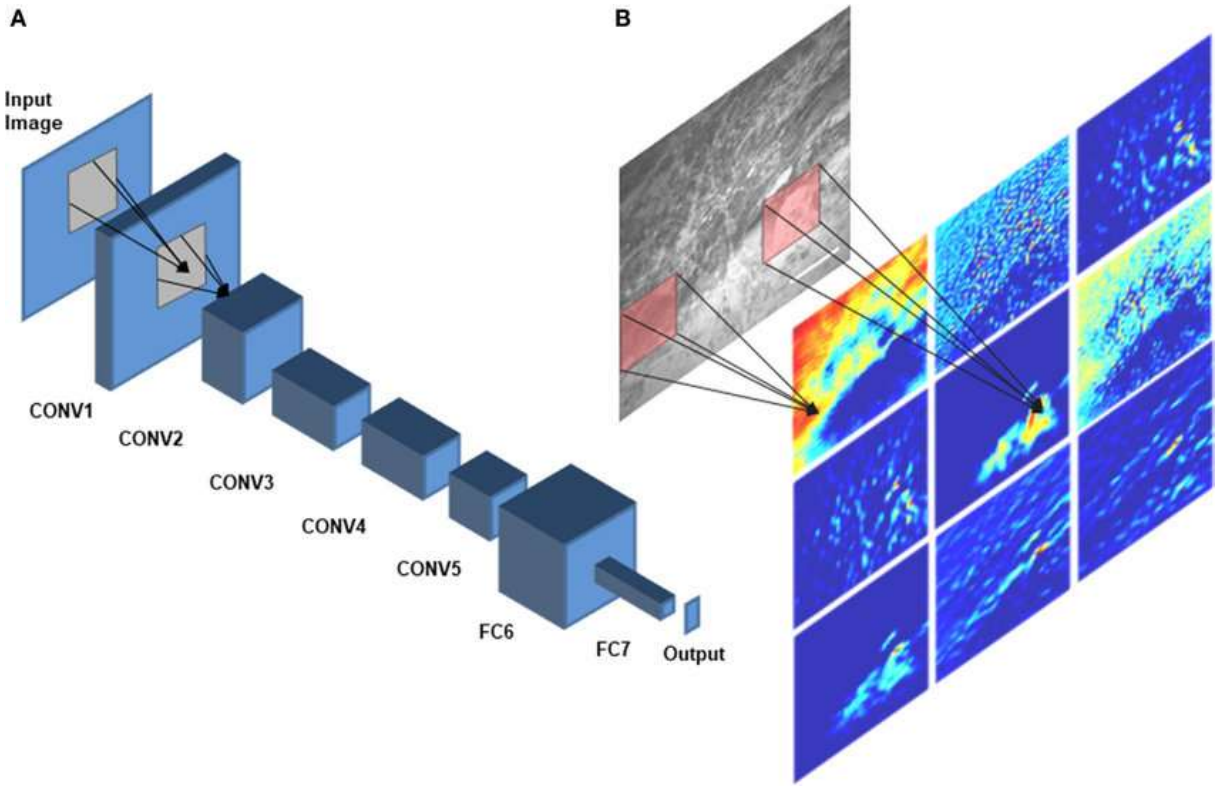

Figura 23. Red neuronal convolucional (Izadyyazdanabadi, y otros, 2018).

En el caso de nuestro generador utilizamos hasta 7 capas de convolución en el encoder, la primera de ellas con 64 filtros 4x4 y las siguientes con 128, 256 y 512 filtros las últimas. El decoder, como veremos más adelante será simétrico en configuración al encoder. Observamos la complejidad exponencial que añade que cada una de las capas aplique a la capa anterior hasta 512 filtros, generando canales adicionales. Para el encoder-decoder utilizado se han cuantificado un total de 54.429.315 parámetros.

Durante el entrenamiento de la red GAN se han obtenido imágenes de las capas intermedias del encoder, de tal forma que es posible intuir la complejidad en la que la red convolucional codifica y procesa la información. En la figura 24 es posible observar algunas de estas capas intermedias.

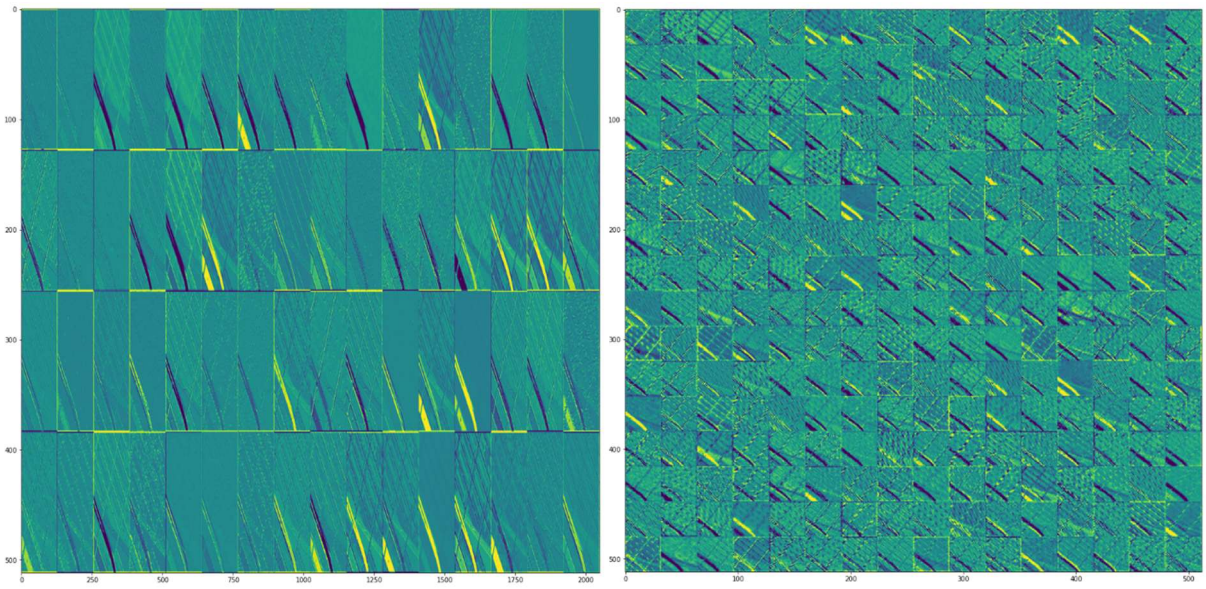

Figura 24. *Capas intermedias del encoder de la red entrenada sat2map.h5* (Elaboración propia)

Para la red generativa se utiliza una arquitectura convolucional encoder – decoder con las capas interconectadas. Esta tipología se denomina U-Net (Ronneberger, Fischer, & Brox, 2015) y aporta varias ventajas sobre los encoder-decoder sin interconexión directa entre capas. El encoder consiste en un conjunto de capas convolucionales seguidas de una batch normalization. El conjunto de capas convoluciones es capaz de inferir información durante el entrenamiento mientras que la capa batch normalization (Ioffe & Szegedy, 2015) contribuye a la estabilidad del modelo. El decoder es un conjunto de capas simétrico al encoder, en este caso con capas upsampling y deconvoluciones. De esta forma el conjunto es capaz de alcanzar en su punto medio una capa de estrangulamiento y a partir de aquí reconstruir la imagen de nuevo.

La tipología generativa del encoder-decoder reside en su capacidad de codificar una imagen de un dominio, por ejemplo, las ortofotos del satélite y reconstruirla en otro dominio, por ejemplo, los mapas. La tipología U-net seleccionada, tal y como recomiendan los autores del algoritmo Pix2Pix (Isola, Zhu, Zhou, & Efros, 2017) conecta de forma directa capas simétricas entre sí, permitiendo el flujo de información de forma directa entre capas simétricas. Esta conexión permite que algunas características que pueden tener una correlación muy directa entre el input y el output puedan transferirse de forma más directa en la red, consiguiendo mejores resultados. En la figura 25 observamos un esquema de una red convolucional, de tipología encoder – decoder, donde además se presentan interconexiones entre capas simétricas. En el punto medio es posible observar la capa de estrangulamiento. Adicionalmente, en la misma figura se ha representado un esquema del encoder-decores utilizado en el presente desarrollo, pudiéndose observar las similitudes en su arquitectura.

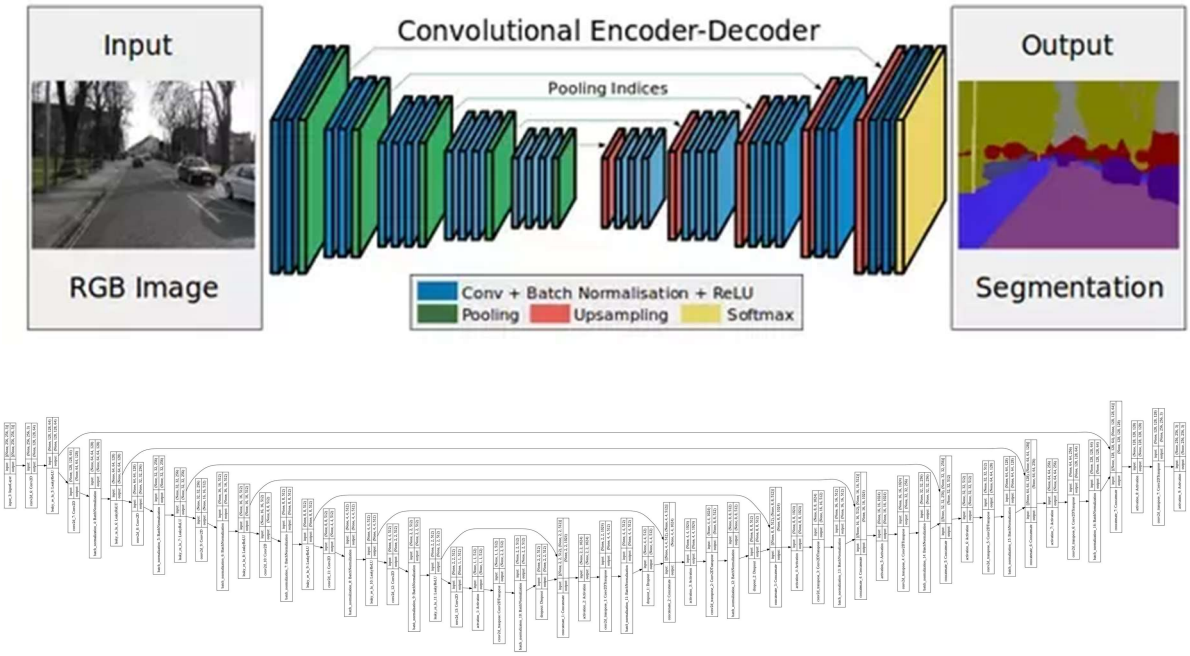

Figura 25. *Esquema de una red AlexNet generativa convolucional (arriba)* (Izadyyazdanabadi, y otros, 2018) *y esquema del encoder-decoder U-net utilizado en el presente desarrollo (abajo)* (Elaboración propia)

Cabe destacar que la capa de estrangulamiento presenta la forma de representación más abstracta de todas las capas intermedias que se han observado en el modelo. Es posible observar un ejemplo del propio encoder-decoder en la figura 26.

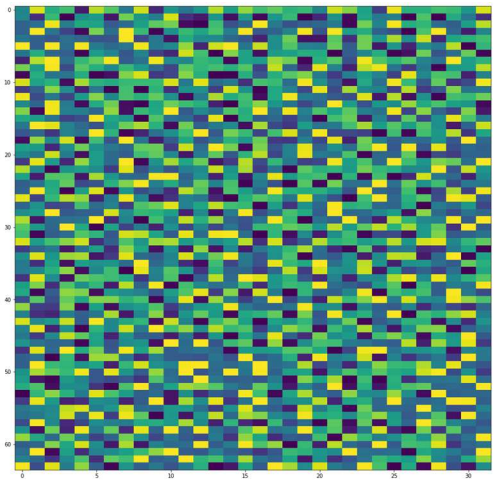

Figura 26. Imagen intermedia de la capa de estrangulamiento encoder-decoder (Elaboración propia)

Respecto a las funciones de activación utilizadas, tal y como recomiendan los autores de la arquitectura GAN pix2pix (Isola, Zhu, Zhou, & Efros, 2017) se utiliza una función ReakyRelu en cada capa del encoder, una función Relu en cada capa del decoder y una función tanh en la última capa del modelo.

De acuerdo con los autores de la arquitectura pix2pix (Isola, Zhu, Zhou, & Efros, 2017) se observa que el uso de una función de activación tanh en la última capa, que genera valores acotados entre [-1,1] permite al modelo un aprendizaje más rápido, especialmente en lo referido a la saturación de color.

Por último, cabe destacar que para que un generador sea capaz de crear nuevas imágenes es necesario introducir al mismo cierta aleatoriedad. Existen modelos GAN que habitualmente añaden como input un vector aleatorio, es decir ruido. En cierto modo aquí reside una de las principales diferencias entre una GAN y una conditional GAN (cGAN).

El generador (f) de una GAN es entrenado para ser capaz de generar imágenes realistas (Y) a partir de un ruido aleatorio (z). Esto podría representarse como f(z)=Y. Las redes GAN condicionales, además del ruido aleatorio (z), entrenan el modelo con la imagen del dominio que consideramos input (x), obteniendo una imagen del dominio output (Y). Esto podría representarse como f (z, x) =Y. En el caso de la arquitectura utilizada, se trata de una conditional GAN donde el ruido z no lo introducimos directamente en la entrada sino de forma distribuida a lo largo de todo el modelo del encoder. La introducción de aleatoriedad al modelo se realiza mediante capas dropout (Hertz, Krogh, & Palmer, 1991). Esta técnica consiste en omitir aleatoriamente neuronas durante el entrenamiento, permitiendo así a la red evitar el overfitting o sobreajuste, además de disponer del caos necesario para generar nuevas imágenes. Se trata por tanto de un encoder-decoder u-net estocástico.

El código se ha desarrollado en Python, utilizando Keras como librería para redes neuronales, combinando funciones propias con una fuerte inspiración en otros desarrollos Pix2Pix bajo Keras (Brownlee, 2019).

5.5. La red neuronal discriminativa.

Una arquitectura GAN se basa en una red neuronal generativa y en una red neuronal discriminativa enfrentadas entre sí en un juego de suma cero. Mientras que la red neuronal discriminativa intentará determinar eficazmente si la imagen que se le muestra es real o generada, la red neuronal generativa intentará crear imágenes que difícilmente puedan distinguirse de las reales. En el apartado anterior ha quedado descrita la red generativa. En el presente apartado describiremos detalladamente la arquitectura de la red discriminativa utilizada. De esta forma en apartados posteriores profundizaremos en su entrenamiento conjunto configurando la GAN completa.

La red discriminativa se trata de una red neuronal convolucional. Como entrada se le proporcionan dos imágenes emparejadas, una del dominio source y otra del dominio target. Es decir, si estamos entrenando el discriminador para pares de imágenes ortofoto-mapa, le proporcionaremos como input una ortofoto y un mapa. La salida de la red neuronal discriminativa será un valor numérico discreto entre [0,1], siendo 1 la determinación de que el par ortofoto-mapa es real o perteneciente a train y 0 si es fake o generado de forma artificial.

La figura 27 muestra un esquema de entradas y salidas de un modelo similar.

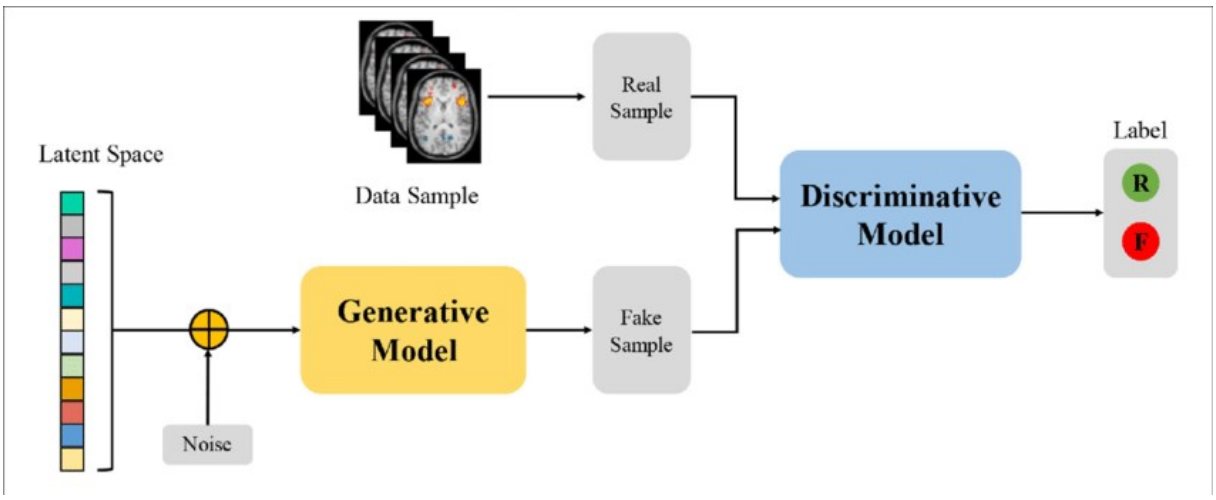

Figura 27. Esquema de entradas y salidas del modelo discriminativo (Zhang, Wang, Liu, & Zhang, 2020)

La red neuronal, don un par de imágenes como input y con un valor en el rango [0,1] como salida está formada por las siguientes capas:

- Capa input con la imagen source y capa input con la imagen target concatenadas entre sí. Cada imagen es 256x256x3 (RGB) normalizada de [0-255] a [-1,1]
- Convolucional 2D de 64 filtros 4x4 con función de activación LeakyReLu.
- Convolucional 2D de 128 filtros 4x4 con función de activación LeakyReLu seguida de una capa Batch Normalization.
- Convolucional 2D de 256 filtros 4x4 con función de activación LeakyReLu seguida de una capa BatchNormalization.
- Convolucional 2D de 512 filtros 4x4 con función de activación LeakyReLu seguida de una capa BatchNormalization.
- Convolucional 2D de 512 filtros 4x4 con función de activación LeakyReLu seguida de una capa BatchNormalization.
- Convolucional 2D de 1 filtros 4x4 con función de activación Sigmoid

Siendo estrictos el resultado de esta convolucional será un array de dos dimensiones, de cuya media podrá obtenerse el rango [0,1] que determinará si el par de inputs corresponden a un conjunto de imágenes real o generado (fake).

El discriminador es inicializado con una distribución de pesos normal con una desviación estándar de 0.2 y utiliza un optimizador Adam con un learning rate de 0.0002. Las funciones de pérdida se analizarán expresamente en apartados sucesivos.

5.6. La arquitectura GAN. Enfrentando al generador y al discriminador.

En apartados anteriores hemos descrito una red neuronal discriminativa (D), capaz de recibir como input un par de imágenes (source (x), target (Y)) y devolver un número (D (x, Y)) en el rango [0,1] donde 0 determina que el par de entrada ha sido generado y 1 determina que el par de entrada es real.

De la misma forma se ha descrito una red neuronal generativa (G), capaz de recibir como input una imagen source (x) y generar una imagen target (Y) que no pertenece al conjunto de entrenamiento original.

La red GAN será la encargada de enfrentar al generador y al discriminador en un juego de suma cero, de tal forma que el generador cada vez genere mejores imágenes, intentado engañar al discriminador. Por su parte el discriminador se volverá más efectivo detectando los fake del generador con cada iteración del entrenamiento.

Una arquitectura GAN se basa en una red neuronal generativa y en una red neuronal discriminativa enfrentadas entre sí en un juego de suma cero. Mientras que la red neuronal discriminativa intentará determinar eficazmente si la imagen que se le muestra es real o generada, la red neuronal generativa intentará crear imágenes que difícilmente puedan distinguirse de las reales.

El discriminador se entrenará aportándole como input de forma sucesiva un par de imágenes (source, target). Si el par de imágenes procede del dataset original, la etiqueta de salida del discriminador que le aportaremos como objetivo será 1, entrenándose así para detectar pares de imágenes reales. Si el par de imágenes procede del generador, la etiqueta de salida del discriminador que le aportaremos como objetivo será 0, entrenándose así para detectar imágenes falsas o generadas de forma sintética.

De esta forma cabe destacar las siguientes líneas de pseudocódigo que definen el entrenamiento del discriminador:

Para cada iteración:

Genera un par ImagenReal (source, target) de dataset

Genera un par ImagenFake (source, target) de G(source)

D_loss1=entrena Discriminador (ImagenReal,1)

D_loss2=entrena Discriminador (ImagenFake,0)

De esta forma el discriminador se entrena en cada iteración para ser capaz de diferenciar entre un par de imágenes real y uno sintético.

El generador (G) se entrena gracias al discriminador (D). El objetivo del generador será minimizar la función de coste del discriminador para imágenes fake marcadas como real. Para conseguir esto el generador se esforzará en crear imágenes más reales. Dada una imagen source, el generador deberá minimizar la función de coste entre la imagen generada y el target. De acuerdo con la figura 28, el Generador minimizará la función de coste que muestra las diferencias o error, dado x, entre G(x) e y. En la figura 28 se grafía los dos modos de funcionamiento de las entradas al discriminador, una de ellas sintética mediante el generador y la otra mediante una imagen real del dataset.

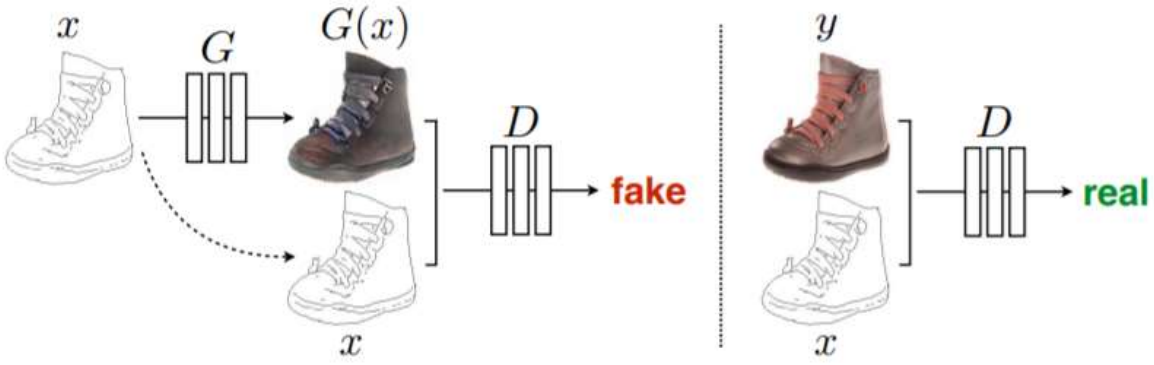

Figura 28. Esquema de entradas y salidas del modelo discriminativo (Zhang, Wang, Liu, & Zhang, 2020)

De esta forma una imagen (x) es suministrada al generador y al discriminador. La salida del generador G(x) será la otra entrada al discriminador que recibe (x, G(x)).

Para el entrenamiento del discriminador mostramos a continuación parte del pseudocódigo utilizado en el desarrollo:

Para cada iteración:

Genera un par ImagenReal (source, target) de dataset

Genera un par ImagenFake (source, target) de G(source)

G_loss = GAN.entrena(source, [1, target])

Donde el modelo GAN se ha montado a partir del generador y del discriminador de la siguiente forma:

GAN=(source,[d_model(source,g_model(source), g_model(source)]

Por tanto, el objetivo final será minimizar una función de coste denominada G_loss y que medirá mediante binary_crossentropy, dada como input una imagen source, la diferencia entre los dos siguientes outputs:

- Diferencia entre la salida del discriminador, cuando le aporto una imagen generada sintética y una matriz de 1. Es decir, en el caso ideal el discriminador reconocería como totalmente reales las imágenes generadas por el generador, siendo entonces esta función de coste igual a 0.

- Diferencia entre la imagen generada por el discriminador y el target real, para un source dado. Es decir, en el caso ideal el discriminador generaría una imagen idéntica al target, siendo entonces esta función de coste igual a 0.

El reparto de pesos entre las dos funciones anteriores es de [1,100] determinando de esta forma una función de coste del generador. La red GAN se representa en la figura 29, con parámetros de entrada source y target, un modelo que genera una imagen sintética a partir de source y una capa final que procesa source y g(source) determinando una matriz donde el discriminador determina si g(source) es fake o real.

Por tanto, durante el entrenamiento observaremos los valores de las tres siguientes funciones de coste:

d1_loss: Capacidad del discriminador para detectar imágenes reales. Un valor de 0 determinaría que el discriminador es capaz de identificar siempre una imagen real.

d2_loss: Capacidad del discriminador para detectar imágenes fake. Un valor de 0 determinaría que el discriminador es capaz de identificar siempre una imagen sintética generada por el generador.

G_loss: Capacidad del generador para crear sintéticas próximas al target y capaces de engañar al discriminador para que las clasifique como reales.

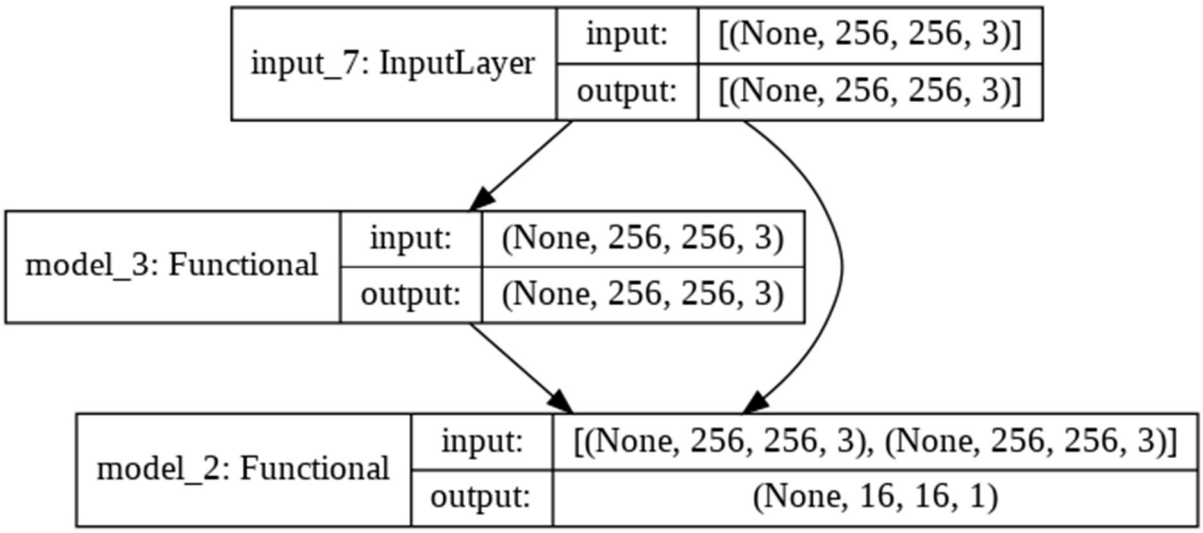

Figura 29. *Esquema de la red GAN que combina generador y discriminador* (Elaboración propia)

En apartados sucesivos observaremos la evolución de estos parámetros y su significado durante el entrenamiento de los diferentes modelos utilizados en el desarrollo de la presente herramienta de software.

El código se ha desarrollado en Python, utilizando Keras como librería para redes neuronales, combinando funciones propias con una fuerte inspiración en otros desarrollos Pix2Pix bajo Keras (Brownlee, 2019).

5.7. El espacio draw y edraw.

Como hemos adelantado en apartados anteriores la herramienta de software permite el trabajo en dos pipelines diferentes. El primero de ellos permite, desde una imagen satélite, obtener un mapa, un mapa mejorado con criterios ambientales y una nueva imagen satélite mejorada.

El segundo pipeline permite al usuario editar los mapas directamente de tal forma que puedan generarse imágenes satélites a demanda del diseñador. La representación en planos en formato Google es adecuada para una correcta visualización de un mapa especialmente para

orientarse durante el tráfico. No obstante, puede no ser la más práctica para dibujar bocetos que puedan convertirse en nuevos mapas y a su vez en ortofotos.

En 2018 de la mano de NVIDIA Corporation se presenta un trabajo (Wang, y otros, High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs, 2018) capaz de utilizar las redes cGAN para generar imágenes fotorrealistas a partir de un etiquetado de la imagen. Puede observarse en la figura 30, siendo este etiquetado un conjunto de manchas de color que responden a diferentes tipologías de la cara, como pueden ser la nariz, los labios o los ojos.

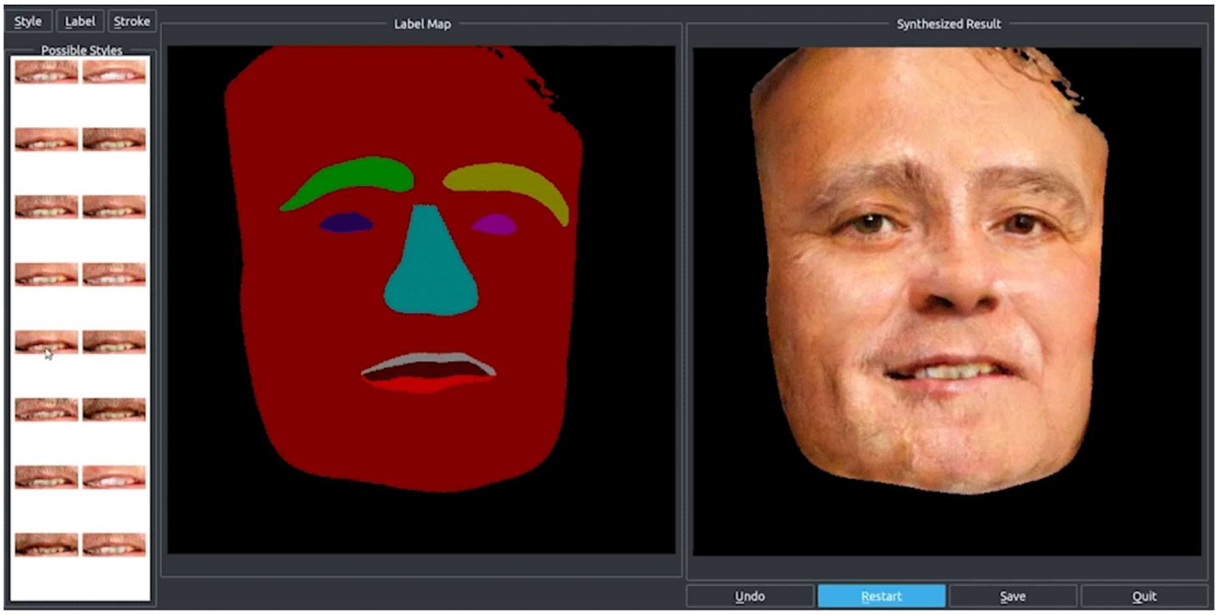

Figura 30. *Generación de imágenes fotorrealistas a partir de un etiquetado* (Wang, y otros, 2018))

Para generar mapas e imágenes fotorrealistas a partir de un conjunto de etiquetas dibujadas por el usuario hemos definido un conjunto de colores que representan los elementos más significativos que podemos identificar en una fotografía aérea; las zonas urbanas, las zonas verdes, el mar, las vías convencionales y las vías de alta capacidad.

En la figura 31 observamos una captura del programa donde se ha cargado una imagen original. A partir de esta imagen original, mediante la red neuronal entrenada denominada sat2map se ha generado un mapa al estilo Google. A partir de este mapa, mediante la función denominada Generador_de_calles se ha generado un conjunto de manchas de color. Ese conjunto de manchas de color ha sido fácilmente editado por el usuario, que ha añadido un espacio verde, un espacio con agua y una vía de alta capacidad. Esta imagen de etiquetas modificada se ha cargado de nuevo de forma automática en el programa que ha generado un

nuevo mapa y una nueva imagen satélite que ya incluyen las mejoras introducidas por el usuario.

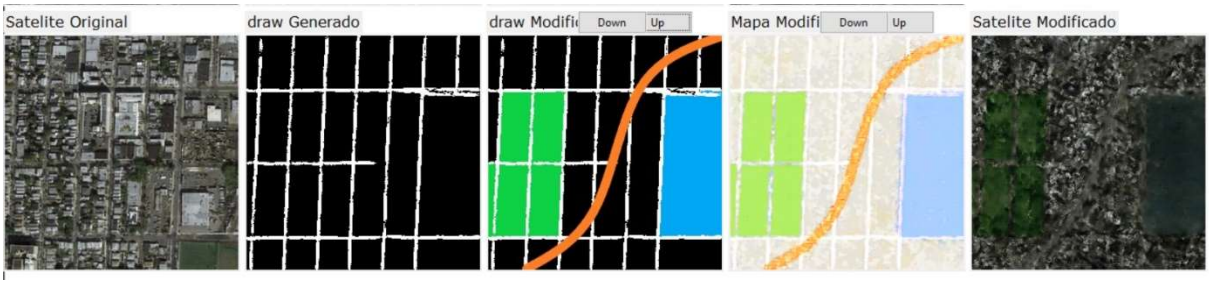

Figura 31. *Pipeline satélite -> draw -> draw modificado ad hoc -> mapa -> foto* (Elaboración propia)

La función detección de calles consiste en la aplicación de un conjunto de máscaras de la librería Open Cv sobre el plano al estilo Google generado. Se han definido rangos de color que captan de forma adecuada los diferentes elementos definidos posibilitando así su etiquetado. A continuación, en la Figura 32, se muestra parte del código de la función por considerarse ilustrativo de su función.

```
def deteccion_de_calles(image):
  calles=cv.inRange(image, (250, 250, 250), (255, 255,255))
  autoviasgc=cv.inRange(image, (220, 130, 20), (255, 180,60))
  autovias=cv.inRange(image, (245, 195, 60), (251, 201,66))
  mar=cv.inRange(image, (170, 202, 248), (181, 212,258))
  parque=cv.inRange(image, (198, 221, 167), (204, 227,173))
  mascara = cv.add(calles,parque)
  mascara = cv.add(mascara,autoviasgc)
  mascara = cv.add(mascara,autovias)
  mascara = cv.add(mascara,mar)
  target = cv.bitwise_and(image,image, mask=mascara)
  targetPIL = Image.fromarray(np.uint8(target)).convert('RGB')

  return targetPIL
```

Figura 32. *Función generar espacio draw* (Elaboración propia)

5.8. Map a emap y creación de un dataset adhoc.

El pipeline superior de la aplicación genera una imagen por satélite mejorada ambientalmente respecto a la original. Para ello se realizan las siguientes conversiones:

Ortofoto Original -> Generación de mapa mediante sat2map.h5 -> Generación de un e-mapa mediante map2emap.h5 -> Generación de una nueva ortofoto mediante map2sat.h5

La conversión de un mapa a otro mapa más sostenible ambientalmente se ha realizado mediante una red neuronal generativa, entrenada mediante una conditional GAN basado en una arquitectura Pix2Pix. Para que una red neuronal pueda generar un nuevo mapa más sostenible es necesario que durante su entrenamiento haya inferido conocimientos acerca de los criterios que pueden hacer más sostenible un urbanismo. Asumimos que se trata de un software experimental, por lo que no se pretende inferir

conocimientos complejos, sino más bien demostrar la capacidad de la red neuronal para inferir esta información y aplicarla de nuevo.

Para entrenar a esta red neuronal, que hemos denominado map2emap.h5 se ha generado un dataset ad-hoc. El dataset consta de 100 imágenes pareadas, mapa y emapa. Los emapas han sido retocados a mano y cumplen los siguientes criterios:

- Si el mapa original tiene suficientes espacios verdes, aproximadamente al menos un 20%, se pueden considerar aceptables y no es necesario modificar este aspecto.
- Si el mapa original tiene pocos espacios verdes se generarán nuevos espacios, intentando en la medida de lo posible que se generen espacios verdes amplios cubriendo manzanas enteras.
- Las zonas de costa deben quedar liberadas de urbanización, por lo que se tenderá a añadir espacios verdes junto a las masas de agua.
- Se podrán incorporar otros aspectos que el diseñador considere que mejoran el urbanismo en la zona.
- En principio se evita cambiar las alineaciones en general de las calles, así como las orientaciones de los edificios.

De esta forma se genera un dataset con los criterios de un diseñador, pero se constata que sería posible entrenar diferentes modelos a partir de diferentes dataset con diferentes criterios o estilos de diseñador. De esta forma sería posible explorar cómo diferentes estilos afectan al plano original. Incluso podrían aplicarse en cadena, atendiendo cada red neuronal a aspectos diferentes. La presente herramienta, como se ha comentado con anterioridad se ha centrado en la incorporación de zonas verdes, considerándose una característica ambiental clave en el urbanismo.

Lehmann formula 15 principios para un urbanismo verde, los criterios anteriormente adoptados corresponderían con el principio 5: “Las ciudades deben integrar jardines urbanos para maximizar su biodiversidad. Estas estrategias de retorno de la biodiversidad y las zonas verdes a la ciudad aseguran un enfriamiento natural de las ciudades” (Lehmann, 2010). En la figura 33 observamos como el algoritmo ha añadido zonas verdes junto a la costa, de acuerdo con los principios comentados anteriormente.

Figura 33. *Map2eMap.h5 sobre una imagen real* (Elaboración propia)

5.9. Métricas de error y entrenamiento de los modelos.

Como se ha descrito con anterioridad las métricas de error durante el entrenamiento son las siguientes:

d1_loss: Capacidad del discriminador para detectar imágenes reales. Un valor de 0 determinaría que el discriminador es capaz de identificar siempre una imagen real.

d2_loss: Capacidad del discriminador para detectar imágenes fake. Un valor de 0 determinaría que el discriminador es capaz de identificar siempre una imagen sintética generada por el generador.

G_loss: Capacidad del generador para crear sintéticas próximas al target y capaces de engañar al discriminador para que las clasifique como reales.

En todas ellas se utiliza la función Binary_Cross_Entropy de acuerdo con las recomendaciones de los autores de la arquitectura Pix2Pix. No se recomienda el uso de métricas L2 o error medio cuadrático por presentar problemas de artefactos en las imágenes generadas.

Cabe destacar que el entrenamiento en una red GAN presenta dos características singulares respecto de la interpretación de los datos:

1.- Los valores de error indicados con anterioridad se calculan para cada iteración en base a un generador y a un discriminador que mejoran, generalmente, de forma incremental. Es por tanto que no es directamente comparable un valor con el anterior, aunque pueden facilitar una idea del correcto entrenamiento del conjunto.

2.- El objeto es la generación de imágenes sintéticas, por lo que no es el objetivo que sean idénticas a imágenes reales, sino más bien que no puedan diferenciarse. Por tanto, al existir cierto caos propio del elemento generativo, es necesaria la opinión humana respecto a los resultados conseguidos.

En general, un buen síntoma de convergencia del entrenamiento de la red GAN consiste en un suave descenso continuado de g_loss lo que significa que gradualmente el generador es capaz de generar imágenes más parecidas al objetivo y que además son capaces de engañar al clasificador.

Por otro lado, d1_loss y d2_loss deben mantenerse equilibradas ya que un valor 0 de alguna de ellas significaría que el discriminador es capaz de diferenciar absolutamente todas las

imágenes reales de las fake dejando poco espacio a la mejora del generador. Esto sería debido a un desequilibrio entre la capacidad de discriminación y la de generación que deben entrenarse de forma pareja para conseguir una mejora continuada.

En la Figura 34 se observan los valores de entrenamiento obtenidos para la red map2sat, es decir, la que convierte mapas a imágenes satélites. Los resultados obtenidos con batches de 1 imagen han sido mejores que los obtenidos con batches de 64, tal y como adelantaban los autores del algoritmo pix2pix (Isola, Zhu, Zhou, & Efros, 2017). No obstante, este mejor comportamiento es observable tras una prueba de los dos modelos con varias imágenes.

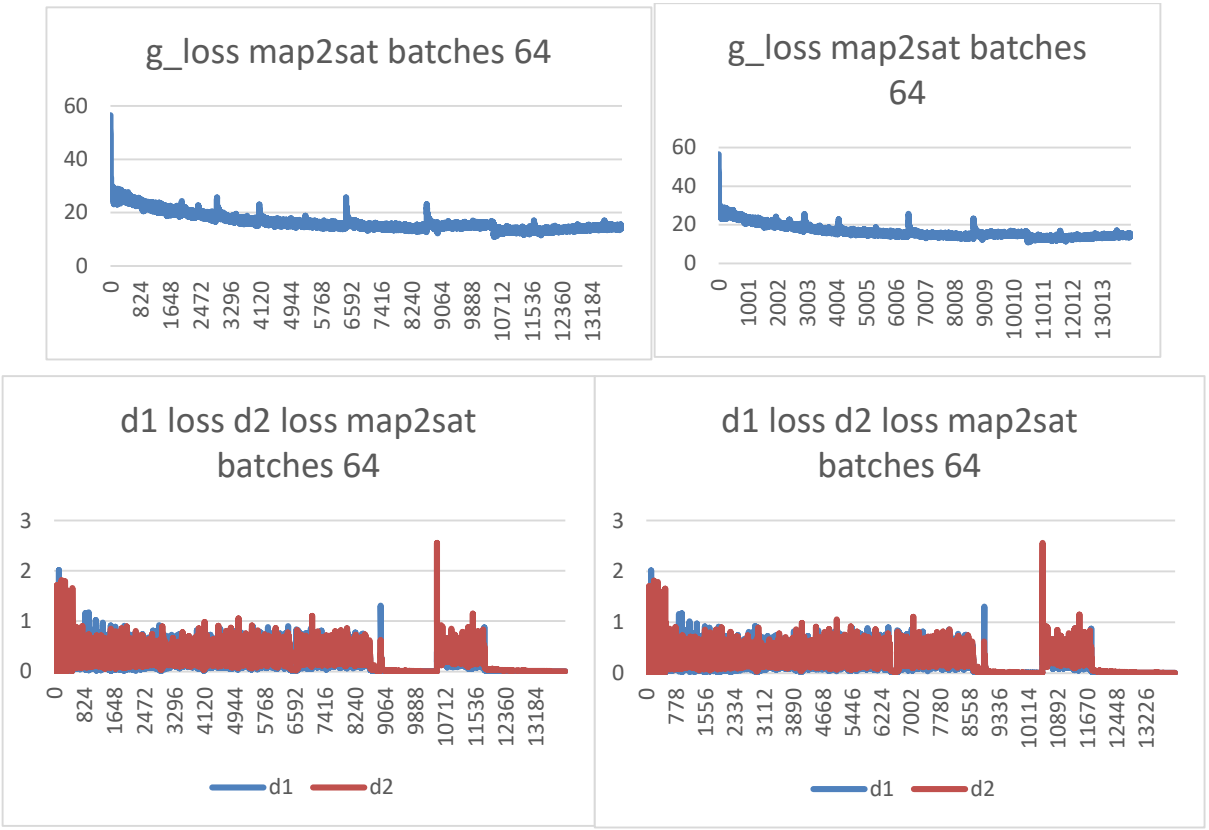

Figura 34. *Map2eMap.h5 sobre una imagen real* (Elaboración propia)

Los entrenamientos se han realizado para las 1096 imágenes del dataset y se han realizado pruebas de 200, 500 y 2000 epoch, verificándose que no se han encontrado mejoras significativas a partir de los 500 epoch. El entrenamiento se ha realizado en Google Colab Pro, en equipos con una GPU tesla v100, con una duración de entrenamiento aproximada de 6 horas por modelo de 2000 epoch.

5.10. Dropout y Data Augmentation.

Las redes neuronales generativas requieren de un vector ruido (z) habitualmente como input para el modelo ya que es necesario cierto caos para generar imágenes nuevas. En este caso no se introduce en el modelo este vector aleatorio de forma directa, sino que se realiza a través de un conjunto de capas dropout con valor 0.5 dentro del proceso de encode. Estas capas son desactivadas automáticamente por Keras en el modo predict, por lo que tienen efecto sólo durante el entrenamiento. Además, son suficientes para evitar el sobreajuste del modelo.

Respecto al data augmentation se ha desechado por los siguientes motivos:

- Cambio de escala. Puede afectar a la inferencia de información importante para el desarrollo urbanistico, ya que todas las imágenes del dataset tienen el mismo zoom y por tanto las dimensiones entre imágenes son equivalentes.

- Giros o flip. Se descarta su uso ya que la orientación puede ser importante, no obstante, en caso de utilizarse data augmentation su uso no sería especialmente invasivo.

5.11. El entorno gráfico GUI y la integración de los modelos ya entrenados.

La aplicación se ha desarrollado en un entorno de programación Python y puede ser utilizada directamente por un experto en inteligencia artificial con conocimientos de cógigo a partir del código fuente suministrado. No obstante, no todos los usuarios tienen la capacidad de trabajar directamente sobre el código fuente por lo que se ha incluido una interface gráfica que facilita el uso de algunas de las características.

Tras el análisis de diferentes librerías para el desarrollo de módulos de interface gráficos en Python se ha optado por Tkinter (Fredrik Lundh, 2009). Se trata, tal y como lo define Bezerra (Beniz & Espíndola, 2017), de un módulo de Python que proporciona una caja de herramientas para el desarrollo de interfaces gráficas, desarrollado en TCL y multiplataforma con soporte para Linux, Mac Os y Windows.

La interfaz gráfica se plantea lo más sencilla posible, facilitando su uso a usuarios sin conocimientos de programación, si bien para obtener la máxima versatilidad del proyecto de software es necesario la interacción entre el profesional y un ingeniero en inteligencia artificial. Se recuerda que se trata de un software experimental.

La interfaz gráfica se muestra en la Figura 35 y consiste en una única pantalla. Se dispone de un menú para la carga y guardado de la imagen original. También hay un segundo desplegable de menús que permite modificaciones básicas sobre las imágenes como la rotación, a diferentes ángulos y la simetría.

A continuación, identificaremos un cuadro de texto donde puede introducirse la dirección de la imagen a utilizar y un botón para proceder a su carga.

Posteriormente, sobre un elemento tk.canva se ubican 9 conjuntos de etiquetas, algunas de ellas de texto y otras de ellas de imagen. De esta forma es posible mostrar la salida de las diferentes transformaciones que realiza el software.

Figura 35. *Interfaz GUI del programa* (Elaboración propia)

Se incorporan dos conjuntos de botones denominados Download y Upload. Al pulsarse estos botones se llamará a una función interna save () o load () que permite tanto grabar imágenes a un directorio por defecto como subirlas y actualizar el resto. De esta forma es posible la edición mediante una herramienta externa de edición de imágenes consiguiéndose resultados y diseños adhoc.

Se ha descartado incorporar al programa una herramienta propia de edición de imágenes, al considerarse suficientemente fácil realizar este proceso mediante la propia herramienta de Windows. De cara a líneas de trabajo futuras podría incorporarse una herramienta de edición de mapasa específicas, solamente con los colores que reconoce el programa, facilitando así

la edición. También podría incorporarse la incorporación directa de imágenes desde servicios API como Google Maps.

Por último, cabe destacar como línea de trabajo futura la posibilidad de migrar la herramienta desde el escritorio a la web, posiblemente con un entorno dipo Django (Django Project, 2021).

5.12. Empaquetado del software para su distribución.

La aplicación se ha desarrollado en un entorno de programación Python y puede ser utilizada directamente desde la línea de comando Python ejecutando el programa eMapGAN.py. Adicionalmente a este archivo Python que es el encargado de ejecutar el programa existen otros archivos que recogen las diferentes funciones desarrolladas. Se describen a continuación:

- eMapGAN.py. Se trata de un fichero de código cuya misión es la de importar el resto de las dependencias y llamar a la ejecución de la aplicación.
- InterfaceGrafica.py. Se trata de un fichero de código que incorpora toda la gestión de la GUI del programa. Genera la ventana principal, los canvas, los menús, los botones y las imágenes.
- TraduceImagenes.py. Se trata de un fichero de código que incorpora todas las funciones de trabajo con redes neuronales o librerías gráficas que permiten la traducción de una imagen de un dominio a otro.
- Carpeta Models. Carpeta que incluye todos los modelos ya entrenados en formato .h5.
- Carpeta Images. Carpeta que incluye imágenes para nuevos entrenamientos y pruebas.
- Carpeta Notebooks. Carpeta que incluye los notebooks utilizados para entrenar las redes en la nube.
- Carpeta build. Carpeta que incluye todos los archivos intermedios generados por pyinstaller para la creación del paquete distribuible .exe.
- Carpeta dist. Carpeta con el software listo para su distribución bajo un entorno Windows.
- Carpeta doc. Documentación de ayuda y adicional.

Con el objeto de facilitar el uso del software se ha generado una versión distribuible en Windows para lo que se ha compilado un archivo .exe. Se recomienda el uso del software bajo el entorno Python, consiguiéndose un mejor rendimiento y aprovechando la actualización

y mejoras que puedan surgir sobre las diferentes librerías. Además, es previsible que la distribución .exe sea más propensa a fallos en determinados entornos.

Para la compilación se ha utilizado la librería PyInstaller (PyInstaller Development Team, 2021), una herramienta que es capaz generar un archivo ejecutable directamente en Windows que incorpora todas las librerías necesarias. Cabe destacar que es necesario que el archivo ejecutable tenga en su mismo directorio una carpeta con los modelos y otra carpeta con las imágenes. Es posible generar nuevos modelos y actualizarlos en la carpeta models introduciendo así nuevos conceptos o redes mejor entrenadas en el software desarrollado. En la figura 36 se observan tanto los archivos intermedios generados como el ejecutable final y sus carpetas.

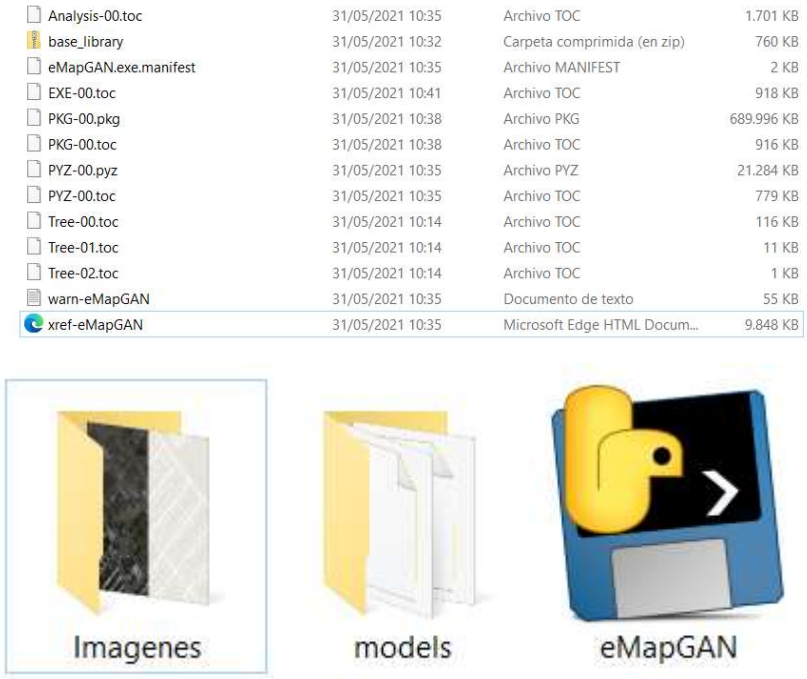

Figura 36. *Archivos generados durante la compilación y ejecutable .exe en entorno Windows* (Elaboración propia)

El anexo IV describe la guía de instalación de la herramienta, así como el acceso al código fuente de esta.

6. Evaluación

Para evaluar la usabilidad de la herramienta desarrollada, así como su utilidad como apoyo al profesional del urbanismo, se han realizado sesiones de testeo del software con diferentes expertos. A continuación, se expondrá la metodología utilizada para la evaluación, así como las conclusiones más relevantes obtenidas del trabajo conjunto con los expertos en diferentes campos. Se han seleccionado expertos de campos diferentes, aunque todos ellos afines a los usos del suelo y su representación gráfica, siendo potenciales usuarios de la herramienta desarrollada.

6.1. Metodología

La metodología planteada para la evaluación ha consistido en el diseño de un formulario, una selección de usuarios expertos en la materia, una sesión conjunta de prueba del software para finalizar completando el formulario y recopilando posibles mejoras o comentarios de los expertos.

Para el diseño del formulario de evaluación se ha utilizado la herramienta Google Forms, que permite la difusión online del formulario, así como la recopilación de las respuestas. El formulario desarrollado se incluye en el anexo II, si bien incluye los siguientes campos:

- En primer lugar, se incluye un texto introductorio que ubica en el contexto al usuario experto. De esta forma es consciente del uso experimental de la herramienta y de su diseño para fases iniciales del planeamiento, donde puede resultar una componente inspiradora adicional.
- Datos personales. Se solicita al experto su nombre, email de contacto y relación con el campo de uso de la herramienta. Estos datos no son publicados, cumpliendo la protección de datos de los profesionales, si bien quedan guardados por si pudieran ser de interés en desarrollos posteriores. También se solicita al experto que indique si ha participado o no en las reuniones previas de concepción del software, o bien si es su primer contacto con la herramienta.
- Forma de uso. Se pregunta al experto sobre la forma en la que ha utilizado la herramienta, bien en su propio ordenador o bien en un ordenador con el software ya instalado previamente. Se solicita además el tiempo dedicado a utilizar y reflexionar sobre la herramienta.
- Utilidad estado actual. Se solicita al experto que puntúe del 1 al 5 la utilidad actual de la herramienta para el desarrollo de su trabajo, siempre relacionado con los usos del suelo, sus infraestructuras y su forma de representarse gráficamente. Específicamente se pregunta además por los dos pipelines de trabajo, el centrado en las mejoras sostenibles automáticas y el centrado en la posibilidad de dibujar mapas a medida y obtener ortofotos.

- Utilidad mejorando prestaciones. Se pide también una valoración de la herramienta mejorando algunas prestaciones. Las capacidades de entrenamiento son limitadas, tanto en dataset disponible como en capacidad de computación, pero es posible invertir recursos en mejorar la resolución de la herramienta. Se pide a los expertos valoración sobre esta hipótesis.
- Potencial de utilidad futuro. Por último, se solicita una puntuación del 1 al 5 sobre el potencial de la herramienta en un plazo de 5 años en el dominio de su profesión. Tratándose de una aplicación de una tecnología reciente a un campo consolidado se considera que refleja la capacidad de la herramienta de mostrar su potencial.
- Mejoras. Se incorpora un apartado para que los expertos puedan incluir sus comentarios y propuestas de mejora.

Para la selección de los usuarios expertos se ha querido abarcar diferentes campos de las ciencias relacionados con los mapas y los usos del suelo, asumiendo que la herramienta puede resultar útil desde el punto de vista de profesionales diferentes. Por ello se han seleccionado expertos en urbanismo, en cartografía, en usos del suelo y en infraestructuras con gran impacto sobre el territorio. Se han seleccionado a personas con una titulación universitaria y una experiencia en grado doctor o en al menos 10 años en la profesión.

Para la evaluación se ha diseñado una sesión de trabajo con cada experto de 30 minutos, repartiéndose los tiempos de la siguiente manera.

Minuto 1 a 5. Se realiza una introducción sobre las redes neuronales, las redes generativas y las redes discriminativas. Se hace ver al experto que la herramienta introduce una herramienta no basada en reglas fijas, sino en redes neuronales capaces de aprender y de generar nuevas realidades a partir del conocimiento inferido.

Minuto 5 a 15. Se realiza una demostración de los dos pipelines de trabajo, con diferentes imágenes de forma guiada por el desarrollador del software.

Minuto 15 a 25. Se permite al experto el uso de la herramienta por sí mismo, o con la ayuda del desarrollador, a su elección y se resuelven todas sus dudas y comentarios.

Minuto 25 a 30. Se facilita al experto el formulario para la evaluación y se recogen sus propuestas de mejora o posibles líneas de trabajo futuras o campos de uso de la herramienta.

Tras las sesiones con los 5 expertos seleccionados se extraen las conclusiones sobre las valoraciones y las propuestas de mejora y líneas de trabajo futuras.

6.1. Evaluación con experto cartógrafo

Se realiza la evaluación con un doctor en geografía, experto en el desarrollo de portales cartográficos, especialmente referidos a las infraestructuras. La evaluación se realiza de forma telemática, durante una sesión de 30 minutos tal y como se ha definido anteriormente.

El experto entiende el funcionamiento de la herramienta y su utilidad, resaltando que resulta útil en fases que requieren inspiración y diferentes pruebas de concepto. En cierto modo la herramienta presenta un sandbox que permite experimentar con diferentes realidades posibles en un mismo territorio. Se muestra especialmente interesado por el espacio latente denominado eDraw, que indica, podría configurarse para otros conceptos como densidad de población, resultando muy útil para sus trabajos.

Asimismo, resalta la utilidad del módulo map2emap, ya que podrían entrenarse diferentes módulos para diferentes usos, como urbanismo antiincendios o urbanismo anti-inundaciones, y utilizar varios de estos módulos en cadena.

En general la puntuación de la herramienta es buena, con un 4/5 en su concepción actual y un 5/5 en una concepción que mejore la resolución. El potencial a 5 años de la herramienta dentro del campo de la cartografía y de los portales cartográficos lo valora en 5/5. El elemento sostenible resulta de especial interés ya que es necesario adecuar las infraestructuras a criterios sostenibles en un horizonte 2030 y 2050, especialmente en Europa.

6.2. Evaluación con arquitecto urbanista

Se realiza la evaluación con un arquitecto con más de 30 años de experiencia y que ha participado en varios planes de ordenación urbana. La evaluación se realiza de forma telemática, durante una sesión de 30 minutos tal y como se ha definido anteriormente.

El experto entiende el funcionamiento de la herramienta y su utilidad, entendiéndola como inspiradora en los momentos previos de diseño. También le resulta útil para ilustrar las diferentes soluciones de formas más comprensibles que los mapas convencionales.

El experto indica que el urbanismo es un proceso complejo, pero que esta herramienta no existía con anterioridad y por tanto puede ser una herramienta más para tener en cuenta. No es determinante ni sustituye procesos habituales del urbanista, pero puede complementar su trabajo siendo una herramienta novedosa.

En general la puntuación de la herramienta es buena, con un 3/5 en su concepción actual y un 4/5 en una concepción que mejore la resolución. El potencial a 5 años de la herramienta

dentro del campo de la arquitectura y el urbanismo lo valora en 5/5. Valora positivamente la posibilidad de proponer alternativas sostenibles. También la posibilidad de que el arquitecto urbanista entrene, junto con el experto en IA, redes que aporten ideas a otros urbanistas. Puede ser una forma nueva de transmitir conocimiento no basado en normas fijas sino más intuitivo.

6.3. Evaluación con ingeniero urbanista

Se realiza la evaluación con un ingeniero industrial que ha participado en diferentes desarrollos de ordenación urbana. También ha participado en diferentes promociones inmobiliarias, industriales y en evaluaciones de impacto ambiental. La evaluación se realiza de forma telemática, durante una sesión de 30 minutos tal y como se ha definido anteriormente.

El experto entiende el funcionamiento de la herramienta y su utilidad, especialmente aportando una forma de sistematizar ciertos procesos urbanísticos. Esto podría ser interesante como herramienta incluso de supervisión de planes en proceso de supervisión.

Propone como nueva funcionalidad que la herramienta pudiera utilizarse online y como plugin de autocad. De esta forma sería posible actuar sobre la edición de planos de forma más directa. La definición de espacios latentes personalizados es un trabajo interesante conjunto del experto en IA y del ingeniero para nuevos desarrollos relacionados con la movilidad o la energía.

En general la puntuación de la herramienta es buena, con un 3/5 en su concepción actual y un 4/5 en una concepción que mejore la resolución. El potencial a 5 años de la herramienta lo valora en 5/5.

6.4. Evaluación con experto en infraestructuras

Se realiza la evaluación con un experto en infraestructuras, adjunto a la dirección de un organismo para el desarrollo de corredores europeos de alta capacidad. Habitualmente su trabajo consiste en evaluar varias opciones de trazado, así como en comunicar esta información en el territorio. La evaluación se realiza de forma telemática, durante una sesión de 30 minutos tal y como se ha definido anteriormente.

El experto entiende el funcionamiento de la herramienta y su utilidad, viendo utilidades concretas, como plasmar en una ortofoto el posible impacto de diferentes alternativas de

infraestructuras sobre el territorio. En este sentido considera más versátil el fulo de trabajo que permite personalizar los mapas, si bien intuye utilidades para el modo automático.

También puede resultar de utilidad en zonas con mucha superficie donde no es posible por falta de inversión realizar estudios detallados, como pueden ser grandes zonas de África.

En general la puntuación de la herramienta es buena, con un 4/5 en su concepción actual y un 5/5 en una concepción que mejore la resolución. El potencial a 5 años de la herramienta dentro del campo del análisis de infraestructuras sobre el territorio lo valora en 5/5.

6.5. Evaluación con experto en valor y usos del suelo

Se realiza la evaluación con un arquitecto técnico y máster en valoraciones del suelo, experto en valoración, usos del suelo y estudio de promociones inmobiliarias. La evaluación se realiza de forma telemática, durante una sesión de 30 minutos tal y como se ha definido anteriormente.

El experto entiende el funcionamiento de la herramienta y su utilidad, no encontrando una utilidad directa que sustituya a ninguna de sus tareas de trabajo. No obstante, si tiene utilidad aprovechar algunas de las imágenes y mapas generados para complementar los estudios sobre promociones inmobiliarias. En todo caso, es una herramienta nueva, con posible utilidad esporádica, que en ningún caso sustituye a lo existente. Plantea la posibilidad de utilizarlo juntamente con sketchup, generando así modelos 3d.

En general la puntuación de la herramienta en su estado actual no es útil para el experto, con un 2/5 en su concepción actual, podría serlo con mayor resolución y versatilidad, con un 3/5 en una concepción que mejore estas funcionalidades. El potencial a 5 años de la herramienta dentro del campo del urbanismo referido a la promoción y a los usos del suelo, lo pondera en 4/5.

6.6. Conclusiones de la evaluación

Tras la evaluación concluimos que los expertos han entendido la funcionalidad y la solución propuesta por la herramienta. En términos generales la herramienta no sustituye a ninguno de los procesos habituales de los expertos, ya que se trata de una funcionalidad totalmente novedosa. En este sentido sí entienden que proporciona una herramienta adicional tanto en la toma de decisiones durante el diseño, como en la comunicación de los desarrollos urbanísticos.

En todos los casos proponen que resultara más útil si se pudiera entrenar con una mayor resolución y si se pudiera utilizar online, directamente sobre un visor de mapas o integrado como plugin de autocad.

Respecto al formulario, los expertos valoran de media la utilidad de la herramienta en su estado actual en 3.4/5.0, mejorándose hasta el 4.0/5.0 en caso de mejorarse la resolución o la integración web o con autocad.

Respecto al potencial de la herramienta a 5 años, los expertos consideran que existe un amplio margen para su uso, valorándolo en 4.8/5.0.

Se concluye por tanto que la herramienta de software presenta una usabilidad suficiente en el ámbito del urbanismo, las infraestructuras y la cartografía, si bien puede ser mejorada de forma significativa. Adicionalmente se considera que la herramienta ha cumplido además el objetivo de demostrar el potencial de la tecnología GAN en disciplinas que se expresan de forma gráfica como el urbanismo y la cartografía.

7. Conclusiones y trabajo futuro

A continuación, se exponen las conclusiones obtenidas y se detallan las líneas de trabajo identificadas que pudieran resultar de interés en aproximaciones posteriores al estudio realizado.

7.1. Conclusiones

La evolución en los últimos años de la capacidad de cómputo de los ordenadores, así como del volumen de datos disponibles ha permitido el desarrollo de aplicaciones basadas en redes neuronales. La disponibilidad de unidades GPU y TPU en la nube a bajo coste permite acceder de forma asequible a la experimentación con redes profundas convolucionales. El desarrollo de nuevas arquitecturas como las GAN permite generar nuevas imágenes, a partir de un modelo entrenado con otras imágenes existentes. La aplicación temprana de estas tecnologías a disciplinas consolidadas y cuya forma de expresión es gráfica, como la arquitectura y el urbanismo abre la posibilidad a explorar nuevas herramientas.

En este contexto se ha desarrollado una herramienta que genera nuevas realidades, expresadas en forma de mapa y de ortofoto, más sostenibles que la imagen original. Estas nuevas realidades se generan a partir de un conocimiento inferido desde un conjunto de mapas, en este caso la generación de zonas verdes, siendo capaces de transferir el conocimiento del profesional urbanista am modelo de inteligencia artificial. Tras la evaluación con los diferentes expertos, han comprendido el sentido y el uso de la herramienta y han concluido que podría ser una herramienta adicional en su proceso de toma de decisiones. Se considera por tanto que el objetivo final se ha cumplido, si bien la herramienta necesita evolucionar para resultar de más utilidad para los usuarios expertos.

La herramienta no es suficientemente versátil, ya que no funciona en tiempo real desde un proveedor de mapas online. Además, el entrenamiento se ha basado en un conjunto limitado de imágenes, y, por tanto, el modelo tiende a ajustarse a ese conjunto. Es necesario un experto en inteligencia artificial para entrenar nuevos modelos. A priori, y entendiendo que se trata de una herramienta experimental y para investigadores, esto no supone un inconveniente, pero un modelo más avanzado debería poder entrenarse con facilidad para cualquier ciudad que se seleccione.

Las limitaciones respecto a la memoria disponible, los recursos de computación y el tiempo de entrenamiento, limitado a 12 horas para las GPU en Google Colab, han generado modelos mejorables en resolución.

Las contribuciones del presente estudio a los campos de las redes neuronales convolucionales bajo una arquitectura generativa adversaria condicional y a las disciplinas de la cartografía y del urbanismo se resumen a continuación.

Se ha identificado una arquitectura GAN existente y se ha utilizado para entrenar modelos entre diferentes dominios de la imagen, como ortofoto, mapa, emapa y edraw resultando suficientemente versátil y asequible en términos de computación disponible en la nube. La tipología de GAN Condicional denominada pix2pix (Isola, Zhu, Zhou, & Efros, 2017) ha resultado adecuada para dataset entre 100 y 1000 imágenes con entrenamientos inferiores a 12 horas en GPU estándar de Google Colab y RAM hasta 16 Gb.

Se han definido dos flujos de trabajo que han resultado útiles para los usuarios expertos, uno de ellos automático, capaz de generar realidades más sostenibles, expresadas como mapa y como ortofoto simplemente introduciendo a la interfaz gráfica una ortofoto original. El segundo flujo de trabajo permite generar mapas y ortofotos a medida, para lo que se ha definido un sistema de colores útil para el diseñador y comprensible y entrenable para el modelo de inteligencia artificial.

Se ha identificado un conjunto de datos que ha permitido el entrenamiento del modelo, si bien resulta limitado y deberán realizarse entrenamientos adicionales con conjuntos de datos más extensos y heterogéneos.

Se ha implementado una interfaz de usuario mínima que permite el uso de la aplicación a un usuario sin conocimientos de programación. De esta forma se abre una ventana al uso de las redes GAN a profesionales de la cartografía y del urbanismo. Esta interfaz es limitada y meramente exploratoria, por lo que tiene un margen de mejora importante, incluso pudiendo llegar a ser una aplicación web o integrarse en herramientas como AutoCAD.

Se ha testeado la herramienta con usuarios expertos, que han entendido su sentido y la han considerado de posible utilidad. En ningún caso la herramienta sustituye a ninguna de sus dinámicas habituales de trabajo, pero entienden su utilidad y podrían utilizarla como un criterio más, especialmente en las fases iniciales que requieren de inspiración y en las fases de exposición, que requieren de abundante expresión gráfica. Todos los usuarios han considerado que la herramienta y su concepto tienen un alto potencial de desarrollo a medio plazo.

7.2. Líneas de trabajo futuro

Las redes generativas adversarias son una tecnología reciente y su aplicación a campos consolidados como la cartografía o el urbanismo es temprana. Es por tanto momento de testear diferentes soluciones y utilidades, inicialmente con modelos genéricos, pudiendo concretarse herramientas específicas en el futuro. De esta forma se entiende el software desarrollado y su evaluación con expertos como una forma de conexión entre esta disciplina de la IA y las disciplinas que se expresan en mapas.

El modelo capaz de generar nuevas realidades más sostenibles supone una forma de demostrar la forma de empaquetar conocimiento en una red neuronal, para su uso posterior. Todos los usuarios expertos han identificado un alto potencial en el concepto y en la herramienta, indicando todos ellos algunas líneas de trabajo futuras que se exponen a continuación.

En el campo de las infraestructuras es habitual la concertación con el territorio. Esto significa que se plantean diferentes alternativas de trazado, por ejemplo, para una autovía o una línea de ferrocarril, y existe una fase de valoración del impacto. Es necesario utilizar todas las herramientas al alcance del proyectista para poder explicar el impacto a todos los niveles, desde el residente en la zona, sin conocimientos técnicos, hasta las administraciones que autorizan. En este sentido resulta de especial utilidad el flujo de trabajo que permite dibujar diferentes mapas y generar su ortofoto. Esto permitiría mostrar de forma más intuitiva el impacto de la infraestructura. Como línea de trabajo se propone evolucionar la herramienta hacia una integración con portales cartográficos de tal forma que sea posible ver a diferentes niveles de zoom el impacto de una infraestructura en el rango desde los pocos metros hasta los 300 km.

Una segunda línea de trabajo futura planteada fue la posibilidad de utilizar la herramienta, con ajustes o modelos preentrenados de acuerdo con el resultado que se pretenda, para explorar zonas a bajo coste. Por ejemplo, urbanismos o análisis de forma muy preliminar en zonas muy extensas de África, donde el presupuesto del proyecto no posibilite invertir demasiado tiempo al detalle. Una herramienta como la planteada podría sistematizar el proceso.

Del trabajo conjunto con el experto en valoraciones y usos del suelo, se concluyó un interés en utilizar la herramienta para mostrar diferentes opciones futuras para desarrollos urbanísticos, como urbanizaciones. No tanto para la toma de decisiones, como en la expresión gráfica. En todo caso, requirió más resolución y exactitud en las imágenes generadas.

Conseguir esta resolución, mediante entrenamientos más largos, con dataset mayores y con diferentes configuraciones de capas es una línea de trabajo de mejora de la herramienta.

El usuario experto más relacionado con el desarrollo de conceptos experimentales y la investigación trabaja en portales GIS desde su posición de doctor en geografía. Del trabajo conjunto resultó especialmente interesante el módulo map2emap, capaz de inferir conocimientos en este caso sobre zonas verdes, y aplicarlos a nuevos mapas. Este concepto en sí mismo ya resultó de utilidad en su campo de trabajo y además pudimos concluir que una línea de trabajo futura consistiría en desarrollar diferentes modelos similares. Por ejemplo, un modelo capaz de generar mapas de urbanismos anti-incendio o anti-inundaciones. Incluso podrían aplicarse en serie sobre un mapa original.

Los usuarios expertos relacionados directamente con el urbanismo plantearon el uso de la herramienta tal y como está diseñada en conceptos previos del diseño, donde es necesaria la inspiración. Se concluyó la posibilidad de entrenar modelos diferentes con ideas de diferentes arquitectos, relacionadas con las zonas verdes, las orientaciones, la tipología de las manzanas, la ubicación de colegios y centros de salud, y de esta forma poder aplicar uno u otro a conveniencia en esa fase de diseño. También se planteó la opción de utilizar el discriminador, bien entrenado, para validar de forma rápida y previa planeamientos de forma sistemática, por ejemplo, desde las administraciones públicas.

En conclusión, el potencial de la herramienta es elevado, y ha cumplido su objetivo como propuesta de unión entre las redes GAN y las disciplinas basadas en la representación cartográfica, siendo capaz además de inferir conocimientos complejos como la ubicación de zonas verdes y, mediante modelos entrenados para tal fin, proponer nuevas realidades. Este punto de unión entre disciplinas abre la puerta a diferentes líneas de trabajo como las planteadas. Hay por tanto por delante un intenso trabajo conjunto entre profesionales de la inteligencia artificial y profesionales del urbanismo, la cartografía y la sostenibilidad.

8. Bibliografía

Alomari, Z., El Halimi, O., Sivaprasad, K., & Chitrang, P. (2015). Comparative Studies of Six Programming Languages. *arXiv:1504.00693*.

Anttiroiko, A.-V. (2012). Urban Planning 2.0. *International Journal of E-Planning Research (IJEPR), 1*, 16–30. doi:https://doi.org/10.4018/IJEPR.2012010103

Beniz, D., & Espíndola, A. (2017). Using Tkinter of python to create graphical user interface (GUI) for scripts in LNLS. *11st International Workshop os Personal Computers and Particle Accelerator Controls.* Geneva: JACoW. doi:https://doi. org/10.18429/JACoW-PCaPAC2016-WEPOPRPO25

Brownlee, J. (2019). Part 6. Image Translation. En J. Brownlee, *Generative Adversarial Networks with Python.* Machine Learning Mastery.

Chaillou, S. (2019). *AI + Architecture Towards a New Approach.* Harvard University.

Chaillou, S. (2019). *Archigan: a generative stack for apartment building design*. Obtenido de NVIDIA.Developer: https://developer.nvidia.com/blog/archigan-generative-stack-apartment-building-design/

Choi, Y., Choi, M., Kim, M., Ha, J.-W., Kim, S., & Choo, J. (2018). StarGAN: Unified Generative Adversarial Networks Multi-Domain Image-Image translation. *In Proceedings of the IEEE conference on computer vision and pattern recognition* (págs. 8789-8797). Salt Lake City, UT, USA: IEEE.

del Campo, M., & Manninger, S. (2020). Imaginary Maps - a Posthuman Urban Design Method based on Neural Style Transfer. *Utopia vs. the city P+ARG Biennial Conference.* Michigan, USA: Taubman College for Architecture and Urban Planning.

del Campo, M., Carlson, A., & Manninger, S. (2020). Towards Hallucinating Machines - Designing with Computacional Vision. *International Journal of Arquitectural Computer*, 1478077120963366.

Django Project. (2021). *Django makes it easier to build better Web apps more quickly and with less code.* Obtenido de https://www.djangoproject.com/

Doran, G. T. (1981). There's a S.M.A.R.T. way to write management's goals and objectives. *Management Review (AMA FORUM), 70*, 35-36.

Efros, A., & Freeman, W. (2001). Image quilting for texture synthesis and transfer. *In Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (págs. 341-346). Association for Computing Machinery.

Emami, H., Aliabadi, M., Dong, M., & Chinnam, R. (2020). SPA-GAN: Spatial Attention GAN for Image-to-Image Translation. *IEEE Transactions on Multimedia, 23*, 391-401.

Fedorova, S. (2021). Generative Adversarial Networks for Urban Block Design. *arXiv preprint arXiv:2105.01727*.

Francel, A., & Uribe Kaffure, C. (2020). *Métodos de investigación formativa en historia de la arquitectura y el urbanismo.* Tolima Colombia: Sello Editorial. Universidad Del Tolima.

Fredrik Lundh. (2009). *tkinter — Python interface to Tcl/Tk*. Obtenido de https://docs.python.org/3/library/tkinter.html

Gharibi, H., & Habib, A. (2018). True Orthophoto Generation from Aerial Frame Images and LiDAR Data: an update. *Remote Sensing, 10*(4), 581.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., . . . Bengio, Y. (2014). Generative Adversarial Networks. *arXiv preprint arXiv:1406.2661*.

Google. (2015). *Keras InceptionResNetV2*. Obtenido de https://keras.io/api/applications/inceptionresnetv2/

Hasso Plattner Institute of Design at Stanford. (2019). *Design Thinking Process Diagram*. Obtenido de https://empathizeit.com/design-thinking-models-stanford-d-school/

Hertz, J., Krogh, A., & Palmer, R. (1991). Introducción a la teoria de la computacion neuronal. Redwood City, California: Addison-Wesley Pub. Co.

Hertzmann, A., Jacobs, C., Oliver, N., Curless, B., & Salesin, D. (2001). Images Analogies. *Hertzmann, A., Jacobs, C. E., Oliver, N., Curless, B., & Salesin, D. H. In Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, (págs. 327-340).

Ioffe, S., & Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *In International conference on machine learning* (págs. 448-456). PMLR.

Isola, P., Zhu, J.-Y., Zhou, T., & Efros, A. (2017). Image-to-image Translation with Conditional Adversarial Networks. *In Proceedings of the IEEE conference on computer vision and pattern recognition*, (págs. 1125-1134).

Izadyyazdanabadi, M., Belykh, E., Mooney, M., Eschbacher, J., Nakaji, P., Yang, Y., & Preul, M. (2018). Prospects for Theranostics in Neurosurgical Imaging: Empowering Confocal Laser Endomicroscopy Diagnostics via Deep Learning. *Frontiers in Oncology 8*, 240.

Karras, T., Laine, S., & Aila, T. (2018). A Style-Based Generator Architecture for Generative Adversarial Networks. *In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, (págs. 4401-4410).

Kingma, D., & Welling, M. (2014). Auto-Encoding Variational Bayes. *arXiv preprint arXiv:1312.6114*.

Lehmann, S. (2010). Green Urbanism: Formulating a Series of Holistic Principles. *Surveys and Perspectives Integrating Environment and Society SAPI EN S, 3*(2).

Li, J., Chen, Z., Zhao, X., & Shao, L. (2020). MapGAN: An Intelligent Generation Model for Network Tile Maps. *Sensors, 20*(11), 3119.

Linnainmaa, S. (1970). *The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors. Master's Thesis (in Finnish).* Helsinki: Univ. Helsinki.

London School of Economics and Political Science. (2014). *Urban age project*. Obtenido de https://urbanage.lsecities.net/#data

Magrinya, F. (2010). El Ensanche de Barcelona y la modernidad de las teorías urbanísticas de Cerdà. *Ingeniería y territorio: revista del Colegio de Caminos, Canales y Puertos*(88), 68-75.

Malave, R., & Aceves, O. (2019). Influencia de la ciudad colonial hispanoamericana en el Ensanche de Barcelona: análisis morfológico del Plan Cerdà para Barcelona de 1859. *XI Seminario Internacional de Investigación en Urbanismo. 11.* Barcelona - Santiago de Chile: Departament d'Urbanisme i Ordenació del Territori. Universitat Politècnica de Catalunya.

Ministerio de Transportes, Movilidad y Agenda Urbana. Gobierno de España. (2015). *Plan Nacional de Ortofotografía Aérea (PNOA)* . Obtenido de https://pnoa.ign.es/presentacion-y-objetivo

Moore, G. (1975). Progress in digital integrated electronics. *Technical Digest of the International Electron Devices Meeting, 21*, 11-13.

Nauata, N., Kai-Hung, C., Chin-Yi, C., Mori, G., & Furukawa, Y. (2020). House-GAN: Relational Generative Adversarial Networks for Graph-constrained House Layout Generation. *In European Conference on Computer Vision* (págs. 162-177). Cham: Springer.

Olazaran, M. (1993). A Sociological History of the Neural Network Controversy. *Advances in Computers, 37*, 335-425.

Oliván, R. (2020). Instituciones que aprenden. HIP: un modelo de innovación pública para la era post-COVID. *Resumen ejecutivo SEGIB para la XXVII Cumbre Iberoamericana de Andorra.* Madrid: SEGIB.

Open Street Map. (2021). *Open Street Map*. Obtenido de https://www.openstreetmap.org/

Pengyang, W., Fu, Y., Dongjie, W., Bo, H., & Chang-Tien, L. (2020). Reimagining City Configuration: Automated Urban Planning via Adversarial Learning. (A. f. Machinery, Ed.) *Proceedings of the 28th International Conference on Advances in Geographic Information Systems*, 497–506. doi:10.1145/3397536.3422268

PyInstaller Development Team. (2021). *PyInstaller. PyInstaller freezes (packages) Python applications into stand-alone executables, under Windows, GNU/Linux, Mac OS X, FreeBSD, Solaris and AIX.* Obtenido de https://www.pyinstaller.org/

RedMonk analyst firm. (2021). *The RedMonk Programming Language Rankings: January 2021*. Obtenido de https://redmonk.com/sogrady/2021/03/01/language-rankings-1-21/

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *Ronneberger, O., Fischer, P., & Brox, T. (2015, OctobeIn International Conference on Medical image computing and computer-assisted intervention* (págs. 234-241). Cham: Springer.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review, 65*(6), 386.

Ruiz, I. (2018). A Random Forests classification method for urban land-use mapping integrating spatial metrics and texture analysis A Random Forests classification method for urban land-use mapping integrating spatial metrics and texture analysis. *International Journal of Remote Sensing, 39*(4), 1175-1198.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., & Chen, X. (2016). Improved Techniques for Training GANs. *arXiv preprint arXiv:1606.03498*.

Salisbury, J. d. (1159). *Metalogicon.* Chartres.

Schwaber, K. (1997). SCRUM Development Process. *In Business object design and implementation*, 117-134.

Song, J., Li, J., Chen, H., & Wu, J. (2021). MapGen-GAN: A Fast Translator for Remote Sensing Image to Map Via Unsupervised Adversarial Learning. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 14*, 2341-2357.

Terán Troyano, F. (2009). Presentación de la edición facsímil del plano de los alrededores de barcelona Y proyecto de su reforma y ensanche, obra de ildefonso cerdá (1859). *Boletín de la Real Academia de Bellas Artes de San Fernando, 108*, 211-212.

Tran, N.-T., Tran, V.-H., Nguyen, N.-B., & Cheung, N.-M. (2019). Self-supervised GAN: Analysis and Improvement with Multi-class Minimax Game. *arXiv preprint arXiv:1911.06997*.

University of Oregon. (2005). *Nolli Maps*. Obtenido de http://nolli.uoregon.edu/

Ushahidi Inc. (2021). *Ushahidi maps*. Obtenido de https://www.ushahidi.com/

Wang, D., Fu, Y., Wang, P., Huang, B., & Lu, C.-T. (2020). Remaining City Configuration: Automated Urban Planning via Adversarial Learning. *Wang, D., Fu, Y., Wang, P., Huang, B., & In Proceedings of the 28th International Conference on Advances in Geographic Information Systems*, (págs. 497-506).

Wang, H., Wang, J., Wang, J., Zhao, M., Zhang, W., Zhang, F., . . . Guo, M. (2018). GraphGAN: Graph Representation Learning with Generative Adversarial Nets. *In Proceedings of the 32nd AAAI Conference on Artificial Intelligence, 32*(1).

Wang, T.-C., Lui, M.-Y., Zhu, J.-Y., Tao, A., Kautz, J., & Catanzaro, B. (2018). High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. *In Proceedings of the IEEE conference on computer vision and pattern recognition*, (págs. 8798-8807).

Wang, T.-C., Lui, M.-Y., Zhu, J.-Y., Tao, A., Kautz, J., & Catanzaro, B. (2018). Hight-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. *NVIDIA Corporation*.

Xueqing, D., Yi, Z., & Shawn, N. (2018). What is it like down there? Generating dense ground-level views and image features from overhead imagery using conditional generative adversarial networks. *Deng, X., Zhu, Y., & Newsam, S. (2018, November). What is it like down there? Generating dense ground-level views and image featuresIn Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, (págs. 43-52).

Yi, Z., Zhang, H., Tan, P., & Gong, M. (2017). DualGAN: Unsupervised Dual Learning for Image-to-Image Translation. *In Proceedings of the IEEE international conference on computer vision*, (págs. 2849-2857).

Yongxin, Z. (2020). A Survey of Image to Image Translation with GANs.

Zhang, L., Wang, M., Liu, M., & Zhang, D. (2020). A Survey on Deep Learning for Neuroimaging-Based Brain Disorder Analysis. *Frontiers in neuroscience, 14*.

Zhu, J.-Y., Taesung, P., Isola, P., & Efros. (2017). Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. *In Proceedings of the IEEE international conference on computer vision*, (págs. 2223-2232).

Zhu, J.-Y., Zhang, R., Pathak, D., Darrel, T., Efros, A., Wang, O., & Shechtman, E. (2017). Toward Multimodal Image-to-Image Translation. *arXiv preprint arXiv:1711.11586*.

Anexos

Anexo I. Esquema de las redes GAN utilizadas

La siguiente figura 37 muestra el esquema del discriminador utilizado.

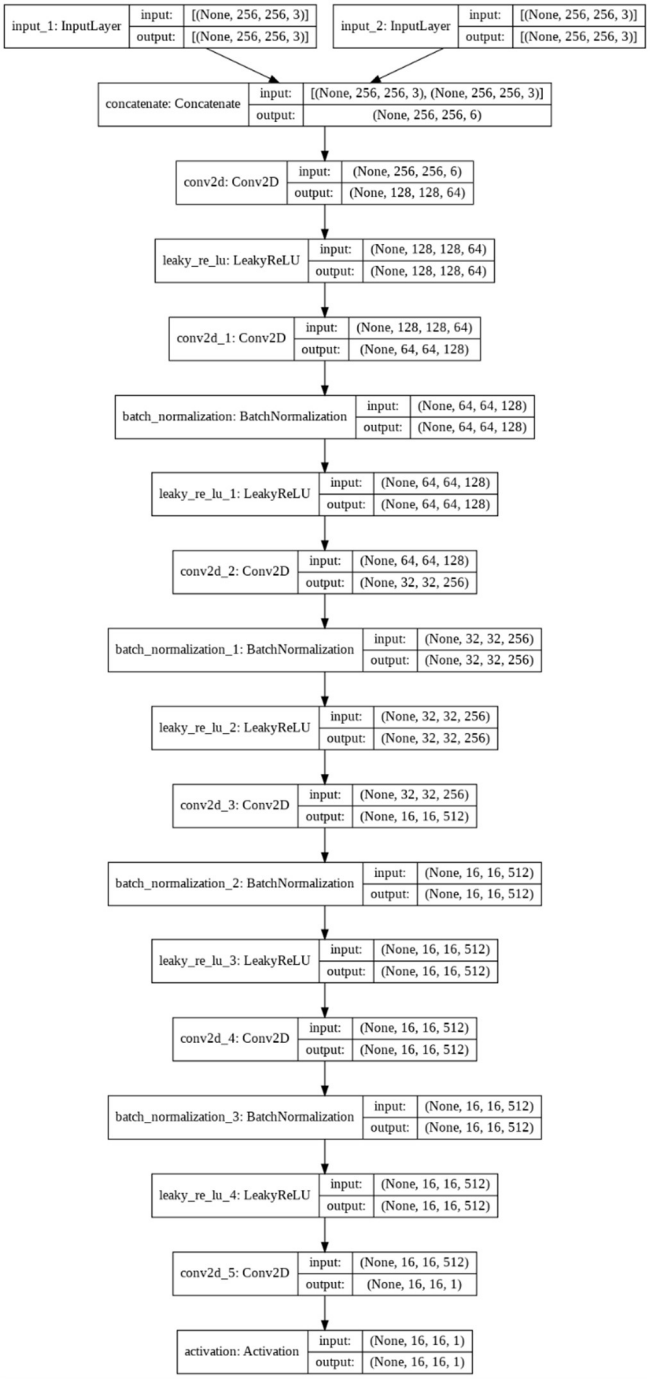

Figura 37. *Esquema de capas del discriminador (*Elaboración propia)

La siguiente figura 38 muestra el esquema del generador.

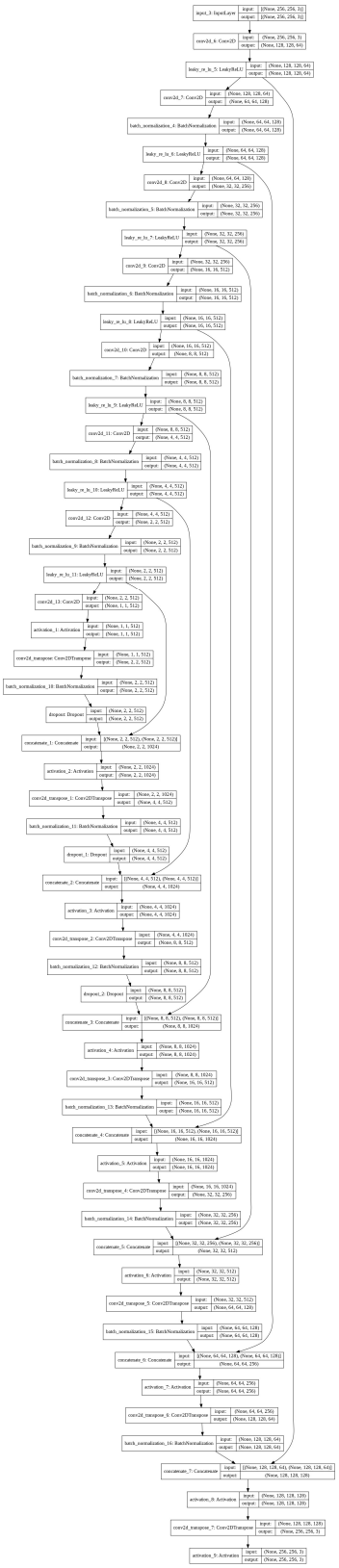

Figura 38. *Esquema del generador* (Elaboración propia)

La siguiente figura 39 muestra el esquema del de la red GAN

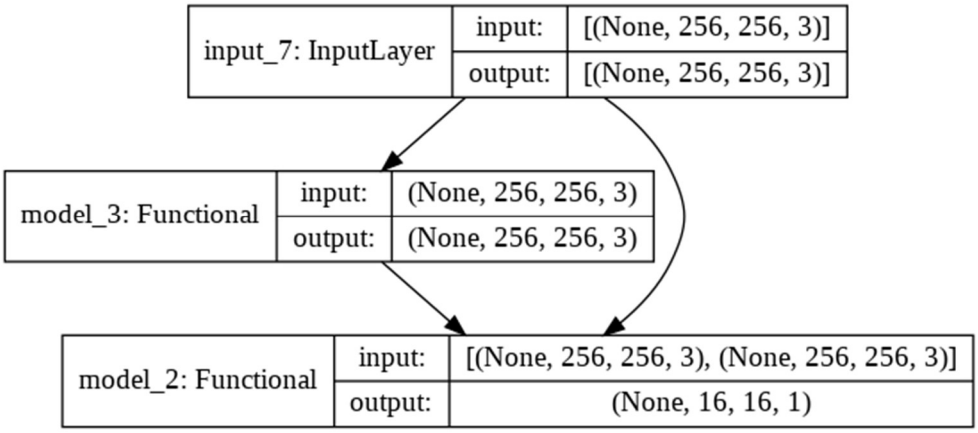

Figura 39. *Esquema de la red GAN* (Elaboración propia)

Anexo II. Cuestionario de evaluación de la herramienta

Evaluación Herramienta eMapGAN

Estimado colaborador,

Has tenido la oportunidad de probar una herramienta de inteligencia artificial cuyo objetivo es explorar las posibilidades de las redes neuronales generativas en los procesos urbanísticos. El objeto de la herramienta es servir de inspiración en procesos tempranos del desarrollo urbanístico, facilitando la visualización de ortofotos de nuevas realidades generadas de forma automática o de forma manual. El módulo automático incluye una funcionalidad para aplicar criterios ambientales que mejoran urbanismos existentes.

El siguiente cuestionario pretende evaluar tanto la utilidad real de la herramienta, como el potencial de desarrollo futuro, tratándose de la aplicación de una tecnología reciente, como las GAN, a una disciplina consolidada, como el urbanismo. Cabe recordar que el sistema ha sido configurado para una porción de mapas de la ciudad de NY, si bien podría configurarse para cualquier otra ciudad del mundo con la colaboración de un experto en IA.

Nombre

Tu respuesta

email de contacto

Tu respuesta

¿Ha participado en las fases previas de concepción y testeo de la herramienta?

- ○ Si, he participado
- ○ No, es la primera vez que utilizo la herramienta
- ○ Otros:

Profesión

Tu respuesta

¿Cuál es su relación con el urbanismo, la inteligencia artificial o las redes generativas adversarias?

Tu respuesta

¿Has probado la herramienta eMapGAN en su versión de escritorio?

- ○ Si, desde mi propio ordenador de escritorio
- ○ Sí, desde el ordenador que se me ha facilitado
- ○ No, lo siento, no he tenido la oportunidad de probar la herramienta

¿Cuánto tiempo has dedicado a probar y reflexionar sobre la herramienta?

- ○ Menos de 5 minutos
- ○ Entre 5 y 10 minutos
- ○ Más de 10 minutos
- ○ Otros:

¿Cuánto tiempo has dedicado a probar y reflexionar sobre la herramienta?

- ○ Menos de 5 minutos
- ○ Entre 5 y 10 minutos
- ○ Más de 10 minutos

¿Resulta la herramienta, tal y como ha podido verificarla, útil para el desarrollo de planeamiento en la actualidad?

	1	2	3	4	5	
Ninguna utilidad	○	○	○	○	○	Mucha utilidad

¿Resultaría la herramienta, con un entrenamiento más completo y generando imágenes fotorealistas, útil para el desarrollo de planeamiento en la actualidad?

	1	2	3	4	5	
Ninguna utilidad	○	○	○	○	○	Mucha utilidad

¿Resultaría la herramienta inspiradora en el proceso de desarrollo de planos?

	1	2	3	4	5	
Ninguna utilidad	○	○	○	○	○	Mucha utilidad

¿Resulta de utilidad el módulo map2emap que aporta condicionantes sostenibles a mapas urbanísticos?

	1	2	3	4	5	
Ninguna utilidad	○	○	○	○	○	Mucha utilidad

¿Resulta de utilidad el módulo draw que permite dibujar tus propios mapas?

	1	2	3	4	5	
Ninguna utilidad	○	○	○	○	○	Mucha utilidad

¿En términos generales, considera que el desarrollo de esta propuesta puede derivar en una herramienta con alto potencial en un futuro a 5 años?

	1	2	3	4	5	
Poco potencial	○	○	○	○	○	Mucho potencial

¿Le gustaría añadir algún comentario o propuesta?

Tu respuesta

Enviar

Anexo III. Pruebas y resultados

Las figuras 40, 41 y 42 muestran algunos de los resultados generados durante el entrenamiento, para diferentes epoch.

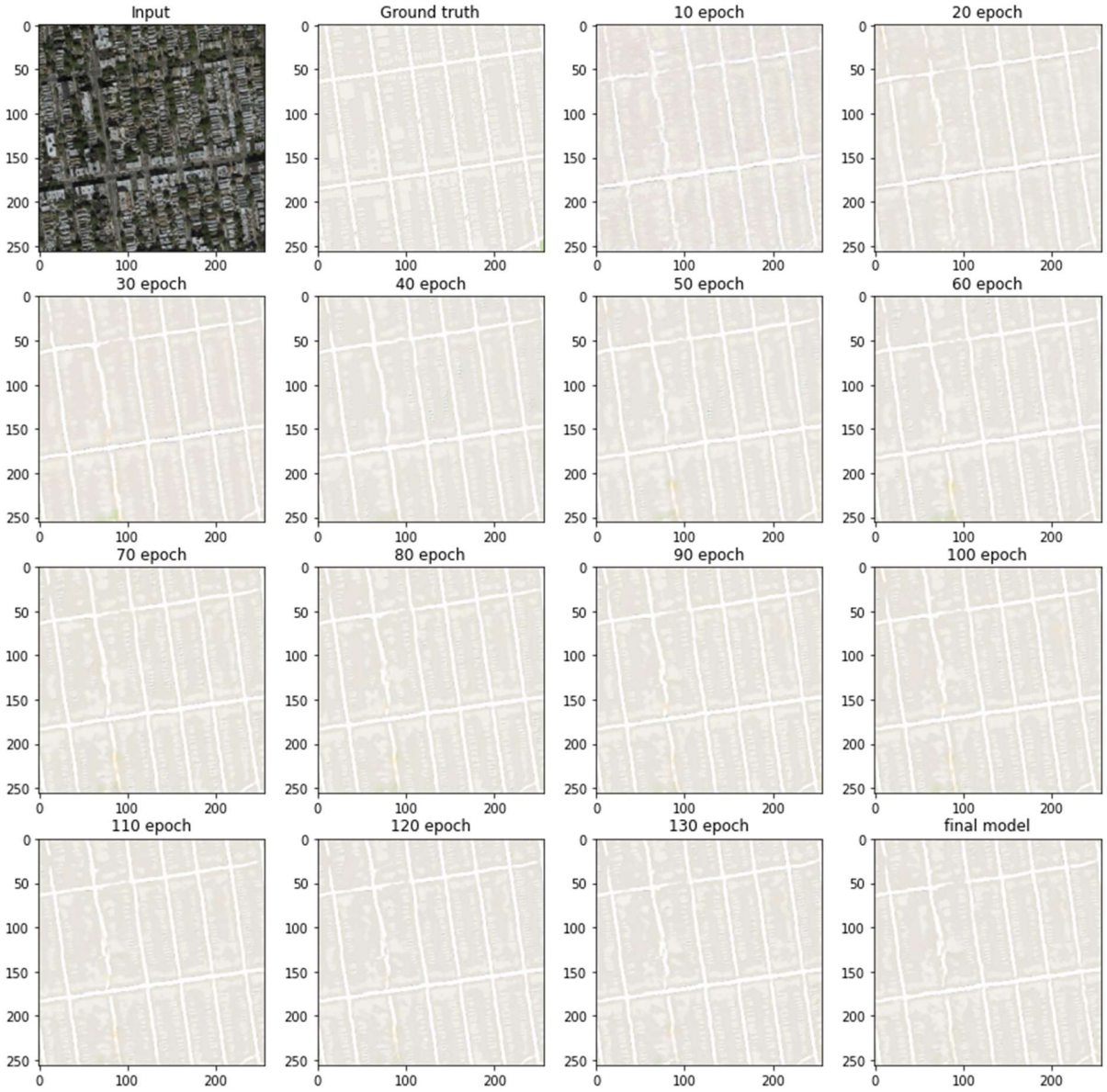

Figura 40. *Mapas generados durante el entrenamiento* (Elaboración propia)

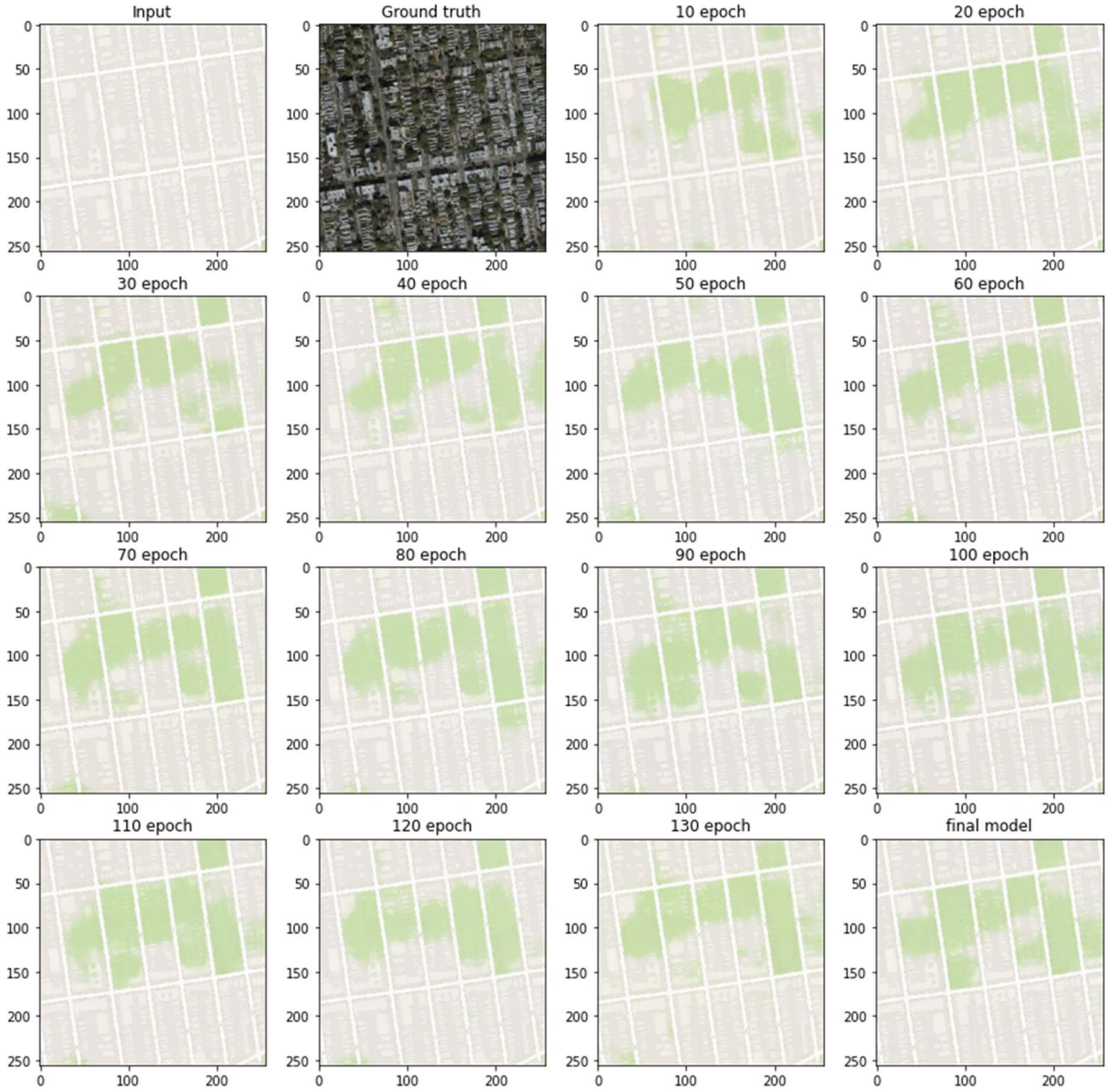

Figura 41. *Mapas sostenibles generados durante el entrenamiento* (Elaboración propia)

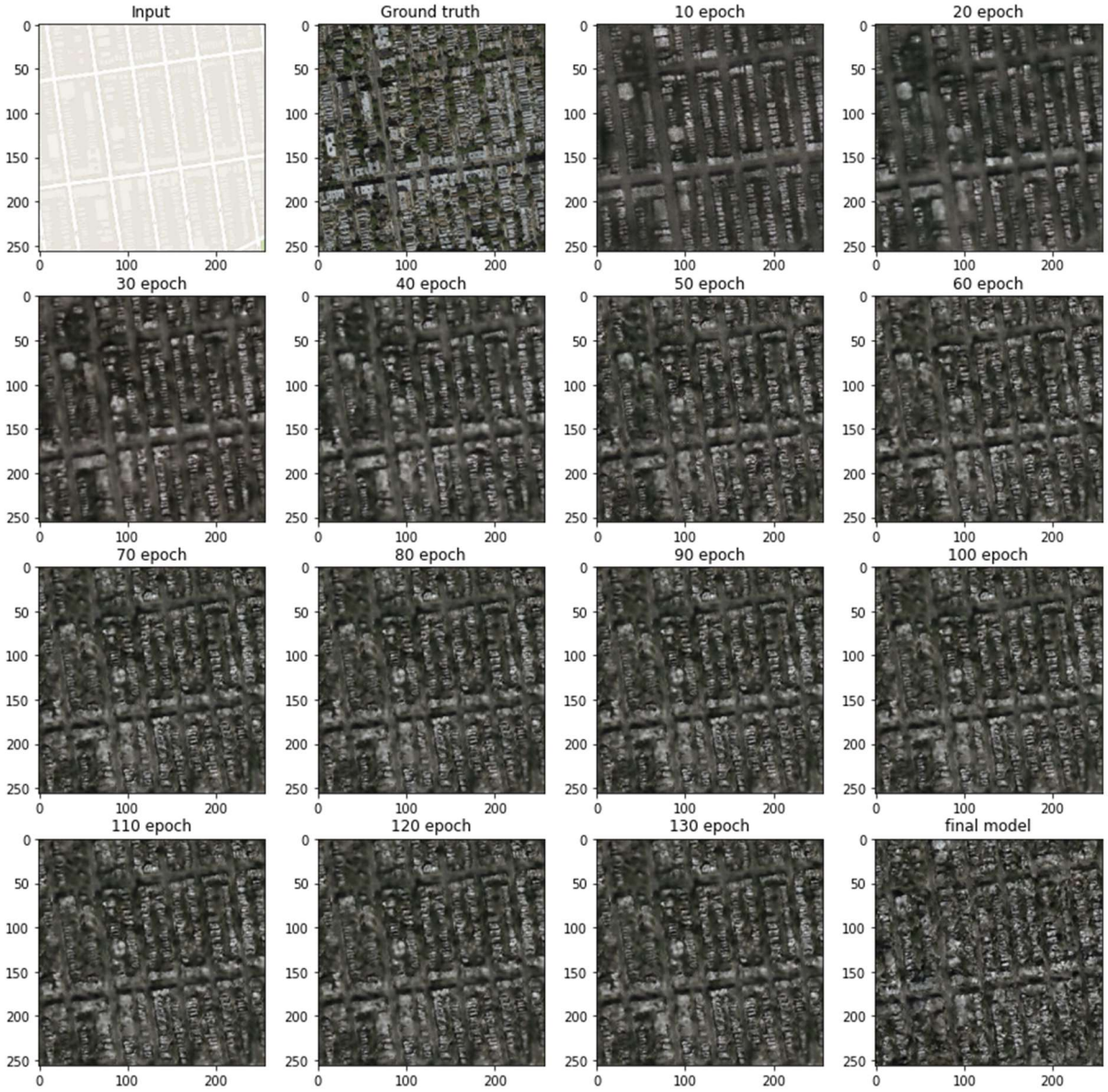

Figura 42. *Imágenes satélites generadas durante el entrenamiento* (Elaboración propia)

Las métricas durante los entrenamientos se muestras en las figuras 43, 44, 45, y 46 verificando la convergencia de la red gan.

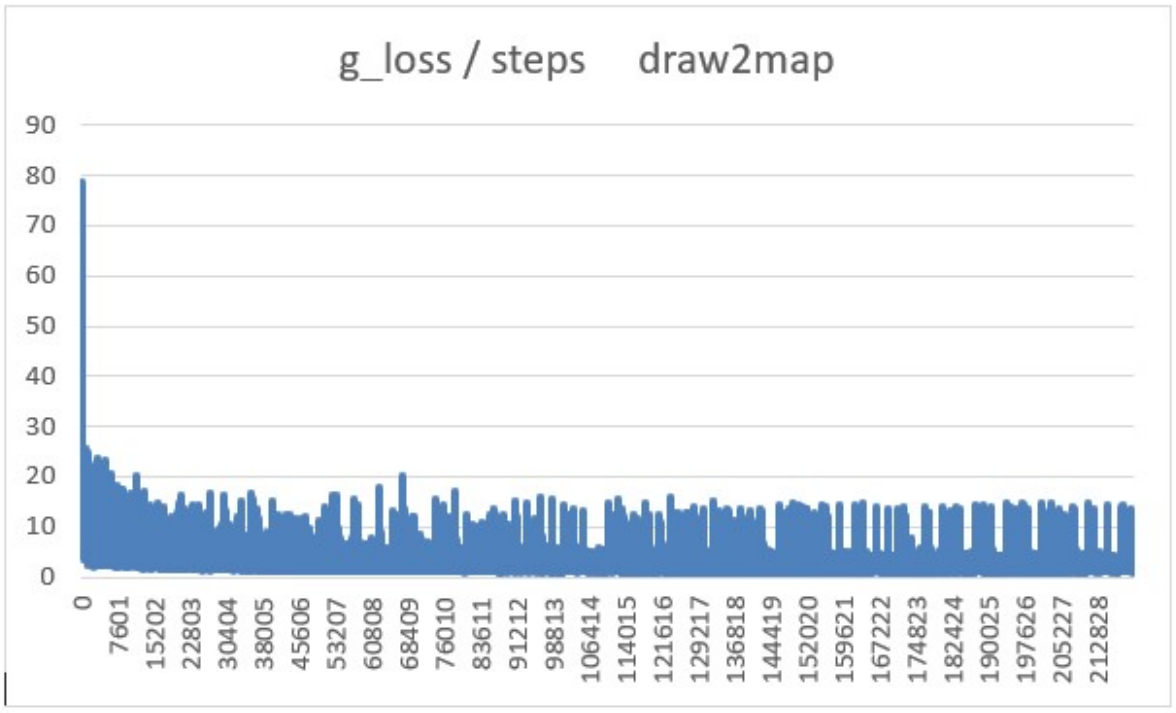

Figura 43. Convergencia g_loss durante entrenamiento draw2map (Elaboración propia)

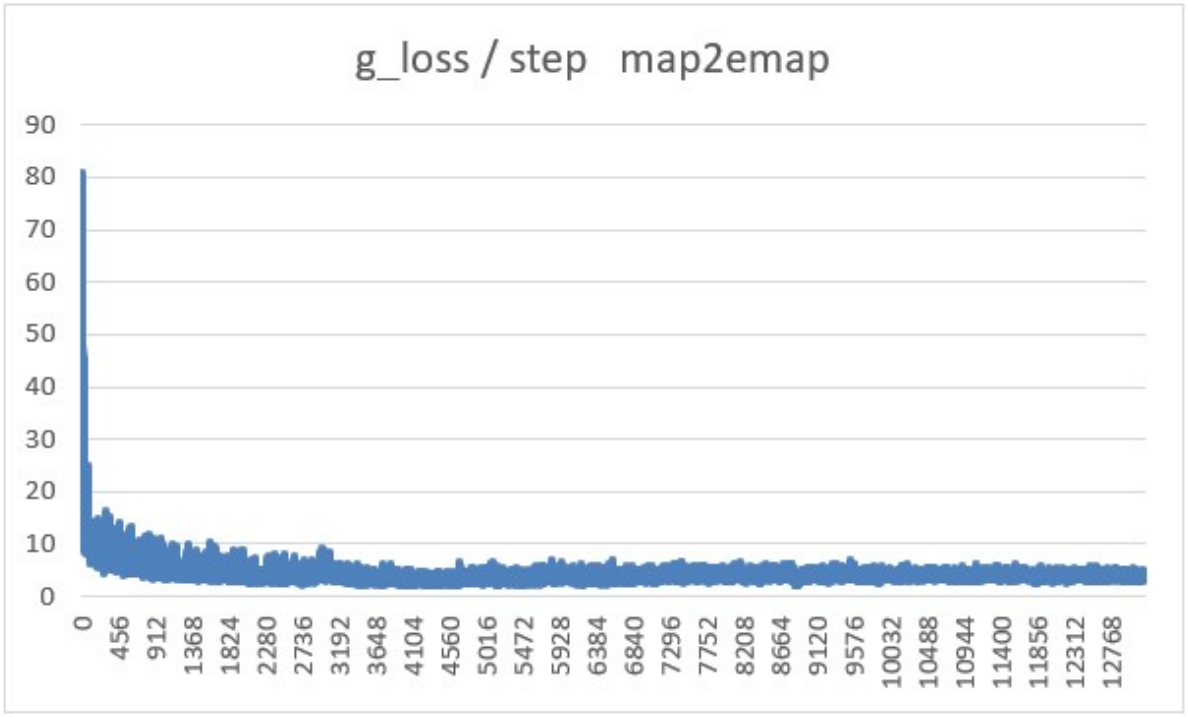

Figura 44. Convergencia g_loss durante entrenamiento map2emap (Elaboración propia)

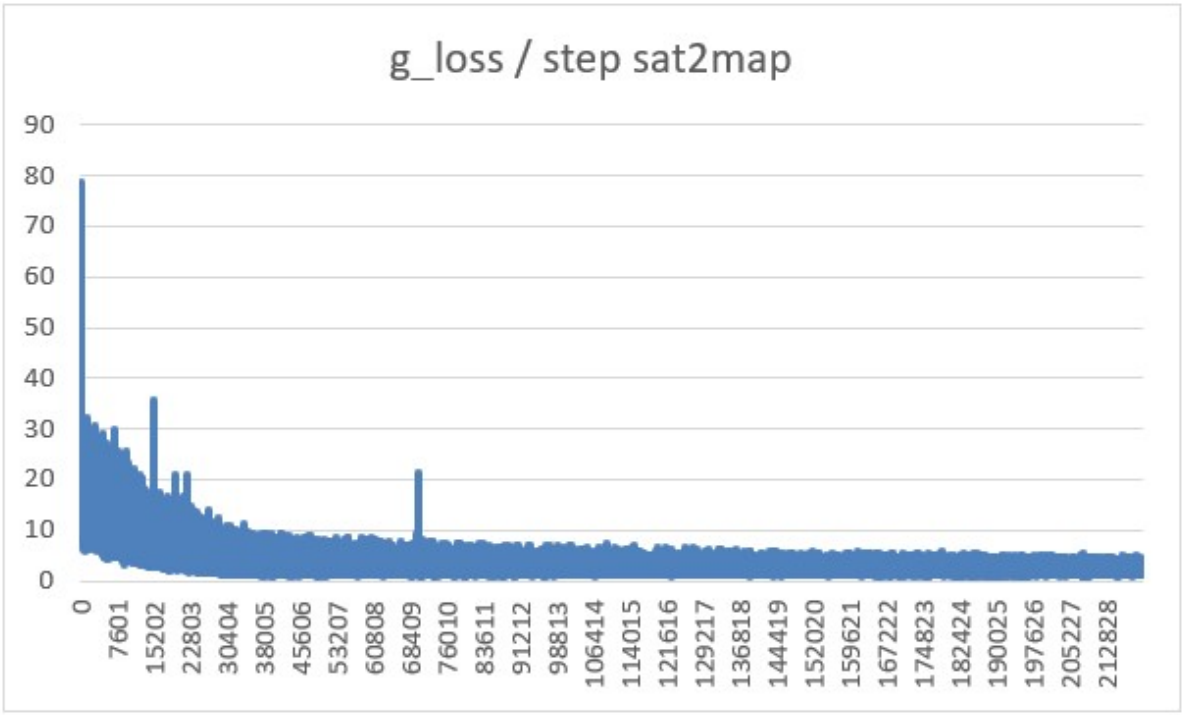

Figura 45. Convergencia g_loss durante entrenamiento sat2map (Elaboración propia)

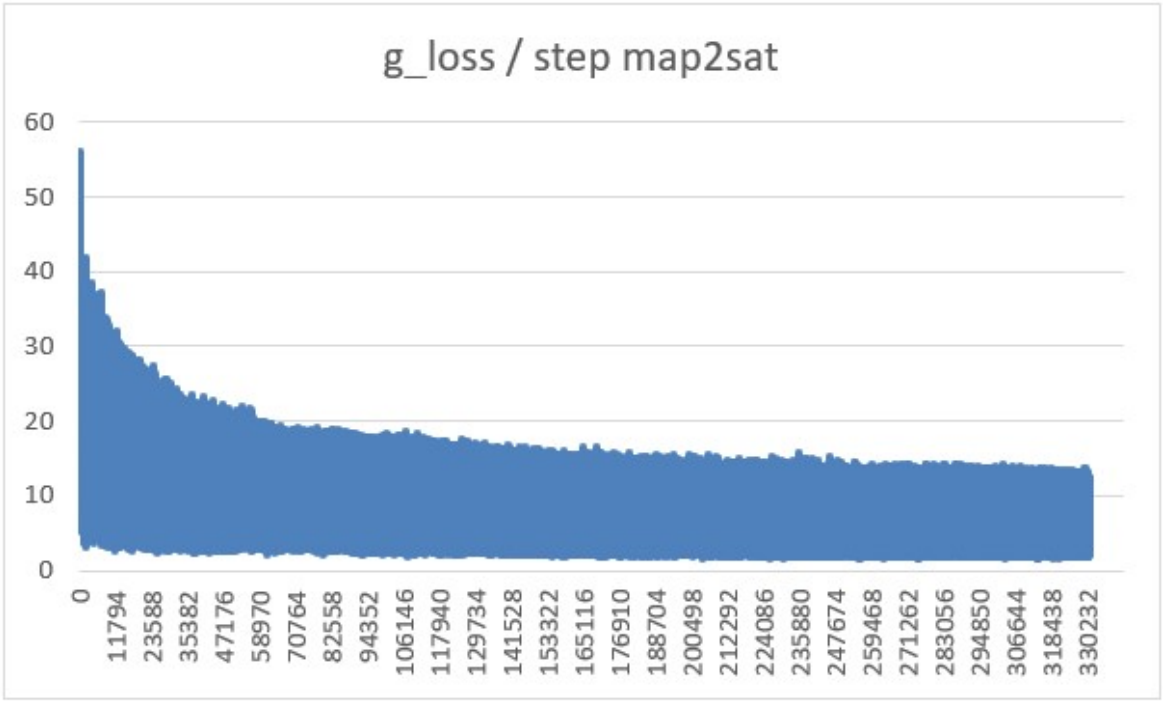

Figura 46. Convergencia g_loss durante entrenamiento map2sat (Elaboración propia)

Anexo IV. Instalación de la herramienta eMapGAN

Para instalar la herramienta en Windows 10 es necesario descargarse de la web un archivo comprimido, descomprimirlo en el ordenador local y ejecutarlo. A continuación, se indican los pasos a seguir.

PASO 1. Acceder al siguiente enlace: https://sites.google.com/view/emapgan

El enlace da acceso a la página web mostrada en la figura 47 que contiene tanto el ejecutable como el código fuente del proyecto.

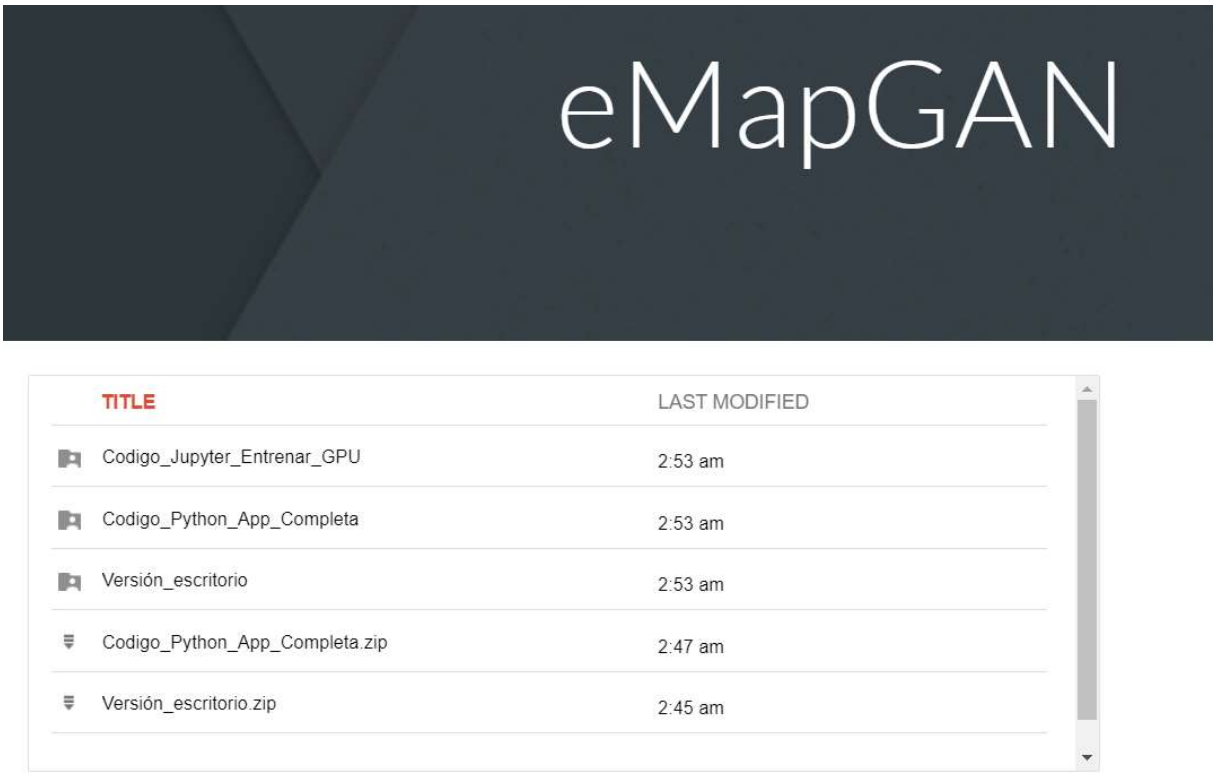

Figura 47. *Website para la descarga de la herramienta* (Elaboración propia)

PASO 2. Descargar Versión_escritorio.zip

PASO 3. Descomprimir.

Una vez que se descomprima el archivo se identifica el esquema de carpetas mostrado en la figura 48.

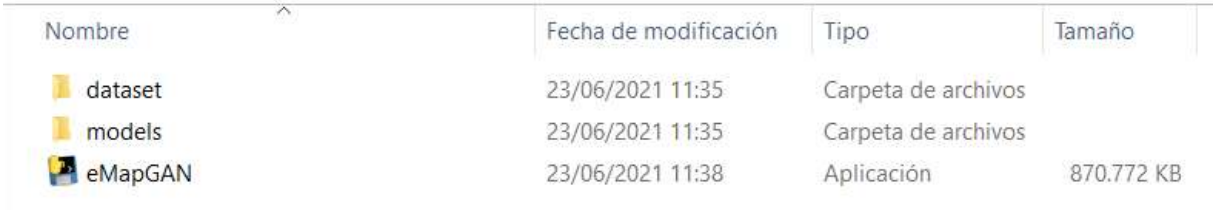

Figura 48. *Esquema de carpetas de la instalación de escritorio* (Elaboración propia)

PASO 4. Ejecutar eMapGAN.exe

Se mostrará la pantalla inicial, grafiada en la figura 49, donde se debe introducir una imagen. Puede ser cualquiera de las carpetas proporcionadas. En caso de querer trabajar con una creada por el usuario deberá tener el mismo formato que el dataset. Se recomienda, por ejemplo, las siguientes imágenes:

dataset/ValidationImages/500.jpg

dataset/TrainImages/400.jpg

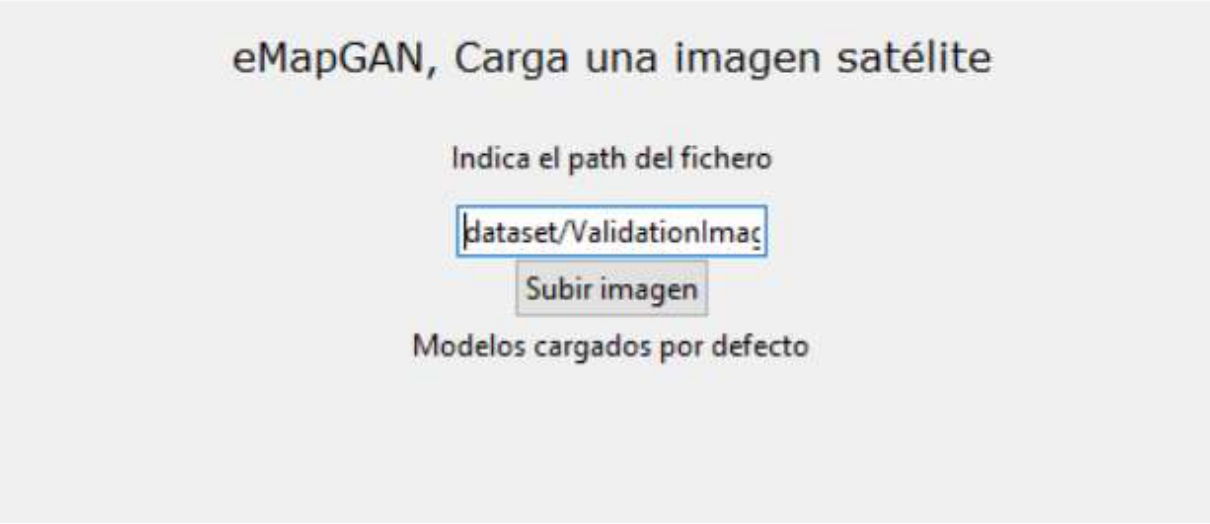

Figura 49. Pantalla inicial de la herramienta (Elaboración propia)

Una vez pulsado el botón “subir imagen” se abrirá una ventana como las mostrada en la figura 50 con las diferentes imágenes generadas.

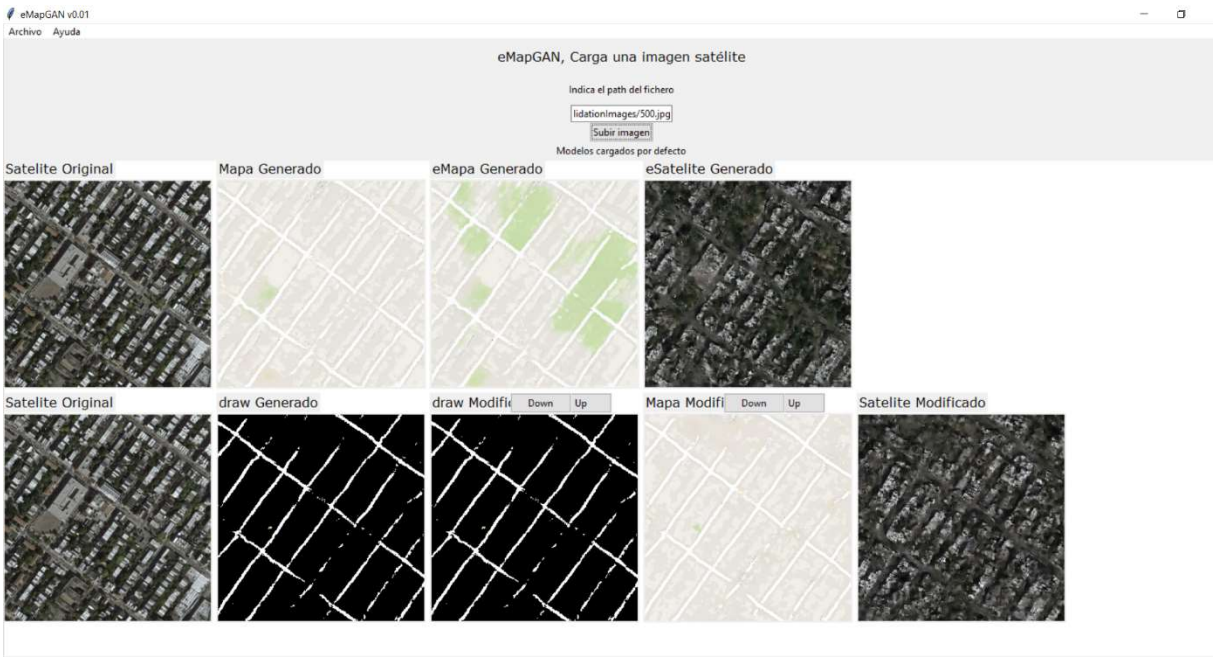

Figura 50. Pantalla de trabajo de la herramienta (Elaboración propia)

NOTA 1. Para editar las imágenes del modelo edraw o el mapa es necesario pulsar el correspondiente botón Down. La imagen aparecerá en el directorio raíz del programa. Es posible editarla con cualquier programa de dibujo, sin cambiarle el nombre. Al pulsar el botón up el programa actualizará la imagen automáticamente.

NOTA 2. Utilizar diferentes modelos. Los modelos se encuentran en la carpeta modelo, pudiendo así actualizarse fácilmente o intercambiarse con diferentes modelos entrenados con diferentes fines.

Anexo V. Artículo de investigación

A continuación, se incluye un artículo de investigación que resume el trabajo realizado y los principales resultados obtenidos.

Planificación urbanística basada en GAN sensible a criterios ambientales

Adán Piñero Alquegui

Universidad Internacional de la Rioja, Logroño (España)

19 de junio de 2021

Resumen

Las Generative Adversarial Network (GAN) son una arquitectura de redes neuronales de reciente desarrollo capaces de generar nuevas realidades a partir de un conjunto de datos de entrenamiento. Con el objeto de explorar el potencial de las redes generativas convolucionales en disciplinas que se expresan gráficamente, como el urbanismo, se desarrolla una herramienta de software denominada eMapGAN. Esta herramienta genera a partir de una imagen satélite del terreno, un mapa convencional, un mapa más sostenible e incluso una ortofoto de una realidad no existente, pero más sostenible que la original. Incluye también la posibilidad de editar mapas intermedios generando así nuevas realidades, todo ello mediante una interfaz gráfica de usuario. Se concibe como una herramienta experimental, percibida como útil por los usuarios, que explora las posibilidades de las redes GAN para inferir conocimientos complejos y constituir una herramienta de apoyo en las fases de concepción del diseño de las ciudades.

Palabras Clave

Cartografía, redes generativas adversarias, redes neuronales convolucionales, sostenibilidad, urbanismo.

I. Introducción

La concentración de la población en ciudades es una tendencia ascendente desde 1950, cuando 732 millones vivían en zonas urbanas hasta las previsiones para el año 2050 cuando 6.500 millones de personas residirán en grandes urbes [1]. Este rápido crecimiento de las zonas urbanas supone un reto para los profesionales del urbanismo, que deben afrontar el diseño de grandes áreas cumpliendo con requisitos de sostenibilidad, movilidad y calidad de vida.

El urbanismo es una disciplina consolidada que funciona como un integrador de información, ambiental, social, estética o arquitectónica a partir de la cual genera diseños para urbanizar los espacios. Estos diseños tienen un alto impacto sobre el entorno y sobre sus habitantes, aún décadas después de su implantación, por lo que el trabajo de diseño es complejo y se tienen en cuenta numerosas fuentes de datos. La principal forma de plasmar la información es gráfica, mediante planos.

El incremento de la capacidad de computación y de las fuentes de datos disponibles, como las imágenes por satélite, han permitido el desarrollo de soluciones basadas en redes neuronales convolucionales. Estas soluciones están en fase de adopción temprana en disciplinas consolidadas, como puede ser el urbanismo.

Durante el presente trabajo se ha desarrollado una aplicación basada en redes neuronales que es capaz de generar nuevas realidades, más sostenibles, a partir de una imagen por satélite de una porción del territorio. De esta forma es posible facilitar al profesional urbanista la concepción de nuevas realidades, visualizadas como ortofotos de lugares que nunca han existido.

La herramienta se basa en diferentes modelos entrenados bajo una arquitectura de redes generativas adversarias, de tipología convolucional. A partir de un dataset de ortofotos y mapas se han entrenado 4 modelos diferentes capaces de traducir entre dos dominios de la imagen:

- Ortofoto satélite a mapa
- Mapa a eMapa (Un mapa más sostenible)
- Mapa a ortofoto satélite
- Espacio de dibujo a mano a mapa.

La herramienta incluye una interfaz gráfica (GUI) que permite un uso de dos flujos de trabajo. El primero genera una nueva realidad más sostenible, en forma de ortofoto, a partir de la imagen por satélite original. El segundo permite la edición sencilla de un espacio de dibujo para que la herramienta genere un mapa y una imagen por satélite de la posible realidad dibujada.

De esta forma el urbanista puede intuir como afectarán sus decisiones a la imagen real del territorio, sirviendo como una herramienta experimental adicional a las habituales en fases iniciales de diseño.

Se ha testeado la herramienta con expertos urbanistas, geógrafos y responsables de infraestructuras, que han comprendido el concepto y el uso de la propuesta de software. Les ha resultado útil, si bien no sustituye a sus herramientas habituales y, en todos los casos, han detectado un elevado potencial de desarrollo en sus ámbitos de trabajo.

II. Estado del Arte

La disciplina que desarrolla la planificación necesaria para rediseñar los espacios urbanos es el urbanismo. Se trata de un sistema integrador de la información del entorno generando diseños que tendrán un alto impacto sobre el desarrollo del entorno [2]. Cabe desatacar la forma en la que el proyecto de Ensanche de Barcelona, aprobado en 1860, define la vida en parte de la ciudad en los último 160 años. Y no sólo en Barcelona, sino que ha tenido un impacto significativo Santiago de Chile y Montevideo [3]. Es por tanto una idea de la posibilidad de

empaquetar un conocimiento complejo y sutil para ser reutilizado en otras ciudades diferentes para aquellas para las que fue originalmente desarrollado.

La integración de la información se ha apoyado en las tecnologías de la información, hasta la denominada City 3.0 [4], donde se capta en tiempo real información de redes sociales, satélites o IoT, con un impacto sobre las decisiones de planificación urbana.

En 2014 se desarrolla el concepto de redes neuronales adversarias [5], capaces de entrenarse y generar nuevas realidades de forma eficaz. La contraposición de una red generativa y una discriminativa, aprendiendo una a través de la otra, constituye una arquitectura capaz de crear nuevos registros no diferenciables del dataset de entrenamiento.

En el caso de la cartografía se proponen mapas imaginarios gracias a la transferencia de estilos [6]. También se experimenta capturando el estilo de los mapas Nolli [7], del siglo XIX, y aplicándolos a nuevos desarrollos urbanos, o trasferencia de Barroco a Modernismo [8], mediante arquitecturas CycleGan [9]. También se han entrenado redes para conseguir imágenes fotorrealistas a nivel de suelo desde imágenes por satélite [10]. Soluciones menos gráficas, pero también efectivas proponen el uso de valores discretos para variables como la densidad de ocupación y el uso de grafos para determinar las interconexiones entre zonas contiguas [11].

Un problema abordado habitualmente ha sido la conversión de una imagen satélite a un mapa, lo que resulta de utilidad en aplicaciones de navegación y en portales cartográficos. Desde el punto de vista de las redes GAN se ha abordado con diferentes estrategias. Entendiendo una imagen por satélite y su correspondiente mapa como asimilables a grafos se ha desarrollado MapGAN [12]. Se han utilizado otras arquitecturas para resolver este problema como BicycleGAN [13], MSGAN [14] o DualGAN [15].

Desde el punto de vista del uso de redes neuronales generativas para realizar desarrollos basados en planos complejos, destacan las aproximaciones para generar arquitecturas completas de edificios [16] basadas en CycleGAN [9] y Pix2Pix [17].

Desde el punto de vista del urbanismo, hay redes neuronales que determinan el mejor posible uso futuro de un suelo, a modo de clasificador, mediante la arquitectura LUCGAN [18].

No obstante, cabe destacar que el entrenamiento de redes GAN, para conseguir imágenes de ortofotos, su inversa, mapas sostenibles desde mapas originales o incluso convertir bocetos en mapas constituyen una tipología de problema único; la traducción entre dos campos diferentes de imágenes. Este problema se ha abordado mediante arquitecturas más genéricas, con menos exactitud que las desarrolladas ad hoc, pero con mayor versatilidad.

El algoritmo Pix2Pix [17] es capaz de entrenarse a partir de 400 imágenes de resolución 256 x 256 emparejadas entre los dos dominios a interconectar. Un desarrollo posterior de esta herramienta en alta resolución será Pix2Pix HD [19]. En cambio, si disponemos de un mayor número de imágenes, pero no emparejadas entre sí podremos entrenar redes CycleGAN [9].

El caso de un algoritmo genérico y versátil entre más de dos dominios es posible aplicar las arquitecturas basadas en StarGan [20].

Por último, cabe destacar que se han identificado herramientas de conversión entre satélite a mapa, de carácter propietario y vinculadas a grandes corporaciones, como Google Maps. No se han detectado herramientas destinadas al profesional urbanista, a escala de su escritorio habitual, como puede ser Autocad. Y en todo caso no se ha detectado herramientas, basadas en GAN, que aporten una solución más sostenible desde una ortofoto original o que permitan generar nuevos mapas y ortofotos desde un boceto del usuario.

Del análisis del estado del arte concluimos que existe interés y potencial en la aplicación de una tecnología emergente, como las redes neuronales convolucionales bajo una arquitectura adversaria, en una disciplina consolidada y de alto contenido gráfico como el urbanismo o el planeamiento de infraestructuras. También se concluye el interés por el concepto de inferir conocimientos sutiles, como la ubicación de zonas verdes, y su aplicación de forma automatizada. Por último, se ha definido la arquitectura pix2pix, versátil y con la posibilidad de entrenarse con un dataset a partir de 400 imágenes, como la arquitectura adecuada para el desarrollo de la herramienta de software.

III. Objetivos y Metodología

El objetivo general es desarrollar una herramienta de software capaz procesar una ortofoto aérea de un área de una ciudad y generar una nueva realidad, más sostenible, mostrándola tanto en mapa como en ortofoto. La sostenibilidad se define como un incremento sustancial de zonas verdes, que un algoritmo GAN pre entrenado determina.

La herramienta se circunscribe al dataset disponible de parte de la ciudad de Nueva York y será testeado por arquitectos urbanistas, entendiendo que se trata de una herramienta experimental como inspiración para los procesos previos de diseño. Adicionalmente es posible retocar etapas intermedias del proceso, como los planos o las representaciones denominadas eDraw, de tal forma que sea posible ajustar los resultados obtenidos.

Específicamente se han fijado 5 objetivos.

O. 1.- Identificar la arquitectura GAN adecuada para realizar la traslación entre dos dominios de imágenes, con dataset limitados a 1000 imágenes en color de baja resolución y apto para los recursos de computación consistentes en menos de 50 horas de GPU en la nube. Incorporará una función de pérdida capaz de converger entre dominios de foto a foto y de foto a contornos.

O. 2.- Definir un pipeline entre varias redes GAN capaz de generar una ortofoto con urbanismo sostenible a partir de una ortofoto de la ciudad de Nueva York. Además, cada una de esas redes GAN entre dos dominios se entrenará adecuadamente con las siguientes tipologías:

- Foto aérea a mapa
- Mapa a imagen de machas según usos (agua, zona verde, vivienda o carretera)
- Imagen de manchas a imagen de manchas más sostenible
- Imagen de manchas a mapa
- Mapa a foto aérea.

La capa imagen de manchas puede ser prescindible. Se testeará su eficacia y se optará por una solución de conversión mapa a mapa si procede.

O. 3.- Identificar un conjunto de datos utilizable para el entrenamiento consistente en imágenes aéreas emparejadas con mapas. Se deberá abordar la forma de pretratar estas imágenes para su uso en la red GAN.

O. 4.- Diseñar e implementar una mínima interfaz de usuario que permita un uso razonable del pipeline. De esta forma será capaz de visualizarse la ortofoto original y la final mejorada, además de estados intermedios, como el mapa. Se trata de una interfaz básica y limitada al área geográfica de entrenamiento del dataset, en ningún caso incorporará la posibilidad de entrenar nuevos datasets para otras áreas geográficas, siendo esto una labor

destinada al ingeniero en inteligencia artificial si procediera realizarse de nuevo.

O. 5.- Testear con expertos en el dominio; un arquitecto urbanista, un ingeniero en planificación y un artista conversando sobre la funcionalidad experimental que aporta la herramienta e identificar posibles líneas de trabajo futuras que pudieran converger en utilidades comerciales.

El software se ha desarrollado mediante una metodología SCRUM. La matriz propuesta por Schwaber [21] concluye la metodología SCRUM como la de mayor posibilidad de éxito para un proyecto con coste y fechas definidas durante el desarrollo, aprendizaje de la tecnología durante el propio desarrollo y creatividad en cada iteración del prototipo.

Se han generado 4 prototipos. Un primero en papel, reflexionando sobre el concepto y las necesidades. Un segundo prototipo sobre imágenes editadas a mano. Un tercer prototipo sobre modelos ya entrenados pero accesibles desde el entorno y programación. Finalizando con un producto mínimo viable, con modelos entrenados y accesible desde un entorno gráfico.

IV. Contribución

La herramienta desarrollada se ha denominado eMapGAN, dado que su principal característica es la de generar mapas y ortofotos más sostenibles, eMap, a partir de una imagen satélite original.

Se ha desarrollado un software con una interfaz gráfica y que puede ser ejecutado en un ordenador con un sistema operativo Windows 10 y está diseñado para utilizar dos flujos de trabajo diferentes.

El primero de ellos se denomina el modo automático, y consiste en cargar una imagen satélite de una zona determinada. Una vez cargada esta imagen en software, de forma automática, cargará los modelos neuronales ya entrenados y mostrará una nueva realidad más sostenible. Para ello generará en pantalla un mapa, a partir de la imagen por satélite original. Un mapa más sostenible que el anterior, a partir del modelo entrenado para determinar la necesidad de zonas verdes y posicionarlas. Y por último generará una nueva imagen, asimilable a una ortofoto, de esa realidad más sostenible. Este flujo de trabajo puede observase en la línea de imágenes de la Figura 1.

Figura1. Map2eMap.h5 sobre una imagen real (Elaboración propia)

Para el diseño de este flujo de trabajo se han entrenado tres redes neuronales, todas ellas con arquitecturas basadas en la tipología pix2pix.

El dataset de entrenamiento ha consistido en 1000 pares de imágenes 600x600 satélite mapa pareadas entre sí y obtenidas del propio repositorio pix2pix. Es posible entrenar los modelos con otras ciudades, mejorando así la resolución o facilitando la inferencia de diferentes estilos. De este dataset se ha hecho una extracción de mapas, generando a mano mapas más sostenibles, hasta un total de 200 registros. A partir de aquí se ha diseñado una función a medida de data aumentation, basada en flip, aumentando el dataset de entrenamiento del modelo map2emap hasta los 660 registros. Se han desechado técnicas de data augmentation como rotación o zoom ya que tienen influencia en las decisiones sobre ubicación de parques que se quieren inferir, como la orientación o la escala y podrían distorsionarlas.

A continuación, se han convertido las imágenes de 600x600 a una resolución de 256x256, se han normalizado a valores entre -1 y 1, y se han utilizado para el entrenamiento de cada una de las tres redes neuronales.

La tipología de redes utilizada han sido las GAN basadas en Pix2Pix. Para ello se utiliza una red neuronal convolucional configurada como discriminador y otra como generador. Puede observarse su composición en la Figura 2.

La red discriminativa tiene como input un par de imágenes, entre el dominio A-B, con valores por píxel entre -1 y 1 para cada pixel de la imagen del dominio A (ej. mapa) y de la imagen del dominio B (ej. eMapa), en sus tres colores RGB. Su salida es un conjunto de valores, que termina ponderándose a un único valor entre 0 y 1. El objetivo es conseguir una red discriminativa que identifique como real, valor 1, un par de imágenes real A-B del dataset. Pero a su vez, este discriminador, será capaz de identificar como falso, valor 0, un conjunto de imágenes donde A pertenece al dataset, pero B ha sido generada de forma artificial por un generador.

La red discriminativa se compone por una concatenación de las dos imágenes de entrada, una convolucional de 64 filtros de 4x4, con stride 2, y cuatro capas de 128,256,512 y 512 filtros con batch normalization tras cada una de ellas. El optimizador utilizado es Adam, con un learning rate de 0.0002 y tratará de minimizar una función de coste en la que mediante una métrica binary_crossentropy se verifica el ajuste del modelo a una predicción real o falso. Esta configuración es la propuesta por la propia arquitectura pix2pix y ha sido la que ha funcionado de forma más eficaz. Se ha implementado en Keras con parte de código desarrollado ad hoc y parte profundamente inspirado en el propio pix2pix y en otros desarrollos en Keras de esta arquitectura [22].

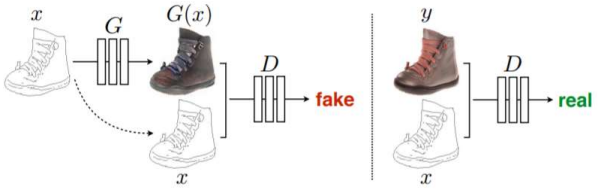

Figura 2. Esquema de entradas y salidas del modelo discriminativo (Zhang, Wang, Liu, & Zhang, 2020)

El generador consiste en una red neuronal basada en Pix2Pix, implementada en Keras, utilizando implementaciones GAN en Python existentes [22] y añadiendo cambios al código para ajustarlo al problema tratado. Se trata de un Encoder-Decoder en tipología U-Net. El encoder tiene como entrada una imagen del dominio A, y aplica diferentes sucesiones de capas convolucionales, batch normalization y dropout hasta una capa de estrangulamiento. A partir de esta capa de estrangulamiento, se recompone la imagen en el dominio B, a partir de un conjunto simétrico de capas Upsampling, batch normalization y dropout. Se concatenan capas simétricas del encoder y del decoder de tal forma que es posible que fluya información a través de toda la red, facilitando la transferencia por ejemplo de colores. Se han probado diferentes

configuraciones siendo la original Pix2Pix la que ha presentado un mejor rendimiento, con una capa de 128 filtros, 256, 512,512,512,512,512 con filtros 4x4. La figura 3 muestra algunas de las capas intermedias.

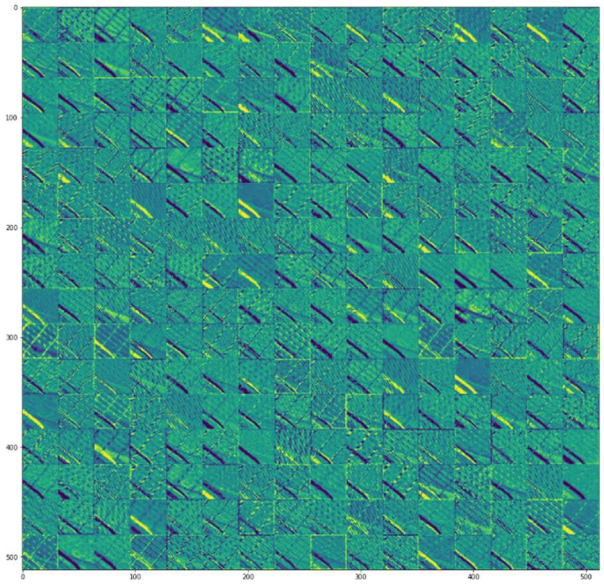

Figura 3. Capas intermedias del encoder de la red entrenada sat2map.h5 (Elaboración propia)

Por último, se configura un modelo gan, basado en Keras, bajo arquitectura Pix2Pix y con código fuertemente basado en otros desarrollos Pix2Pix bajo Keras. El modelo GAN concatena en generador y el discriminador, tal y como se muestra en la figura 2. De esta forma el generador aprende epoch tras epoch a generar imágenes más realistas, capaces de engañar al discriminador. Esta gan intentará minimizar una función de coste referida a una imagen generada por el generador, pero etiquetada 1, es decir, entrenada para engañar al discriminador. De este enfrentamiento surgen tres modelos, de los cuales el generador será capaz de crear nuevas realidades.

El segundo pipeline que permite la aplicación consiste en crear mapas a medida y observar su evolución futura hasta la ortofoto. De esta forma es posible retocar a mano los mapas creados en el modo automático o bien generar nuevos mapas. Para ello se ha definido un espacio denominado eDraw, con manchas de color, donde el negro representa zonas urbanas, el blanco para viales, el azul agua, el verde zonas verdes y el naranja representa vías de alta capacidad.

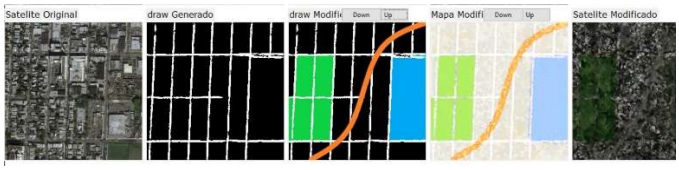

Figura4. Pipeline Satelite -> draw -> draw modificado ad hoc -> mapa -> foto (Elaboración propia)

Para configurar este modelo se ha utilizado en primer lugar una red pix2pix entrenada para convertir la imagen satélite original a mapa. A partir de este mapa se ha diseñado un algoritmo en openCV que genera el espacio denominado draw en manchas de color. Para el paso de draw a edraw el usuario podrá descargar y subir imágenes de tal forma que han podido editarse fácilmente con cualquier herramienta de edición. Desde edraw se utiliza el modelo entrenado anteriormente para converir el boceto a mapa y análogamente desde mapa hasta una nueva imagen satélite.

La combinación de los dos flujos de trabajo, implantados en una interfaz gráfica desarrollada en Tkinter puede observarse en la figura 5.

Figura 5. Pipeline Satelite -> draw -> draw modificado ad hoc -> mapa -> foto (Elaboración propia)

Es posible ejecutar el modelo en Python o a través del ejecutable .exe generado para Windows 10.

El entrenamiento de las diferentes redes no se ha mostrado viable en un ordenador de escritorio con procesador i3 y 4Gb de RAM, dado que si bien la red se entrena durante horas es habitual que ocurran errores, fundamentalmente relacionados con el desbordamiento de RAM. Se ha optado por tanto por un entrenamiento en la nube bajo el servicio Google Colab Pro que da acceso a una GPU Tesla v100 con 16 GB de RAM. Adicionalmente se han podido hacer algunas pruebas con RAM más elevado, utilizando la opción destinada a tal fin en la propia plataforma.

V. Evaluación y Resultados

Cabe destacar dos tipos de evaluaciones diferentes. Una primera consiste en valorar la propia convergencia del entrenamiento de las redes neuronales y sus resultados. Cabe indicar que estos resultados no son objetivos ya que el objetivo es generar imágenes nuevas, no existentes con anterioridad y, por tanto, en cierto modo, subjetivas. En todo caso se plantean las métricas utilizadas para valorar la adecuada convergencia del entrenamiento de los modelos.

En segundo lugar, se expondrá la evaluación de la herramienta desarrollada por usuarios expertos que podrían hacer uso de ella.

Métricas utilizadas y convergencia de los entrenamientos

Para valorar la convergencia de las redes gan se han utilizado las siguientes métricas, observables para un modelo en la Figura 5:

d1_loss: Capacidad del discriminador para detectar imágenes reales. Un valor de 0 determinaría que el discriminador es capaz de identificar siempre una imagen real.

d2_loss: Capacidad del discriminador para detectar imágenes fake. Un valor de 0 determinaría que el discriminador es capaz de identificar siempre una imagen sintética generada por el generador.

G_loss: Capacidad del generador para crear sintéticas próximas al target y capaces de engañar al discriminador para que

las clasifique como reales.

En todas ellas se utiliza la función Binary_Cross_Entropy de acuerdo con las recomendaciones de los autores de la arquitectura Pix2Pix.

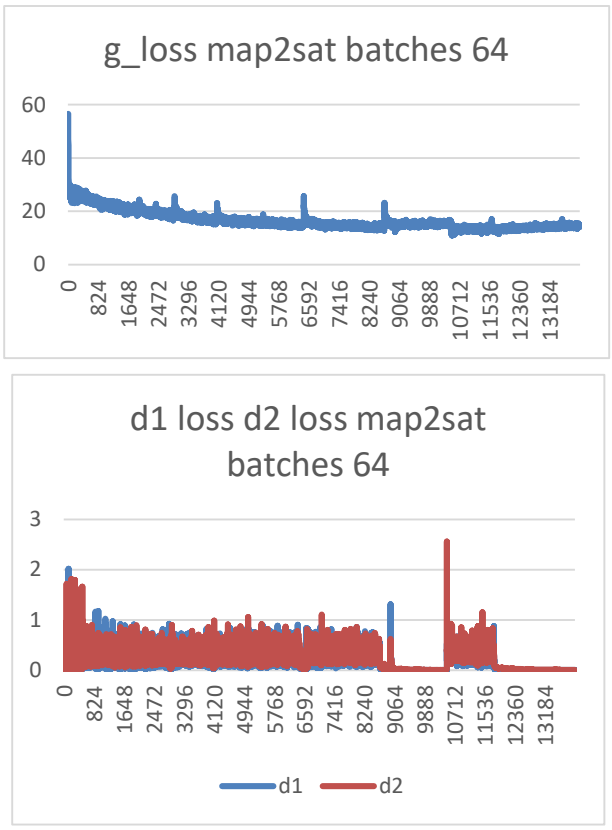

Figura 5. Map2eMap.h5 sobre una imagen real (Elaboración propia)

Cabe destacar que el entrenamiento en una red GAN presenta dos características singulares respecto de la interpretación de los datos:

1.- Los valores de error indicados con anterioridad se calculan para cada iteración en base a un generador y a un discriminador que mejoran, generalmente, de forma incremental. Es por tanto que no es directamente comparable un valor con el anterior, aunque pueden facilitar una idea del correcto entrenamiento del conjunto.

2.- El objeto es la generación de imágenes sintéticas, por lo que no es el objetivo que sean idénticas a imágenes reales, sino más bien que no puedan diferenciarse. Por tanto, al existir cierto caos propio del elemento generativo, es necesaria la opinión humana respecto a los resultados conseguidos. [23]

En general, un buen síntoma de convergencia del entrenamiento de la red GAN consiste en un suave descenso continuado de g_loss lo que significa que gradualmente el generador es capaz de generar imágenes más parecidas al objetivo y que además son capaces de engañar al clasificador.

Por otro lado, d1_loss y d2_loss deben mantenerse equilibradas ya que un valor 0 de alguna de ellas significaría que el discriminador es capaz de diferenciar absolutamente todas las imágenes reales de las fake dejando poco espacio a la mejora del generador. Esto sería debido a un desequilibrio entre la capacidad de discriminación y la de generación que deben entrenarse de forma pareja para conseguir una mejora continuada.

En la Figura 5 se observan los valores de entrenamiento obtenidos para la red map2sat, es decir, la que convierte mapas a imágenes satélites. Los resultados obtenidos con batches de 1 imagen han sido mejores que los obtenidos con batches de 64, tal y como adelantaban los autores del algoritmo pix2pix [17]. No obstante, este mejor comportamiento es observable tras una prueba de los dos modelos con varias imágenes.

Evaluación con usuarios expertos

Se ha realizado una evaluación con usuarios expertos, consistente en una sesión de uso del software y una encuesta posterior. La evaluación se ha realizado con arquitectos urbanistas, ingenieros del ámbito del urbanismo, expertos en geografía, proyectistas de infraestructuras y expertos en el uso y valoración del suelo.

Prácticamente todos ellos han comprendido el uso de la herramienta y han podido utilizarla. Coinciden en que supone una herramienta adicional, que no sustituye a ninguno de sus flujos de trabajo, pero que podría resultar de utilidad, ahora o en el futuro.

Todos ellos identifican un alto potencial al uso de GAN en la disciplina del urbanismo y la cartografía, especialmente frente a retos como el rápido crecimiento de las ciudades y los objetivos verdes marcados por los países para 2030 y 2050.

VI. Discusión

Al utilizarse una arquitectura genérica como Pix2Pix, su propio diseño versátil para diferentes dominios ha hecho que las redes converjan y se entrenen con resultados razonables a partir de las 200 epoch. El uso de redes generadas a medida, o arquitecturas especializadas habría mejorado previsiblemente alguna de las conversiones, y empeorado otras. Por tanto, se considera razonable el uso de una arquitectura versátil, así como los resultados obtenidos.

Como se ha indicado anteriormente, generar imágenes nuevas, no diferenciables de un conjunto de referencia supone que estas imágenes no son comparables con una etiqueta preexistente. Por tanto, la subjetividad humana es clave para discernir los resultados. De las evaluaciones realizadas con usuarios expertos, y de la encuesta que se les ha realizado, se concluye que el resultado es aceptable ya que sirve para intuir el impacto de las infraestructuras sobre mapas y ortofotos, así como para intuir de forma clara la propuesta de mapa sostenible realizado por la herramienta.

No obstante, las imágenes y mapas generados son diferenciables de la realidad por un usuario experto con facilidad por lo que tienen un importante rango de mejora.

De la evaluación de usabilidad de la herramienta con usuarios expertos se concluye que entienden la herramienta y la consideran útil, especialmente en fases de diseño muy iniciales y también para complementar exposiciones gráficas del proyecto. En ningún caso la herramienta sustituye procesos que los usuarios expertos realicen habitualmente, sino que se trata de un software adicional.

Todos los usuarios coinciden en el alto potencial de uso de redes GAN sobre imágenes en el campo de la cartografía y del urbanismo para afrontar los retos del crecimiento constante de las ciudades y los crecientes estándares de calidad de vida y de sostenibilidad requeridos.

VII. Conclusiones

La evolución en los últimos años de la capacidad de cómputo

de los ordenadores, así como del volumen de datos disponibles ha permitido el desarrollo de aplicaciones basadas en redes neuronales. La disponibilidad de unidades GPU y TPU en la nube a bajo coste permite acceder de forma asequible a la experimentación con redes profundas convolucionales. El desarrollo de nuevas arquitecturas como las GAN permite generar nuevas imágenes, a partir de un modelo entrenado con otras imágenes existentes. La aplicación temprana de estas tecnologías a disciplinas consolidadas y cuya forma de expresión es gráfica, como la arquitectura y el urbanismo abre la posibilidad a explorar nuevas herramientas.

En este contexto se ha desarrollado una herramienta que genera nuevas realidades, expresadas en forma de mapa y de ortofoto, más sostenibles que la imagen original. Estas nuevas realidades se generan a partir de un conocimiento inferido desde un conjunto de mapas, en este caso la generación de zonas verdes, siendo capaces de transferir el conocimiento del profesional urbanista am modelo de inteligencia artificial. Tras la evaluación con los diferentes expertos, han comprendido el sentido y el uso de la herramienta y han concluido que podría ser una herramienta adicional en su proceso de toma de decisiones. Se considera por tanto que el objetivo final se ha cumplido, si bien la herramienta necesita evolucionar para resultar de más utilidad para los usuarios expertos.

La herramienta no es suficientemente versátil, ya que no funciona en tiempo real desde un proveedor de mapas online. Además, el entrenamiento se ha basado en un conjunto limitado de imágenes, y, por tanto, el modelo tiende a ajustarse a ese conjunto. Es necesario un experto en inteligencia artificial para entrenar nuevos modelos. A priori, y entendiendo que se trata de una herramienta experimental y para investigadores, esto no supone un inconveniente, pero un modelo más avanzado debería poder entrenarse con facilidad para cualquier ciudad que se seleccione.

Las limitaciones respecto a la memoria disponible, los recursos de computación y el tiempo de entrenamiento, limitado a 12 horas para las GPU en Google Colab, han generado modelos mejorables en resolución.

Las contribuciones del presente estudio a los campos de las redes neuronales convolucionales bajo una arquitectura generativa adversaria condicional y a las disciplinas de la cartografía y del urbanismo se resumen a continuación.

Se ha identificado una arquitectura GAN existente y se ha utilizado para entrenar modelos entre diferentes dominios de la imagen, como ortofoto, mapa, emapa y edraw resultando suficientemente versátil y asequible en términos de computación disponible en la nube. La tipología de GAN Condicional denominado Pix2Pix [17] ha resultado adecuada para dataset entre 100 y 1000 imágenes con entrenamientos inferiores a 12 horas en GPU estándar de Google Colab y RAM hasta 16 Gb.

Se han definido dos flujos de trabajo que han resultado útiles para los usuarios expertos, uno de ellos automático, capaz de generar realidades más sostenibles, expresadas como mapa y como ortofoto simplemente introduciendo a la interfaz gráfica una ortofoto original. El segundo flujo de trabajo permite generar mapas y ortofotos a medida, para lo que se ha definido un sistema de colores útil para el diseñador y comprensible y entrenable para el modelo de inteligencia artificial.

Se ha identificado un conjunto de datos que ha permitido el entrenamiento del modelo, si bien resulta limitado y deberán realizarse entrenamientos adicionales con conjuntos de datos más extensos y heterogéneos.

Se ha implementado una interfaz de usuario mínima que permite el uso de la aplicación a un usuario sin conocimientos de programación. De esta forma se abre una ventana al uso de las redes GAN a profesionales de la cartografía y del urbanismo. Esta interfaz es limitada y meramente exploratoria, por lo que tiene un margen de mejora importante, incluso pudiendo llegar a ser una aplicación web o integrarse en herramientas como AutoCAD.

Se ha testeado la herramienta con usuarios expertos, que han entendido su sentido y la han considerado de posible utilidad. En ningún caso la herramienta sustituye a ninguna de sus dinámicas habituales de trabajo, pero entienden su utilidad y podrían utilizarla como un criterio más, especialmente en las fases iniciales que requieren de inspiración y en las fases de exposición, que requieren de abundante expresión gráfica. Todos los usuarios han considerado que la herramienta y su concepto tienen un alto potencial de desarrollo a medio plazo.

Líneas de trabajo futuro

Las redes generativas adversarias son una tecnología reciente y su aplicación a campos consolidados como la cartografía o el urbanismo es temprana. Es por tanto momento de testear diferentes soluciones y utilidades, inicialmente con modelos genéricos, pudiendo concretarse herramientas específicas en el futuro. De esta forma se entiende el software desarrollado y su evaluación con expertos como una forma de conexión entre esta disciplina de la IA y las disciplinas que se expresan en mapas.

El modelo capaz de generar nuevas realidades más sostenibles supone una forma de demostrar la forma de empaquetar conocimiento en una red neuronal, para su uso posterior. Todos los usuarios expertos han identificado un alto potencial en el concepto y en la herramienta, indicando todos ellos algunas líneas de trabajo futuras que se exponen a continuación.

En el campo de las infraestructuras es habitual la concertación con el territorio. Esto significa que se plantean diferentes alternativas de trazado, por ejemplo, para una autovía o una línea de ferrocarril, y existe una fase de valoración del impacto. Es necesario utilizar todas las herramientas al alcance del proyectista para poder explicar el impacto a todos los niveles, desde el residente en la zona, sin conocimientos técnicos, hasta las administraciones que autorizan. En este sentido resulta de especial utilidad el flujo de trabajo que permite dibujar diferentes mapas y generar su ortofoto. Esto permitiría mostrar de forma más intuitiva el impacto de la infraestructura. Como línea de trabajo se propone evolucionar la herramienta hacia una integración con portales cartográficos de tal forma que sea posible ver a diferentes niveles de zoom el impacto de una infraestructura en el rango desde los pocos metros hasta los 300 km.

Una segunda línea de trabajo futura planteada fue la posibilidad de utilizar la herramienta, con ajustes o modelos entrenados de acuerdo con el resultado que se pretenda, para explorar zonas a bajo coste. Por ejemplo, urbanismos o análisis de forma muy preliminar en zonas muy extensas de África, donde el presupuesto del proyecto no posibilite invertir demasiado tiempo al detalle. Una herramienta como la planteada podría sistematizar el proceso.

Del trabajo conjunto con el experto en valoraciones y usos del suelo, se concluyó un interés en utilizar la herramienta para mostrar diferentes opciones futuras para desarrollos urbanísticos, como urbanizaciones. No tanto para la toma de decisiones, como en la expresión gráfica. En todo caso, requirió más resolución y exactitud en las imágenes generadas. Conseguir esta resolución, mediante entrenamientos más largos, con dataset mayores y con diferentes configuraciones de capas es una línea de trabajo de mejora de la herramienta.

El usuario experto más relacionado con el desarrollo de conceptos experimentales y la investigación trabaja en portales GIS desde su posición de doctor en geografía. Del trabajo conjunto resultó especialmente interesante el módulo map2emap, capaz de inferir conocimientos en este caso sobre zonas verdes, y aplicarlos a nuevos mapas. Este concepto en sí mismo ya resultó de utilidad en su campo de trabajo y además pudimos concluir que una línea de trabajo futura consistiría en desarrollar diferentes modelos similares. Por ejemplo, un modelo capaz de generar mapas de urbanismos anti-incendio o anti-inundaciones. Incluso podrían aplicarse en serie sobre un mapa original.

Los usuarios expertos relacionados directamente con el urbanismo plantearon el uso de la herramienta tal y como está diseñada en conceptos previos del diseño, donde es necesaria la inspiración. Se concluyó la posibilidad de entrenar modelos diferentes con ideas de diferentes arquitectos, relacionadas con las zonas verdes, las orientaciones, la tipología de las manzanas, la ubicación de colegios y centros de salud, y de esta forma poder aplicar uno u otro a conveniencia en esa fase de diseño. También se planteó la opción de utilizar el discriminador, bien entrenado, para validar de forma rápida y previa planeamientos de forma sistemática, por ejemplo, desde las administraciones públicas.

En conclusión, el potencial de la herramienta es elevado, y ha cumplido su objetivo como propuesta de unión entre las redes GAN y las disciplinas basadas en la representación cartográfica, siendo capaz además de inferir conocimientos complejos como la ubicación de zonas verdes y, mediante modelos entrenados para tal fin, proponer nuevas realidades. Este punto de unión entre disciplinas abre la puerta a diferentes líneas de trabajo como las planteadas. Hay por tanto por delante un intenso trabajo conjunto entre profesionales de la inteligencia artificial y profesionales del urbanismo, la cartografía y la sostenibilidad.

REFERENCIAS

[1] United Nations, Department of Economic and Social Affairs, Population Division (2014). World Urbanization Prospects: The 2014 Revision, Highlights (ST/ESA/SER.A/352).

[2] Francel, A., & Uribe Kaffure, C. (2020). Métodos de investigación formativa en historia de la arquitectura y el urbanismo. Tolima Colombia: Sello Editorial. Universidad Del Tolima.

[3] Malave, R., & Aceves, O. (2019). Influencia de la ciudad colonial hispanoamericana en el Ensanche de Barcelona: análisis morfológico del Plan Cerdà para Barcelona de 1859. XI Seminario Internacional de Investigación en Urbanismo. 11. Barcelona - Santiago de Chile: Departament d'Urbanisme i Ordenació del Territori. Universitat Politècnica de Catalunya.

[4] Anttiroiko, A.-V. (2012). Urban Planning 2.0. International Journal of E-Planning Research (IJEPR), 1, 16–30. doi:https://doi.org/10.4018/IJEPR.2012010103

[5] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, Bengio, Y. (2014). Generative Adversarial Networks. *arXiv preprint arXiv:1406.2661*.

[6] del Campo, M., & Manninger, S. (2020). Imaginary Maps - a Posthuman Urban Design Method based on Neural Style Transfer. *Utopia vs. the city P+ARG Biennial Conference.* Michigan, USA: Taubman College for Architecture and Urban Planning.

[7] University of Oregon. (2005). *Nolli Maps*. Obtenido de http://nolli.uoregon.edu/

[8] del Campo, M., Carlson, A., & Manninger, S. (2020). Towards Hallucinating Machines - Designing with Computacional Vision. *International Journal of Arquitectural Computer*, 1478077120963366.

[9] Zhu, J.-Y., Taesung, P., Isola, P., & Efros. (2017). Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. *In Proceedings of the IEEE international conference on computer vision*, (págs. 2223-2232).

[10] Xueqing, D., Yi, Z., & Shawn, N. (2018). What is it like down there? Generating dense ground-level views and image features from overhead imagery using conditional generative adversarial networks. *Deng, X., Zhu, Y., & Newsam, S. (2018, November). What is it like down there? Generating dense ground-level views and image featuresIn Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, (págs. 43-52).

[11] Wang, D., Fu, Y., Wang, P., Huang, B., & Lu, C.-T. (2020). Remaining City Configuration: Automated Urban Planning via Adversarial Learning. *Wang, D., Fu, Y., Wang, P., Huang, B., & In Proceedings of the 28th International Conference on Advances in Geographic Information Systems*, (págs. 497-506).

[12] Li, J., Chen, Z., Zhao, X., & Shao, L. (2020). MapGAN: An Intelligent Generation Model for Network Tile Maps. *Sensors, 20*(11), 3119.

[13] Zhu, J.-Y., Taesung, P., Isola, P., & Efros. (2017). Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. *In Proceedings of the IEEE international conference on computer vision*, (págs. 2223-2232).

[14] Tran, N.-T., Tran, V.-H., Nguyen, N.-B., & Cheung, N.-M. (2019). Self-supervised GAN: Analysis and Improvement with Multi-class Minimax Game. *arXiv preprint arXiv:1911.06997*.

[15] Yi, Z., Zhang, H., Tan, P., & Gong, M. (2017). DualGAN: Unsupervised Dual Learning for Image-to-Image Translation. *In Proceedings of the IEEE international conference on computer vision*, (págs. 2849-2857).

[16] Chaillou, S. (2019). *AI + Architecture Towards a New Approach.* Harvard University.

[17] Isola, P., Zhu, J.-Y., Zhou, T., & Efros, A. (2017). Image-to-image Translation with Conditional Adversarial Networks. *In Proceedings of the IEEE conference on computer vision and pattern recognition*, (págs. 1125-1134).

[18] Pengyang, W., Fu, Y., Dongjie, W., Bo, H., & Chang-Tien, L. (2020). Reimagining City Configuration: Automated Urban Planning via Adversarial Learning. (A. f. Machinery, Ed.) *Proceedings of the 28th International Conference on Advances in Geographic Information Systems*, 497–506. doi:10.1145/3397536.3422268

[19] Wang, T.-C., Lui, M.-Y., Zhu, J.-Y., Tao, A., Kautz, J., & Catanzaro, B. (2018). High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. *In Proceedings of the IEEE conference on computer vision and pattern recognition*, (págs. 8798-8807).

[20] Choi, Y., Choi, M., Kim, M., Ha, J.-W., Kim, S., & Choo, J. (2018). StarGAN: Unified Generative Adversarial Networks Multi-Domain Image-Image translation. *In Proceedings of the IEEE conference on computer vision and pattern recognition* (págs. 8789-8797). Salt Lake City, UT, USA: IEEE.

[21] Schwaber, K. (1997). SCRUM Development Process. *In Business object design and implementation*, 117-134.

[22] Brownlee, J. (2019). Part 6. Image Translation. En J. Brownlee, *Generative Adversarial Networks with Python.* Machine Learning Mastery.

[23] Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., & Chen, X. (2016). Improved Techniques for Training GANs. *arXiv preprint arXiv:1606.03498*.