Universidad Internacional de La Rioja (UNIR)

ESIT

Máster Universitario en Inteligencia Artificial

Técnicas de inteligencia artificial aplicadas a la planificación de construcción de plantas fotovoltaicas

Trabajo Fin de Máster

Presentado por: López Ferreiro, Manuel Ángel

Director: Gil Ruiz, Jesús

Ciudad: Santiago de Compostela
Fecha: 14 de julio de 2022

Agradecimientos

En primer lugar, a Jesús Gil, por ofrecerme este reto y colaborar desde su experiencia previa, marcando la línea del éxito de este trabajo.

A todas las personas con las que he compartido mi experiencia profesional, por los inmensos aprendizajes que me han regalado. Sería demasiado extenso citarlas en esta página y demasiado injusto dejar a algunas de lado.

A mis padres, por el cariño y la educación que me han dado. A ellos se lo debo todo.

A mi esposa, Yolanda, por su ánimo, apoyo y ayuda en este camino y por hacer de la vida algo bonito y sencillo.

A mi hermana, Ana, por estar siempre a mi lado y por su trabajo fascinante e inspirador en el campo de la neurología.

A mi abuelo, Ramón, por su amor incondicional y admiración. Ha sabido contagiarme desde niño su fascinación por las matemáticas. Este trabajo no hubiese existido nunca sin él.

Resumen

La generación fotovoltaica es una de las tecnologías clave para alcanzar los retos de transición energética. Así, en los últimos años se ha observado un incremento en la ejecución de proyectos de grandes plantas fotovoltaicas. La principal dificultad para cumplir con el plazo de construcción es la elaboración de una planificación adecuada. Una herramienta que genere cronogramas de forma automática puede resultar de gran ayuda a la hora de establecer planificaciones iniciales de referencia. A tal fin, este trabajo compara cinco técnicas de inteligencia artificial, sobre un conjunto de datos formado por 25 planificaciones. La evaluación de los resultados obtenidos sobre datos de test demuestra que ANFIS es la técnica que obtiene mejor rendimiento en todas las métricas de error, si bien conlleva un alto coste computacional. El modelo así obtenido consigue generar un cronograma completo de construcción con un error de un 8% sobre la duración total.

Palabras Clave: Inteligencia Artificial, Planificación de Proyectos, Cronograma de Construcción, ANFIS, Neuroevolución.

Abstract

Photovoltaic generation is a key technology to meet the challenges of energy transition. Thus, in recent years there has been an increase in the execution of large photovoltaic plant projects. The main difficulty in meeting construction deadlines is the elaboration of an adequate planning. A tool that automatically generates schedules can be of great help to set up an initial baseline planning. To this end, this work compares five artificial intelligence techniques, on a data set consisting of 25 schedules. The evaluation of the results obtained on test data shows that ANFIS is the technique that obtains the best performance in all error metrics, although it entails a high computational cost. The model thus obtained manages to generate a complete construction schedule with an error of 8% of the total duration.

Keywords: Artificial intelligence, Project Planning, Construction Schedule, ANFIS, Neuroevolution.

Índice de contenidos

Índice de tablas

Índice de figuras

1. Introducción

La **generación fotovoltaica** es una de tecnologías clave en el futuro de la producción de energía renovable. Supone en torno a un 25% de la potencia instalada en energías renovables a nivel mundial (IRENA, 2020) y se trata de la tecnología que está experimentando un crecimiento más rápido en valor absoluto, tal y como se puede observar en la figura 1, que muestra datos históricos de la evolución de potencia renovable instalada en los años 2019 y 2020 y la previsión para los años 2021 y 2022. El total de potencia instalada en el año 2020 a nivel mundial asciende a 760 GW, lo que ha supuesto un incremento del 21% frente a la capacidad instalada en el año 2019 (UNEF, 2021), muestra del potente crecimiento que está experimentando esta tecnología.

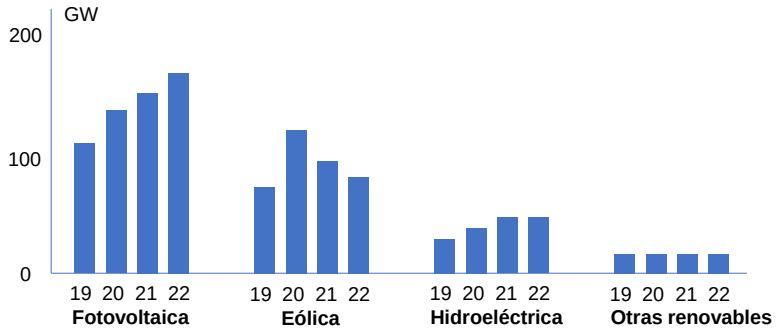


Figura 1. *Incremento anual de potencia instalada, por tecnología renovable.*

Es evidente el papel de liderazgo que debe jugar la energía fotovoltaica en la descarbonización del sistema energético, en orden a dar cumplimiento al objetivo de neutralidad climática expuesto en el Pacto Verde Europeo (Comisión Europea, 2019). Asimismo, se trata de un pilar fundamental para garantizar la seguridad del suministro eléctrico y promover la reducción de los costes de la generación de electricidad.

Un factor clave para alcanzar estos objetivos en el plazo fijado es una adecuada **planificación de los proyectos de construcción** de estas plantas. El presente trabajo pretende ser de utilidad para esta finalidad. Su objetivo es sistematizar la elaboración de cronogramas de construcción de forma precisa, minimizar los sesgos y facilitar la toma de decisiones. Para ello se emplearán técnicas de inteligencia artificial basadas en aprendizaje automático. Esta línea de investigación da continuidad a la iniciada por Gil et al. (2021a), si bien este trabajo incorpora las siguientes novedades:

- Se centrará únicamente en la fase de **construcción.**
- Realizará una **comparativa entre diversos modelos** de diferentes características y complejidad, para así evaluar cuál es el que ofrece resultados más satisfactorios y proceder a su implementación.

- Utilizará un **conjunto de datos más amplio**, con el objeto de incrementar su capacidad predictiva.

Para ello, este trabajo compara cinco técnicas de inteligencia artificial, sobre un conjunto de datos formado por 25 planificaciones. El resultado esperado de dicha comparativa es poder determinar una técnica que permita obtener resultados satisfactorios a la hora de elaborar cronogramas de construcción de forma automática.

El procedimiento seguido para ello partirá de una revisión del estado del arte, con la que se determinarán los modelos que formarán parte de la comparativa. Estos modelos se elaborarán inicialmente desde un punto de vista conceptual. Tras el análisis del conjunto de datos y su preparación, se procederá a la implementación concreta de los modelos en software, a través del lenguaje Python. Estos modelos se entrenarán con un conjunto de datos ("conjunto de entrenamiento") diferente del que servirá para obtener las métricas de rendimiento ("conjunto de test"). Los resultados obtenidos serán objeto de validación y análisis, a fin de determinar un posible modelo óptimo, que se utilizará para su despliegue en la generación automática de cronogramas.

1.1. Motivación

El motivo práctico que subyace tras este trabajo atañe, en esencia, a la **mejora en la planificación de la construcción de infraestructuras**. En concreto, se tratará de **grandes plantas de generación fotovoltaica,** entendiendo como tales aquellas cuya potencia instalada supera 1 MW.

Tal y como mencionan Bowen et al. (2012), el éxito de un proyecto se ha definido de forma clásica en términos de consecución de objetivos de plazo, coste y calidad. Esta definición ha puesto de relieve la existencia de **sesgos** en cuanto a la fijación de objetivos de plazo y coste, incidiendo en fallos sistemáticos en la entrega en los plazos establecidos. Es preciso, pues, realizar un esfuerzo específico en la gestión del plazo de los proyectos para cumplir con los objetivos de entrega acordados.

La causa raíz más importante en el retraso en proyectos de construcción es la **elaboración de una planificación inadecuada** (Agyekum-Mensah y Knight, 2017). La percepción que se tiene sobre el realismo de los clientes de los proyectos de construcción, en cuanto a los plazos, es, según Bowen et al. (2012), de un 57%. Esta escasez de realismo en cuanto a los plazos constituye la denominada por Kahneman y Tversky (1982) "falacia de planificación".

La motivación de este trabajo es, pues, la mejora de las planificaciones en la construcción de plantas fotovoltaicas, a fin de mitigar el principal factor de riesgo en el éxito del cumplimiento

de plazo de estos proyectos y, con ello, colaborar a la **consecución en plazo de los objetivos de descarbonización, seguridad de suministro y reducción de costes del sistema eléctrico**.

Existen, asimismo, dos motivos adicionales, relacionados con el contexto académico y personal, para la elección de este tema: el primero de ellos es que la Universidad Internacional de La Rioja cuenta con una línea de investigación relacionada con la aplicación de inteligencia artificial a la planificación y gestión de proyectos, dentro de la que se enmarca este trabajo. Un último motivo concierne a mi experiencia laboral previa, que ha estado relacionada durante años con el sector energético y, en buena parte, con la planificación de la construcción de infraestructuras.

1.2. Planteamiento del trabajo

Una de las medidas preventivas más eficaces para corregir la falacia de planificación es el empleo de **algoritmos basados en datos históricos representativos** (Kahneman y Tversky, 1982). En el caso que aplica al presente trabajo, estos datos históricos corresponden a planificaciones de construcción de plantas fotovoltaicas.

La propuesta que se lleva a cabo en este trabajo consiste en emplear distintas **técnicas de inteligencia artificial** para modelar la planificación de la construcción de una planta fotovoltaica, entendiendo como tal el conjunto de fechas de inicio y fin previstas para cada una de las distintas fases de su construcción.

La selección de los algoritmos se realizará considerando aquellos:

1. Más aptos para la evaluación de tareas de planificación.
2. Que supongan una novedad con respecto al estado del arte.
3. Que entre sí tengan un carácter complementario, a fin de poder dar una cobertura amplia de soluciones al problema. Un criterio importante a tener en cuenta será la explicabilidad de los algoritmos.

Los resultados que devuelvan las distintas técnicas permitirán establecer cuál de ellas resulta más adecuada para la tarea encomendada y, por tanto, la que se implante para la elaboración de las planificaciones.

1.3. Estructura de la memoria

La memoria se estructura en un cuerpo principal y cuatro anexos: implementación del modelo, tratamiento de datos, pruebas y artículo de investigación. El cuerpo principal continúa con:

- **Sección 2: Contexto y estado del arte**. Esta sección describirá el estado del arte en cuanto a la aplicación de inteligencia artificial a la dirección y gestión de proyectos para, a continuación, tratar sobre la inteligencia artificial en la planificación de proyectos. Se expondrán los trabajos relacionados con la aplicación de inteligencia artificial a la generación de cronogramas, diferenciando dos tipos de técnicas: básicas e híbridas. Por último, se resumirá el estado del arte y se establecerá su aplicación al problema objeto de este trabajo.
- **Sección 3: Objetivos y metodología del trabajo**. En este punto se expondrá el objetivo general del trabajo, que se subdividirá, a su vez, en un conjunto de objetivos específicos. Se expondrá, además, la metodología de trabajo que se propone para alcanzar dichos objetivos.
- **Sección 4: Planteamiento de la comparativa**. Esta sección describirá y justificará la elección de las distintas soluciones alternativas que se proponen para la obtención de las planificaciones. En esta misma sección también se establecerán los criterios y métricas que se considerarán para realizar la comparativa.
- **Sección 5: Desarrollo de la comparativa**. En este punto se llevará a cabo la comparativa detallada entre los modelos propuestos. Se presentarán los resultados y mediciones correspondientes a cada uno de los modelos.
- **Sección 6: Discusión y análisis de resultados**. Tras la presentación de los resultados, en este punto se procederá a su discusión, profundizando en su significado. Se analizarán las ventajas y desventajas de los diferentes modelos.
- **Sección 7: Conclusiones y trabajo futuro**. En este apartado se resumirá el trabajo realizado y las conclusiones obtenidas. Se pondrá de relieve las contribuciones realizadas por el trabajo y se discutirán estas aportaciones, en relación con los objetivos inicialmente previstos. Asimismo, se sugerirán diversas líneas de trabajo como consecuencia del trabajo realizado.
- **Sección 8: Bibliografía**. En esta última sección se expone la bibliografía empleada para la elaboración del trabajo, con la referencia en detalle de todas las citas mencionadas en el texto. El formato que se empleará es el correspondiente a la American Psychological Association (APA), en su versión séptima.

2. Contexto y estado del arte

Este apartado se inicia con el contexto del problema, esto es, el entorno que rodea a la problemática específica que trata de resolver. A continuación del contexto, se expondrán los trabajos relacionados, que servirán como base para la investigación realizada.

2.1. Contexto del problema

A fin de dar el adecuado marco al estudio, en primer lugar, se describirá la importancia y la problemática general que atañe a la gestión de proyectos. Seguidamente, se describirá brevemente una planta fotovoltaica, con enfoque en especial en aquellas de gran dimensión, así como cuáles son los retos que supone su construcción.

2.1.1. Gestión de proyectos

Una definición comúnmente aceptada de un proyecto es "*un esfuerzo temporal que se lleva a cabo para crear un producto, servicio o resultado único*" (Project Management Institute Inc., 2017). El reto de la dirección y gestión de proyectos consiste en integrar y aplicar conocimientos, habilidades, herramientas y técnicas para cumplir con sus requerimientos. Las deficiencias en la dirección de proyectos, o su ausencia, pueden significar consecuencias negativas, como incumplimiento de plazos, sobrecoste, baja calidad y la necesidad de reprocesos. El actual entorno obliga a gestionar proyectos con cronogramas cada vez más cortos, presupuestos y recursos más ajustados y cambios continuos a nivel tecnológico.

Las métricas tradicionales del éxito de un proyecto son el plazo, el coste y la calidad. Otros posibles indicadores del éxito de un proyecto son su rendimiento, su impacto en el cliente, su impacto en el negocio y su preparación para las posibles situaciones futuras (Magaña y Fernández, 2015).

Según Blomquist et al. (2010), en torno a un 30% de los proyectos no alcanzan los objetivos de plazo y presupuesto. Los **factores críticos** para el éxito de un proyecto pueden agruparse en factores relativos 1) al proyecto, 2) al equipo y la dirección, 3) a la organización y 4) al entorno. De todos estos, la **causa fundamental** de fracaso de un proyecto son los errores de estimación en la fase de **planificación**, en un 32% de los casos. Otras causas concretas de fallo relevantes son la ausencia de objetivos claros y la falta de alineamiento en el equipo.

La gestión de proyectos tradicional se basa en cuerpos de conocimiento como PMBOK®, Prince2 e ISO21500, cuyos procesos se apoyan en el conocimiento experto y herramientas

de análisis complementarias. Esta gestión tradicional presenta problemas al aumentar la escala de los proyectos, ya que los requerimientos de precisión son mayores, lo que dificulta la aplicación del juicio experto. Otra de las consideraciones a tener en cuenta sobre los juicios humanos es, tal y como se ha indicado, su susceptibilidad a contener sesgos.

2.1.2. Descripción de una planta fotovoltaica

Las plantas fotovoltaicas convierten la energía solar en energía eléctrica, en base al efecto fotoeléctrico. La energía generada por este tipo de plantas puede ser consumida *in situ* o inyectada a la red eléctrica. Los sistemas fotovoltaicos constan de cuatro componentes principales: los paneles fotovoltaicos, el inversor, la estructura de soporte de los paneles y la instalación eléctrica. A continuación, se realiza una breve descripción de cada uno de estos componentes (Soto et al., 2018):

- **Módulos fotovoltaicos:** los módulos o paneles son el principal componente del sistema fotovoltaico y son los que transforman la energía solar en energía eléctrica, en concreto, en corriente continua. Existen varios tipos de tecnología, si bien los paneles fotovoltaicos cristalinos suponen la tecnología más madura, con una vida útil que puede sobrepasar los 20 años.
- **Inversor**: este equipo electrónico transforma la corriente continua generada por los módulos fotovoltaicos a corriente alterna, para su uso *in situ* o su inyección a la red eléctrica. Además, optimiza la tensión existente en los paneles y permite la vigilancia continua de los parámetros eléctricos, realizando funciones de protección ante sobretensión, sobreintensidad, cortocircuito, variaciones de potencia y frecuencia de la red. La calidad, eficiencia y vida útil de los inversores ha avanzado rápidamente. Actualmente, estos equipos son capaces de transformar la energía hasta con un 98% de eficiencia máxima y su vida útil se puede extender más allá de los 10 años.
- **Estructura de soporte**: fija los módulos fotovoltaicos a la superficie en la que se instalarán. En general, se trata de cubiertas de edificios o, para plantas de gran potencia (>1MW), directamente sobre el suelo. Su diseño depende del tipo de superficie donde se instalará el sistema. Estas estructuras pueden incluir seguidores, que adaptan la orientación de los paneles de acuerdo a la incidencia de la radiación solar.
- **Instalación eléctrica**: se trata de las líneas eléctricas necesarias para para llevar la energía generada hasta el punto de entrega (ya sea consumo propio, o la red eléctrica), así como los transformadores necesarios para adaptar la tensión a la existente en el punto de entrega. La salida de los inversores se produce en corriente

alterna en baja tensión. Si el punto de entrega se encuentra en un nivel de media tensión, serán necesarios centros de transformación elevadores de baja a media tensión. Si el punto de entrega se encuentra en un nivel de alta tensión, será necesario asimismo una subestación elevadora de media a alta tensión. En el caso de instalaciones conectadas a la red, es preciso construir la línea eléctrica que enlace la planta con el punto de conexión, así como la infraestructura eléctrica que realice las funciones de conmutación y control (centro de seccionamiento o subestación, según el nivel de tensión). Las instalaciones eléctricas también incluyen los cuadros eléctricos necesarios para control y medida de la energía, así como los servicios auxiliares para disponer de un nivel mínimo de energía en caso de fallo.

La figura 2 muestra, de modo general, la disposición de los componentes eléctricos de una planta. El cuadro identificado como “AC panel” representa la infraestructura de conexión con la red eléctrica (centros de transformación y, de ser necesarias, subestaciones).

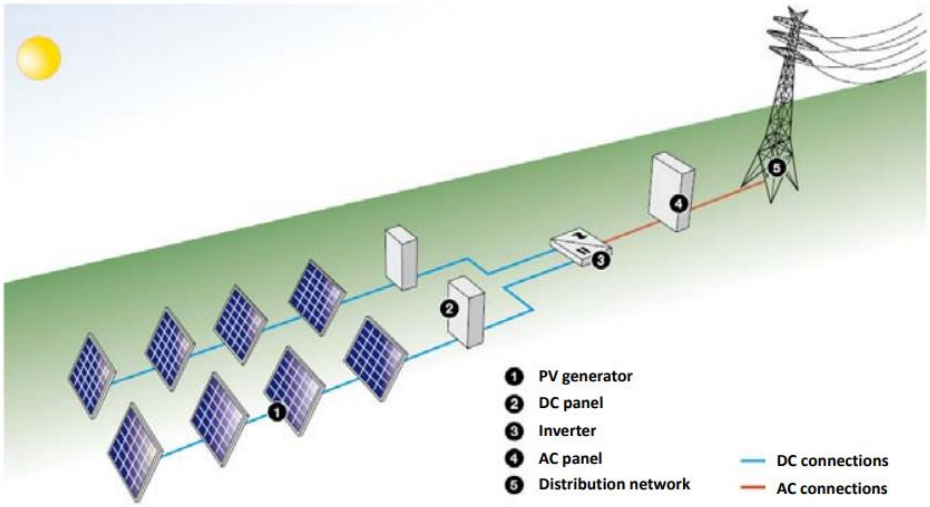


Figura 2. *Esquema de los componentes de una planta fotovoltaica* (Rondon, 2015).

El proyecto de construcción de una gran planta fotovoltaica no supone, en general, un desafío a nivel de complejidad tecnológica, pero en cambio, sí se trata de un importante reto a nivel de organización y gestión. Es precisamente esa sencillez tecnológica la que da lugar a márgenes de beneficio muy ajustados, lo que conlleva la necesidad de que los plazos para su construcción sean muy reducidos. Esto implica la dificultad de coordinar un gran número de recursos de forma simultánea.

En la figura 3 se muestra una gran planta fotovoltaica, que se encuentra en la fase final de su construcción.

Figura 3. *Planta fotovoltaica en la fase final de su construcción* (UNEF, 2021).

Las fases generales de la construcción de una gran planta fotovoltaica que se conecta a la red eléctrica son:

- **Obra civil**, en la que se adecúa el terreno para servir como base para la planta, se realizan los movimientos de tierras, excavaciones, pilares, zanjas y hormigonado. También se ejecutan en esta fase los viales interiores, el sistema de drenaje y el cierre perimetral.
- **Obra mecánica**, en la que se realizan las perforaciones y el montaje de las estructuras, así como la instalación de los módulos.
- **Obra eléctrica**, en la que se instalan los inversores, los centros de transformación y el cableado de media tensión, el cableado de baja tensión y corriente continua, así como otras instalaciones auxiliares (circuito cerrado de videovigilancia -CCTV-, sistema de supervisión y control -SCADA-, estación meteorológica).
- **Línea de transmisión**, en la que se ejecutan las instalaciones para la conexión con la red eléctrica. Comprende la ejecución de la línea que discurre desde la planta fotovoltaica hasta el punto de conexión con la red, la subestación elevadora en la propia planta y la subestación próxima al punto de conexión con la red.
- **Puesta en marcha**. Comprende todas las tareas necesarias para asegurar el inicio del funcionamiento de la planta, como revisiones, ensayos y pruebas.

La tabla 1 muestra el detalle de las tareas correspondientes a cada fase.

Tabla 1. Fases de la construcción de una gran planta fotovoltaica.

Obra civil	Obra mecánica	Obra eléctrica	Línea de transmisión	Puesta en marcha
Limpieza del terreno	Marcado	Instalación de inversores	Subestación	Pre-puesta en marcha
Instalación de infraestructuras	Perforación	Obra eléctrica BT y CC	Línea de transmisión	Obra civil y mecánica
Movimiento de tierras	Montaje de estructuras	Obra eléctrica MT		Obra eléctrica
Accesos y vías interiores	Instalación de módulos	Sala de medida		Fin trabajos
Sistema de drenaje		Instalación CCTV		Puesta en marcha en caliente
Cierre perimetral		Instalación SCADA		Revisión de trabajos
Pilares de estructuras fijas		Estación meteorológica		Fecha de operación comercial
Pilares de inversores				Test de pruebas
Zanjas				Aceptación provisional
Sala de medida				

2.2. Estado del arte

Existe abundante literatura sobre la aplicación de técnicas de inteligencia artificial a la dirección y gestión de proyectos, la planificación de proyectos y la elaboración de cronogramas de construcción. A continuación, se discutirá el estado del arte sobre estos temas, considerando aquellas revisiones y artículos de investigación más relevantes, ya sea por su trascendencia, al haber iniciado una nueva línea de investigación, por su número de referencias, o por resultar los más recientes dentro de su categoría.

2.2.1. La inteligencia artificial en la dirección y gestión de proyectos

Según se ha expuesto, la gestión de grandes proyectos complejos presenta oportunidades de mejora que pueden ser aprovechadas por los avances en el área de la inteligencia artificial. Este campo comprende distintas disciplinas, entre las que destacan el aprendizaje automático, la creación de sistemas cognitivos artificiales, el procesamiento de lenguaje natural, la percepción computacional -con la visión por computador como su máximo exponente-, la planificación y el razonamiento automático e incluso la creación de inteligencias artificiales “fuertes”, que simulen el comportamiento humano. Prieto (2019) espera que la inteligencia artificial sea el mayor foco de inversión en tecnología en el actual decenio. Según este autor, la inteligencia artificial dispone de múltiples casos de uso potenciales dentro de la gestión de proyectos, entre los que, como referencia no exhaustiva, pueden encontrarse áreas transversales como el márketing -p.ej., en la personalización de productos-, ventas -p.ej., en la optimización de precios y predicción de ingresos-, analítica de datos y logística, así como en áreas específicas como son el diseño, la planificación, la gestión de las operaciones y el mantenimiento de las infraestructuras.

Existen numerosas técnicas de inteligencia artificial que han sido aplicadas a la gestión de proyectos. Chua et al. (1997) utilizaron una red neuronal con ocho factores clave para predecir el éxito de un proyecto. También han sido empleados modelos sencillos, como los bayesianos, para predecir factores de éxito en proyectos del campo de tecnologías de la información (Gingnell et al., 2014). Georgy et al. (2005) emplearon un sistema neurodifuso inteligente para predecir el desempeño en un proyecto de construcción, comparando su resultado con el obtenido a través de técnicas estadísticas. Cheng, Lien, et al. (2012) desarrollaron un sistema para predecir el éxito de un proyecto en base a lógica borrosa, algoritmos genéticos, redes neuronales y agrupamiento por *K-means*, en base al cual permitir a los directores de proyecto tomar decisiones de diseño o correctivas para que el proyecto se complete de forma exitosa. Algoritmos bioinspirados, como los basados en colonias de hormigas (Dorigo et al., 2019) han sido aplicados con éxito a diversos problemas, como enrutamiento en redes de

telecomunicaciones, transporte y distribución. Shou (2006) ha demostrado a través de resultados experimentales que este mismo algoritmo es capaz de mejorar el valor actual neto de un proyecto. Otros algoritmos bioinspirados, como evolución diferencial y optimización de enjambre de partículas han servido para resolver muy diversos problemas como, por ejemplo, el de almacenaje de mercancías o el enrutamiento de flotas de vehículos (Kachitvichyanukul, 2012). Los sistemas neurodifusos, como ANFIS, también cuentan con muy diversas aplicaciones, con ejemplos muy variados como la estimación del coste de proyectos, la predicción de la resistencia de materiales de construcción, el estado de mantenimiento de canalizaciones o la predicción de consumo eléctrico (Pellerin et al., 2020; Seresht et al., 2018). Los sistemas neurodifusos hibridados con algoritmos genéticos, como EFNIM o EFHNN, también han sido utilizado en una gran variedad de problemas, como la predicción del éxito de un proyecto (Ko y Cheng, 2007), la determinación del coste de un proyecto (Cheng, Tsai, y Hsieh, 2009; Cheng, Tsai, y Sudjono, 2009) o la evaluación de contratistas (Cheng et al., 2011; Ko et al., 2007). En otra aplicación (Seresht et al., 2018), estos modelos fueron utilizados para predecir el desempeño de los gerentes de los proyectos en función de sus competencias, con el fin de apoyar la toma de decisiones en la selección de gerentes. En estos casos, los criterios de selección cuentan con incertidumbre y subjetividad y por ello están mejor representados por la lógica borrosa. Otro ejemplo de las aplicaciones de este tipo de modelos dentro de la gestión de proyectos ha sido la toma de decisiones en ingeniería geotécnica (Cheng et al., 2008). Otras técnicas de inteligencia artificial como los mapas cognitivos borrosos, las máquinas de vector de soporte o *K-means* han sido también usados para la identificación de factores críticos para el éxito de un proyecto o para la predicción de su éxito (Magaña y Fernández, 2015).

2.2.2. La inteligencia artificial en la planificación de proyectos

El enfoque más tradicional de las tareas de planificación dentro del campo de la inteligencia artificial es llevado a cabo a través de herramientas de *planning* basadas en algoritmos de búsqueda, como STRIPS, PDDL, o, más específicamente en el área de la construcción, a través de redes jerárquicas de tareas (HTN), tal y como muestran Kartam et al. (1991). Según Prieto (2019), este enfoque suele limitar el número de escenarios para conseguir acotar la complejidad del problema de búsqueda. Por ello propone nuevos enfoques dentro de la inteligencia artificial que permitan observar correlaciones más profundas y restricciones ocultas, con los que obtener soluciones novedosas. Buena parte de estos nuevos enfoques se deben a los avances en el área del aprendizaje automático. Así, Cheng et al. (2019) han aplicado modelos de redes neuronales *long short-term memory* para el cálculo del tiempo estimado para la finalización de un proyecto. Portoleau et al., (2020) han aplicado árboles de

decisión para ayudar en la toma de decisiones en problemas de planificación con alto grado de incerteza. Ko y Cheng (2003) han tratado sobre la aplicación de algoritmos genéticos a la planificación de múltiples proyectos que se van a ejecutar simultáneamente, con buenos resultados. También se han empleado técnicas neurodifusas para la planificación de la ubicación de las instalaciones en trabajos de construcción (Song et al., 2017) o la planificación de redes de suministro de agua (Vamvakeridou-Lyroudia et al., 2005). Dentro de las técnicas neurodifusas, ANFIS ha sido utilizada en una gran variedad de problemas relacionados con planificación de proyectos: planificación estratégica de un negocio, predicción de precios de materiales, cálculo de la demanda eléctrica nacional, predicción de perfiles de velocidad de viento para planificación de parques eólicos, planificación de pasos a nivel (Tiruneh et al., 2020) o como herramienta de ayuda para el análisis de proyectos *Green Lean Six Sigma* (Ershadi et al., 2021). Los sistemas bioinspirados también han sido utilizados para la planificación de proyectos. Algunos ejemplos, no exhaustivos, basados en modelos de partículas híbridos, son la elaboración de planificaciones considerando el compromiso entre tiempo y coste (Ashuri y Tavakolan, 2021) o la planificación de la ubicación de grúas de construcción (Lien y Cheng, 2014).

2.2.3. La inteligencia artificial en la elaboración de cronogramas de construcción

Dentro del universo de problemas de planificación, la elaboración automática de cronogramas de construcción ha sido objeto de investigación desde la década de 1960. Una primera clasificación de las distintas aproximaciones a la resolución de este problema es la siguiente (Faghihi et al., 2015):

1. Uso de **modelos generales** que contienen información sobre las características de los proyectos.
2. **Modelos basados en casos** o *case-base reasoning* (CBR). Consiste en la elaboración de nuevos cronogramas a partir de ejemplos históricos con características similares.

Si bien este trabajo se centrará en los modelos basados en casos, es conveniente considerar los trabajos previos sobre el problema conocido como RCPSP (*Resource-Constrained Project Scheduling Problem*), o problema de planificación de proyectos con restricción de recursos, dada la amplia investigación realizada desde la década de 1960 (Agarwal et al., 2011) hasta hoy y la riqueza algorítmica que presentan sus soluciones. Por ello, a continuación, se presentará de forma somera este problema. El RCPSP consiste en encontrar el cronograma de actividades que minimice la duración del proyecto. Un RCPSP es un problema NP-

complejo y, como tal, no determinista. Los métodos para su resolución, aunque se han beneficiado de los avances en el aprendizaje automático, son fundamentalmente heurísticos. Tal y como se muestra en la figura 4, un RCPSP puede definirse formalmente como un grafo acíclico $G = (V, A)$, en el que:

- $V = \{0, 1, \ldots, n, n+1\}$ es el conjunto de vértices. Cada vértice $i = 1, \ldots, n$ representa una actividad, con una duración $d_i > 0, \forall\, i$. El primer y último vértice representan el inicio y final del proyecto, con una duración ficticia igual a cero.
- A es conjunto de arcos. Cada arco $a_{ij} \in A$ representa una relación de precedencia entre dos actividades, por la que cada actividad j no puede iniciarse sin que su predecesora i haya sido completada.
- Existen un conjunto de recursos R, cada uno con capacidad $R_k > 0, \forall\, k = 1, \ldots, K$. Cada recurso se corresponde con un conjunto de trabajadores con una misma especialización o un conjunto de máquinas idénticas.
- Cada actividad $i \in V$ lleva asociada un consumo $r_{ik} < R_k$ de cada uno de estos recursos.

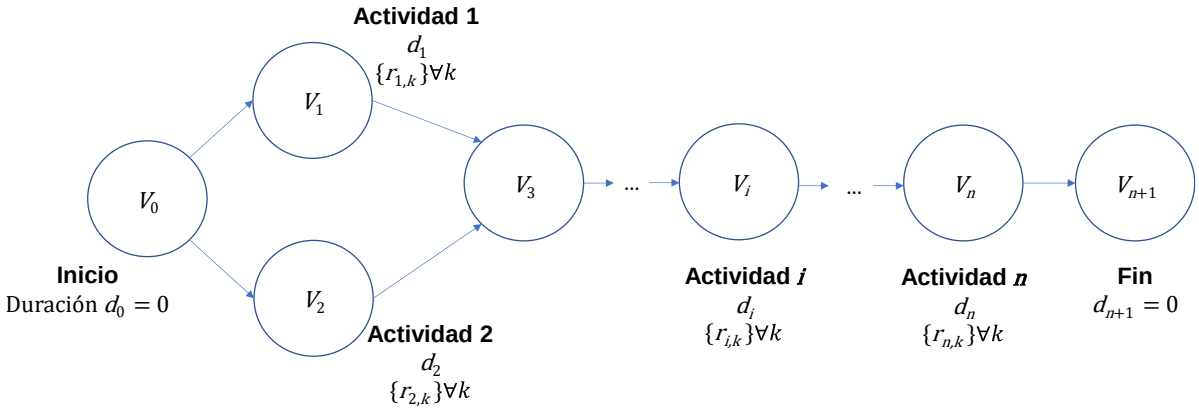


Figura 4. *Definición formal de un RCPSP.*

La limitación de escenarios impuesta por problemas como RCPSP puede ser superada por los **modelos basados en casos**, aportando soluciones novedosas. La manera para aproximarse a estos modelos a partir de la inteligencia artificial es a través de técnicas de aprendizaje automático.

Todos los modelos, generales y basados en casos, se han beneficiado de los avances en redes neuronales, algoritmos evolutivos y bioinspirados. Ello ha permitido el desarrollo de un gran número de técnicas de optimización que se han aplicado a la elaboración de cronogramas, que se discutirán en líneas generales en el siguiente apartado.

La revisión de Faghihi et al. (2015) relaciona las técnicas empleadas para la elaboración de cronogramas, distinguiendo entre los modelos generales y los basados en casos. Fazel et al., (2020) presentan un conjunto exhaustivo de las técnicas empleadas durante los últimos 40 años, habiendo investigado para ello más de 540 artículos. Es de reseñar asimismo el notable conjunto de metaheurísticas que presentan Pellerin et al. (2020) para resolver los RCPSP.

De modo general, las diferentes técnicas se subdividen en dos grupos: **técnicas básicas**, que contemplan un único algoritmo de inteligencia artificial, y **técnicas híbridas**, que combinan varias técnicas básicas. A continuación, se discutirán los ejemplos más relevantes de ambos grupos de técnicas que se encuentran en los trabajos relacionados.

2.2.4. Técnicas básicas aplicables a la elaboración de cronogramas

En este apartado se presentan, en primer lugar, las técnicas básicas que sirven de fundamento a las técnicas híbridas más relevantes: redes neuronales, lógica borrosa, árboles de decisión, aprendizaje por refuerzo y algoritmos evolutivos y, por último, los avances más recientes en técnicas básicas, fundamentalmente constituidos por sistemas emergentes bioinspirados. En cada caso se indicarán las ventajas y desventajas de cada técnica, así como casos de uso en problemas de planificación de proyectos.

A. Redes neuronales artificiales

Las redes neuronales imitan el principio de funcionamiento de las neuronas del cerebro humano, a través de un conjunto de nodos, llamados neuronas, interconectados entre sí, y que intercambian información.

La figura 5 ilustra el esquema de una neurona, en la que se puede observar cómo sus salidas a_i se generan a partir de una serie de entradas a_j, para las que se obtiene una combinación lineal in_i con una serie de pesos $W_{j,i}$. A esta combinación lineal se le aplica una función de activación g, resultando la salida de la neurona a_i.

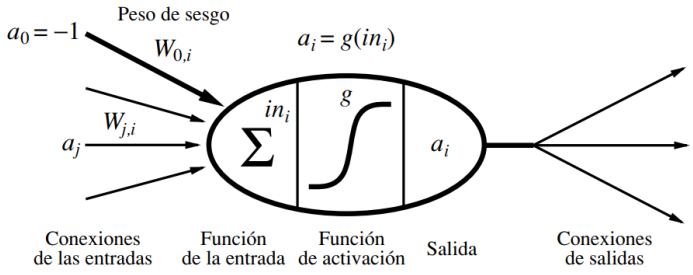


Figura 5. *Red neuronal. Esquema de una neurona* (Norvig y Russell, 2004).

Estas neuronas se combinan en una serie de capas, dando lugar a la red neuronal. Una red neuronal aprende a interpretar los conjuntos de datos con que se entrena y, con ello, predecir el comportamiento ante situaciones similares a las que ha sido entrenada. Un tipo de red habitualmente empleada es el perceptrón multicapa, de la que se muestra un ejemplo en la figura 6, con una capa de entrada de 10 neuronas, una capa oculta de 4 neuronas y una capa de salida con una única neurona. El algoritmo más empleado habitualmente para el aprendizaje es el de retropropagación. De todos modos, la red óptima para resolver un determinado problema es muy dependiente de este y, en todo caso, resulta complejo encontrar el óptimo global (Cheng, Lien, et al., 2012).

Las redes neuronales han sido implementadas en diversas vertientes de la gestión de proyectos. Sin embargo, no fue hasta 1997 cuando, por primera vez, se aplicó esta técnica a la elaboración de cronogramas de construcción, por parte de Adeli y Karim. En la revisión de Faghihi et al. (2015) se relacionan los hitos más relevantes desde finales de la década de 1990 hasta la década de 2010 en la investigación en la elaboración de cronogramas de construcción empleando redes neuronales: el método CONSCOM de Adeli y Karim (1997), el generador de estructuras de desglose de trabajo (*Work Breakdown Structure* o WBS) de Hashemi Golpayegani y Emamizadeh (2007) y el *Single-machine scheduling* de Avila Rondon et al. (2008), que elabora el cronograma a nivel de cada máquina individual.

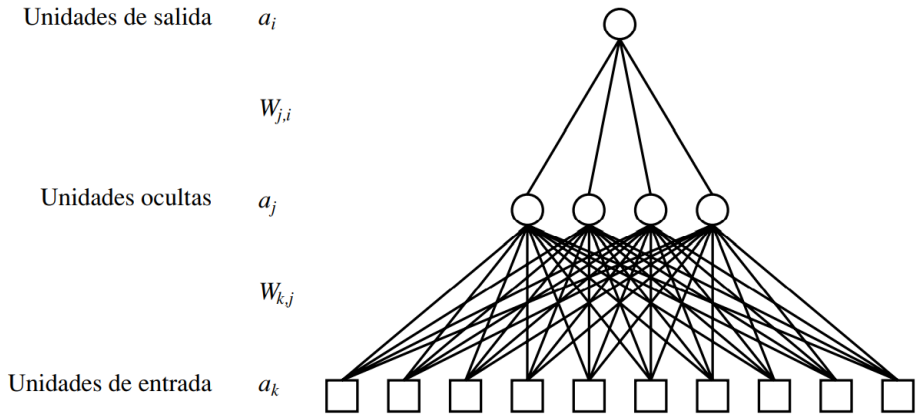


Figura 6. *Red neuronal del tipo perceptrón multicapa* (Norvig y Russell, 2004).

B. Lógica borrosa

La lógica borrosa nació como una herramienta para describir la incertidumbre y la inexactitud del conocimiento, imitando el modo en el que el cerebro humano toma decisiones frente a la incertidumbre o la vaguedad (Cheng, Lien, et al., 2012). Así, la lógica borrosa permite modelar sistemas complejos o difíciles de definir. Los modelos de lógica borrosa se componen de un "fuzzificador", que convierte las entradas en variables borrosas, una base de reglas borrosas,

un motor de inferencia a partir de dichas reglas y un “defuzzificador” que convierte las variables borrosas en salidas.

La lógica borrosa se expresa a través de una serie de “funciones de pertenencia”, que permiten representar la ambigüedad y, con posterioridad, establecer relaciones lingüísticas entre las variables. La figura 7 representa a través de un ejemplo muy sencillo el funcionamiento de este tipo de funciones de pertenencia, que describen de forma linguïstica el precio de un terreno en una zona determinada. Un terreno con un precio de 150.000 dólares se puede describir como con un grado de pertenencia μ = 0,3 al conjunto de “terrenos con precio bajo” y con un grado de pertenencia μ = 0,7 al conjunto de “terrenos con precio medio”.

Así, la lógica borrosa es una técnica ideal para tratar con ciertas características de los problemas de construcción (Fayek y Lourenzutti, 2018), como pueden ser la existencia de entradas y salidas inexactas, el uso de razonamientos heurísticos basados en la experiencia, el uso de información cualitativa, la existencia de varias soluciones posibles y/o que combinan aspectos contradictorios.

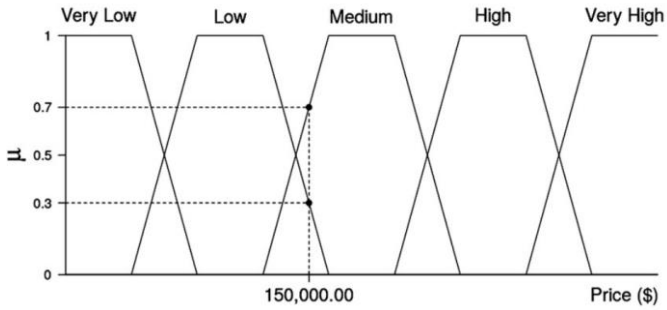


Figura 7. *Ejemplo de funciones de pertenencia* (Seresht et al., 2018).

La aplicación de lógica borrosa a los modelos permite dotarlos de explicabilidad, al introducir términos lingüísticos en lugar de estrictamente numéricos. Las limitaciones de la aplicación de lógica borrosa como técnica básica son su incapacidad para aprender de los datos y su dependencia del conocimiento experto y del contexto, que impide su generalización. Es por ello que la aplicación de lógica borrosa a la planificación de proyectos se realiza en conjunto con otras técnicas, lo que permite salvar las desventajas de cada técnica aislada. Este aspecto se tratará en mayor profundidad en el apartado correspondiente a las técnicas híbridas.

C. Árboles de decisión

Los árboles de decisión permiten establecer una salida, dado un conjunto de atributos de entrada y un esquema en forma de árbol, en el que se va avanzando hacia la salida, al tomar una decisión en cada nodo del árbol, en función del valor de los citados atributos. Los árboles de decisión son uno de los métodos más sencillos y con más éxito para elaborar modelos basados en aprendizaje automático, tanto de clasificación como de regresión (Norvig y

Russell, 2004). Otra de las ventajas de los modelos resultantes de la aplicación de árboles de decisión es su explicabilidad.

El rendimiento de un árbol de decisión aislado puede ser mejorado a través de diversas técnicas basadas en un conjunto de árboles o *ensemble*. Las más representativas son:

- *Boosting*, con algoritmos como *AdaBoost* o *XGBoost*. Consiste en una suma de varios árboles de decisión que se calculan de forma iterativa, de modo que cada nuevo árbol otorga un mayor peso a los ejemplos que han sido incorrectamente clasificados por los árboles anteriores.
- *Bagging,* como el algoritmo *Random forest*. Este algoritmo trabaja en paralelo con varios árboles de decisión, cada uno de los cuales contiene un subconjunto (con repetición) de la muestra inicial, y en cada nodo de selección se limita el número de variables de filtrado. El valor finalmente obtenido para la variable de salida es la media o la mediana del dado por el conjunto de árboles.

En la figura 8 se muestra un ejemplo de uno de los árboles de decisión de un modelo elaborado con *XGBoost*. A partir de los valores de las variables de entrada (‘Number of inverter (PS)’, ‘Number of trackers’, ‘Total Power’) se establece la predicción del valor de salida, en este caso, la duración en días (dada por el valor de cada hoja, ‘leaf’) del intervalo entre el inicio de dos tareas consecutivas para la construcción de una planta fotovoltaica, en este caso, la instalación del circuito de videovigilancia y la instalación del SCADA.

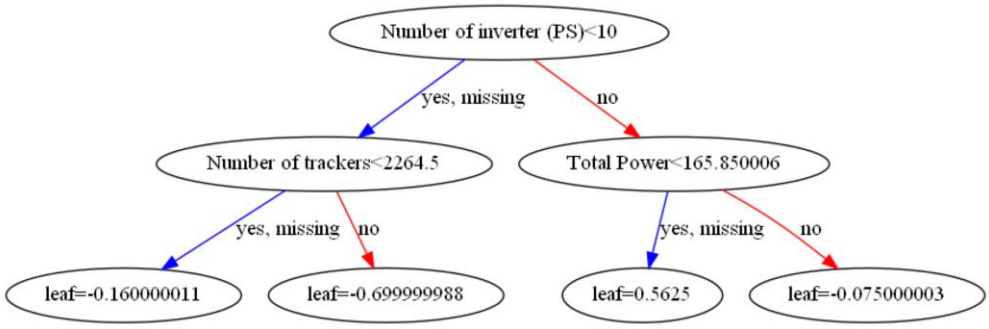


Figura 8. *Árbol de decisión*.

En la generación de cronogramas de proyectos, los árboles de decisión han sido empleados para la obtención de reglas de prioridad para la resolución del RCPSP (Guo et al., 2021).

D. Aprendizaje por refuerzo

El aprendizaje por refuerzo es un área del aprendizaje automático que tiene por objetivo enseñar a un agente a tomar una secuencia de decisiones óptimas en un entorno dado, a fin de maximizar una determinada recompensa -o minimizar un coste- (Sallam et al., 2021). Estas

recompensas o costes se acumulan tras cada acción. Así, esta técnica obtiene una "política" óptima con la que ejecutar una acción óptima en función del estado del entorno y, con ello, maximizar la recompensa total -o minimizar el coste total-.

En muchos dominios complejos, el aprendizaje por refuerzo es la única técnica posible para entrenar un programa para que funcione adecuadamente. Se trata de una técnica muy general y que abarca toda la inteligencia artificial, al enseñar a un agente cómo comportarse en su entorno (Norvig y Russell, 2004).

Una de las ventajas del aprendizaje por refuerzo es su capacidad de adaptación al entorno (Lewis y Vrabie, 2009). Por otro lado, la ausencia de un modelo y de conocimiento previo puede suponer una desventaja en ciertos entornos complejos (Norvig y Russell, 2004).

Aun cuando el aprendizaje por refuerzo ha sido ampliamente empleado para la resolución de problemas de optimización, se ha aplicado en menor medida a la elaboración de cronogramas de proyectos, si bien existen ejemplos recientes en la literatura en la elaboración de cronogramas de proyectos de *cloud manufacturing* (S. Chen et al., 2019) así como para la resolución de RCPSP (Pellerin et al., 2020; Sallam et al., 2021).

E. Algoritmos evolutivos

Los algoritmos evolutivos pretenden simular la evolución de las especies: dado que la inteligencia natural es producto de la evolución, podría obtenerse una inteligencia artificial aplicando procesos evolutivos a los algoritmos (Abraham, 2002). El principio de funcionamiento de los algoritmos evolutivos viene dado por los tres siguientes pasos: en primer lugar, se parte de una población de individuos, generada de forma aleatoria. Cada uno de ellos representa una posible solución al problema. A continuación, se evalúa el desempeño de cada individuo como posible solución al problema. Por último, se genera una nueva población, por evolución de la preexistente. Este proceso se lleva a cabo hasta que alguno de los individuos cumple las condiciones fijadas para la resolución del problema (Kachitvichyanukul, 2012).

La aplicación de estos algoritmos requiere, en primer lugar, determinar cómo se representará el problema en los términos del algoritmo evolutivo. Debe tenerse en cuenta que la inicialización y los procesos de evolución pueden dar lugar a individuos que produzcan soluciones no factibles en la práctica. Además, es preciso seleccionar -habitualmente, de forma empírica- el tamaño de la población y el máximo número de iteraciones, que determinarán los parámetros de rendimiento más importantes: la calidad de la solución y el tiempo de ejecución.

Los algoritmos evolutivos son uno de los métodos de optimización que han recibido mayor atención por parte de los investigadores en los últimos años y han sido ampliamente aplicados en el área de la elaboración de cronogramas de proyectos, siendo esta tipología de algoritmos la que ha devuelto mejores resultados en la resolución de problemas complejos para la elaboración de cronogramas de proyectos, como RCPSP (Pellerin et al., 2020). A continuación, se describirán ejemplos concretos de algoritmos evolutivos que han sido aplicados en este tipo de tareas: los algoritmos genéticos y la evolución diferencial. Los primeros resultan más aptos para problemas de optimización discretos, mientras que los segundos son preferibles para optimización continua.

Algoritmos genéticos

Su principio de funcionamiento es el fundamental de cualquier algoritmo evolutivo: replicar la selección natural y, con ello, la supervivencia de los individuos que mejor se adaptan. En los algoritmos genéticos, las soluciones se representan como cromosomas, que se evalúan y ordenan de mejor a peor según su valor conforme a una función de aptitud o *fitness*. Las nuevas soluciones se generan simulando la selección natural a través de la aplicación de forma repetida de tres operadores genéticos (Kachitvichyanukul, 2012):

- **Selección**: los mejores cromosomas son seleccionados como “padres” para producir descendencia. Para ello, cuanto mayor sea valor de la función de *fitness* de un cromosoma, mayor probabilidad de supervivencia se le asigna.
- **Cruce**: en esta operación, se combinan los cromosomas padres para producir descendencia.
- **Mutación**: con las operaciones anteriores, existe una tendencia a que las soluciones obtenidas sean similares, lo que conduce a su estancamiento. Para ello, y a fin de mantener la diversidad de la población, se realizan mutaciones.

La figura 9 muestra de forma gráfica los conceptos de población, individuo y gen, así como el detalle de las operaciones de cruce y mutación.

Los algoritmos genéticos no presentan grandes requerimientos matemáticos. Su implementación simple permite incorporarlos con éxito a modelos híbridos, tal y como se comprobará en el siguiente apartado. Al mismo tiempo, sus propiedades los hacen efectivos y robustos. Aun cuando no siempre logran obtener el óptimo global, las soluciones subóptimas que encuentran suelen ser aceptables y permiten resolver problemas en múltiples áreas donde otras técnicas encuentran dificultades. En cambio, las versiones simples de estos algoritmos son de difícil implementación directa a la hora de resolver con éxito problemas de optimización complejos.

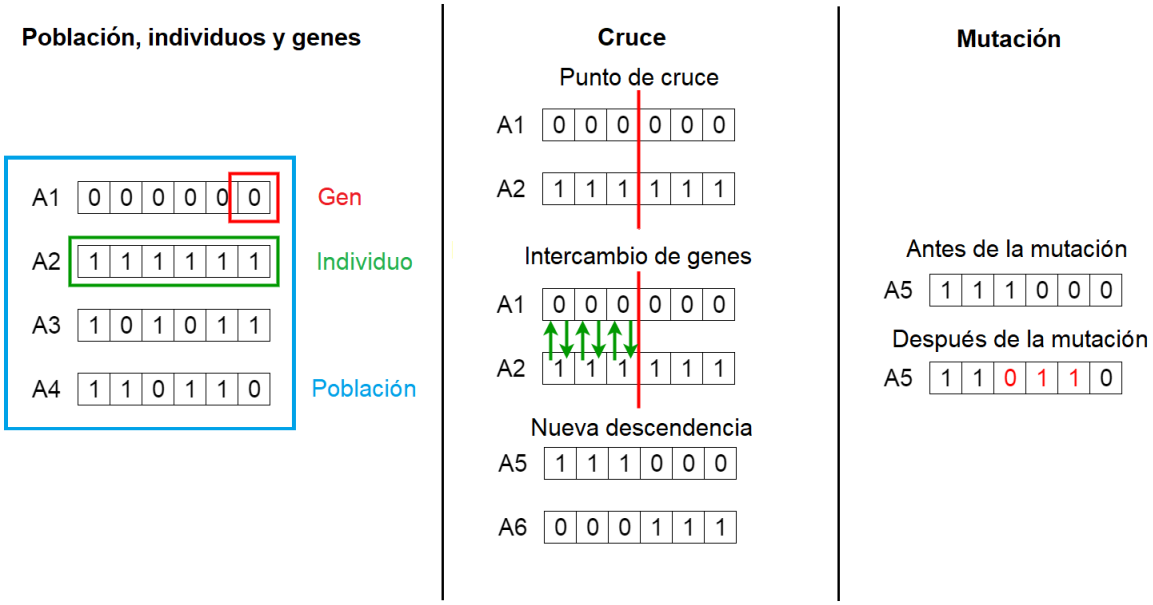


Figura 9. *Algoritmo genético*.

A pesar de los buenos resultados mencionados en su aplicación a problemas de planificación de múltiples proyectos (Ko y Cheng, 2003), es importante destacar que, en algunos casos, las soluciones quedan fuera del alcance del algoritmo, ya que las mejores soluciones no siempre cumplen con todas las restricciones del problema.

Evolución diferencial

La técnica de evolución diferencial permite resolver problemas de optimización de una forma versátil y con buenos resultados. Tal y como el resto de algoritmos evolutivos, se basa en una población inicial. Los elementos de esta población inicial son vectores generados aleatoriamente a partir de una distribución uniforme dentro del hiperespacio de posibles soluciones. La diferencia de esta técnica radica en que genera la descendencia de un vector a través de modificaciones sobre este, obtenidas con la diferencia escalada de otros dos vectores seleccionados aleatoriamente. El vector resultante solamente reemplaza a su "padre" si su rendimiento es superior.

En su definición original, para cada vector x_i de la población inicial, se extraen tres vectores x_r , x_s y x_t de forma pseudoaleatoria (Neri y Tirronen, 2010). Con estos tres vectores se genera una descendencia

$$x'_{desc} = x_t + F(x_r - x_s) \qquad (1a)$$

F es un factor de mutación, que controla la longitud del vector de exploración $(x_r - x_s)$ y, de ese modo, define la distancia de x_i a su descendencia. F puede adoptar valores entre 0 y 1 o, incluso, ligeramente superiores a 1.

En su implementación más habitual, el vector x_t suele ser el que, dentro de toda la población, ofrece mejor valor de una función de *fitness* (SciPy, 2020). Una vez se ha generado la descendencia provisional x'_{desc}, cada "gen" (componente) de este vector puede ser intercambiado con el correspondiente gen de x_i, de la siguiente forma:

$$x_{desc,j} = \begin{cases} x'_{desc,j} & \text{si rand}(0,1) < C_R \\ x_{i,j} & \text{en caso contrario} \end{cases} \quad (1b)$$

, donde j es el índice del gen que se considera, $rand(0,1)$ es un número aleatorio entre 0 y 1 y C_R es un parámetro conocido como constante de recombinación. El descendiente reemplaza a la generación anterior si su *fitness* es superior.

Su marco teórico sencillo y facilidad de implementación -que permiten su control con relativamente pocas variables-, así como su fiabilidad y alto rendimiento -con buenas propiedades de convergencia-, han hecho que esta técnica sea ampliamente empleada.

La principal desventaja de esta técnica es la limitación de posibles movimientos exploratorios y, con ello, la posibilidad de estancamiento en ciertas zonas del hiperespacio de soluciones, sin llegar siquiera a una solución subóptima. Otras desventajas son la dificultad del ajuste de hiperparámetros en problemas complejos y su tendencia a sufrir de maldición de la dimensión. Este tipo de cuestiones han tratado de subsanarse con distintas versiones posteriores del algoritmo (Das y Suganthan, 2011; Neri y Tirronen, 2010).

Dentro del área de elaboración de cronogramas, este algoritmo ha sido empleado para la resolución del RCPSP (W. Chen y Ni, 2014; Rahmani et al., 2015).

F. Sistemas emergentes bioinspirados

Los sistemas emergentes bioinspirados son aquellos que toman como modelo sistemas naturales que en conjunto manifiestan propiedades que no poseen cada una de las partes de dicho sistema. Estos constituyen por sí mismos un importante grupo de algoritmos que han recibido gran atención por parte de la comunidad investigadora. Este tipo de sistemas han sido ampliamente empleados en gestión de proyectos y elaboración de cronogramas, destacando los siguientes: colonia de hormigas, enjambre de partículas, colonia de abejas, algoritmo de luciérnaga, *shuffle frog-leaping* y colonia de termitas. Pueden encontrarse abundantes ejemplos en Pellerin et al. (2020); Seresht et al. (2018) y Tiruneh et al. (2020). A continuación, se discutirán los dos primeros algoritmos, por resultar los más representativos y los más ampliamente utilizados, así como *shuffle frog-leaping*, que ofrece resultados prometedores al incorporar características de los algoritmos evolutivos.

Colonias de hormigas

Este algoritmo metaheurístico se basa en el comportamiento de las hormigas en la búsqueda de comida: aunque estos insectos prácticamente carecen de visión, son capaces de encontrar la ruta más corta entre su nido y la comida. El modo en que lo consiguen es a través del uso de feromonas, que depositan durante su recorrido, y que sirven como método de comunicación. Aunque cada hormiga siga una trayectoria con una componente aleatoria, se encontrará atraída por feromonas que han dejado otras hormigas en su trayectoria. A su vez, una trayectoria que atrae más hormigas dispondrá de una mayor cantidad de feromonas, aumentando nuevamente la atracción de nuevas hormigas y produciéndose una realimentación positiva. Dado que las hormigas que elijan el camino más corto serán aquellas que primero alcancen el nido, depositarán antes feromonas en su trayectoria, incrementando la probabilidad de que las hormigas seleccionen el camino más corto (Dorigo et al., 2019; Eusuff et al., 2006). La figura 10 muestra un estado intermedio de la búsqueda de comida por parte de una colonia de hormigas. El grosor de la línea roja refleja la densidad de feromonas en cada punto del camino. Dado que el camino superior es más corto, permitirá la vuelta de las hormigas en un tiempo más corto y, con ello, el depósito de feromonas a mayor velocidad.

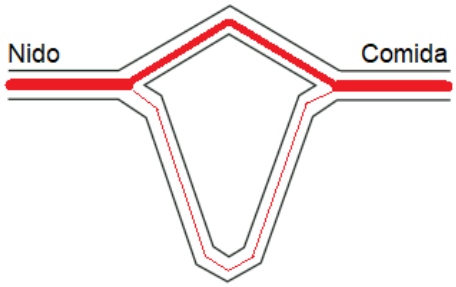


Figura 10. *Trayectoria seguida por una colonia de hormigas en búsqueda de comida.*

Se ha probado que algunos de estos algoritmos son capaces de converger a soluciones óptimas, si bien el tiempo de ejecución puede ser muy elevado. En cualquier caso, resultan un método muy adecuado para la resolución de problemas complejos, obteniendo resultados próximos a los de los mejores algoritmos o incluso siendo la referencia en el estado del arte.

Esta idea ha sido aplicada también en la generación de cronogramas (Merkle et al., 2002), del siguiente modo: un grupo de hormigas busca un cronograma factible de acuerdo a una serie de heurísticas simples, que ordenan la prioridad de las tareas, y a una matriz de feromonas dinámica. De este modo, cada hormiga realiza un recorrido por las distintas tareas. El cronograma resultante en cada iteración será el camino seguido por la mayoría de las hormigas. Antes de comenzar una nueva iteración, se actualiza la matriz de feromonas. El proceso se detiene cuando las soluciones no mejoran con el paso de las iteraciones o bien

cuando se alcanza un número determinado de generaciones. Esta metaheurística no garantiza alcanzar la solución óptima, si bien existen técnicas que mejoran la búsqueda y las soluciones alcanzadas, como son: el cruce de tareas entre distintas soluciones -que mejora la búsqueda de nuevas soluciones-, la mutación por inversión -que impide que se restrinja el hiperespacio de exploración de soluciones-, el elitismo -que premia a aquellas soluciones estrictamente mejores que otras, otorgándoles un mayor número de feromonas-, así como el empleo de búsqueda hacia delante y hacia atrás.

Enjambre de partículas

Los algoritmos de de optimización de enjambre de partículas (*particle swarm optimization*, PSO) se inspiran en el movimiento de sistemas de organismos vivos, como bandadas de aves. Han sido objeto de numerosas publicaciones y han sido aplicados con éxito a la resolución de múltiples problemas de optimización complejos (Kachitvichyanukul, 2012). Estos algoritmos han prosperado por su marco conceptual sencillo y la facilidad de comprensión del proceso de búsqueda de soluciones por su analogía con los sistemas biológicos.

En un enjambre de partículas, cada partícula i representa una posible solución. Cada partícula se mueve en el instante t a una nueva posición $x_{i,t}$, sumando su velocidad $V_{i,t}$ a su posición anterior $x_{i,t-1}$:

$$x_{i,t} = x_{i,t-1} + V_{i,t} \tag{2}$$

Inicialmente esta posición y velocidad son aleatorias.

Una vez que las partículas alcanzan su nueva posición, se evalúa la función de *fitness* para cada una de ellas y se actualiza cuál ha sido el mejor *fitness* de la partícula y del enjambre.

Con esta información, se actualiza la velocidad de cada partícula $V_{i,t}$ como la suma de:

$$V_{i,t} = wV_{i,t-1} + c_1 r_1 \left(x_{i,best} - x_{i,t-1}\right) + c_2 r_2 \left(x_{global,best} - x_{i,t-1}\right) \tag{3}$$

- La velocidad anterior $V_{i,t-1}$, ponderada por un factor de inercia w. Cuanto mayor sea ese factor, en mayor grado se favorecerá una exploración global. Un valor pequeño favorecerá una búsqueda local.
- La diferencia de la posición de la partícula con su mejor posición histórica $x_{i,best}$. Esta diferencia se pondera con un "parámetro cognitivo" c_1, así como con un factor aleatorio r_1 comprendido entre 0 y 1. Este sumando representa el conocimiento individual de la partícula.
- La diferencia de la posición de la partícula con la mejor posición histórica de todas las partículas del enjambre $x_{global,best}$. Esta diferencia se pondera con un "parámetro social"

c_2, así como con un factor aleatorio r_2 comprendido entre 0 y 1. Este sumando dota a la partícula de conocimiento basado en la experiencia grupal.

Este proceso se repite hasta que se alcanza el criterio de parada. Como se puede observar, la actualización de la posición y la velocidad es muy simple. Otro factor que reduce la complejidad computacional es que no resulta necesario ordenar las soluciones en base a su *fitness*, lo que otorga un buen rendimiento a este algoritmo cuando se trabaja con poblaciones grandes. Una desventaja de esta técnica, común a la mayoría de metaheurísticas, es que tiende a devolver soluciones subóptimas.

La figura 11 muestra la trayectoria seguida por un enjambre de partículas, desde su posición inicial aleatoriamente distribuida, hasta una posición próxima a la final, donde todas ellas convergen a una solución.

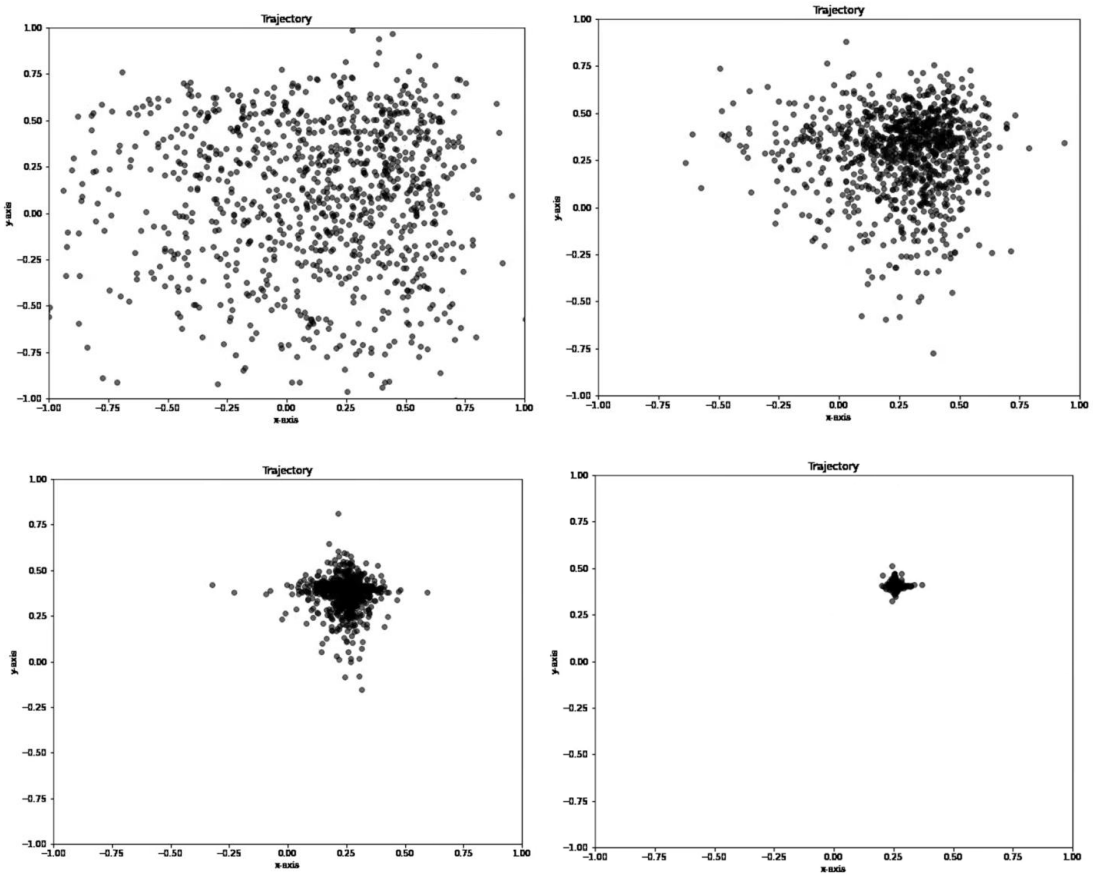


Figura 11. *Trayectoria seguida por un enjambre de partículas.*

Este algoritmo ha sido empleado por Zhang et al. (2006) para la resolución del RCPSP, considerando cada partícula como un posible cronograma, con la capacidad de encontrar óptimos globales, obteniendo incluso mayor eficiencia que los algoritmos genéticos. También ha sido utilizado para obtener planificaciones de trabajos (Ge et al., 2008).

Shuffled frog-leaping

El algoritmo *shuffled frog-leaping* se inspira en el proceso que utilizan las ranas para buscar comida. La analogía se basa en un grupo de ranas situadas en un estanque en el que buscan comida. En este estanque se disponen un conjunto de piedras en localizaciones discretas, a las cuales las ranas pueden saltar, para así encontrar aquella que dispone de la mayor cantidad de comida. Las ranas pueden comunicarse entre ellas, por lo que pueden optimizar su búsqueda con la información de otras ranas (Eusuff et al., 2006; Fang y Wang, 2012).

Así, este algoritmo se basa tanto en la **búsqueda local** como en el **intercambio de información global**. Inicialmente se forma una población inicial aleatoria de posibles soluciones, compuesta por las llamadas “ranas virtuales”. Cada una de ellas se encuentra representada por un vector.

La población total se reparte en una serie de subconjuntos, llamados *memeplexes* (*meme*: idea perteneciente a una cultura). Cada uno de estos *memeplexes* se corresponde con un grupo cultural de ranas, que evoluciona independientemente en el espacio de búsqueda de soluciones.

El algoritmo realiza una búsqueda local dentro de cada *memeplex* utilizando un método similar al enjambre de partículas. La búsqueda local dentro de cada *memeplex* se realiza por subgrupos, distribuidos de forma semialeatoria de modo que se permita dar más peso a las mejores ranas. Dentro de cada subgrupo, la peor rana se dirige hacia la posición de la mejor rana, con un salto de longitud aleatoria (limitado por una longitud máxima de salto). De ese modo, se va actualizando la posición de la nueva peor rana en base a la posición de la nueva mejor rana.

Para asegurar la exploración global, se produce una evolución memética, que permite que las ranas reciban mejores ideas. Esto se consigue reorganizando a las ranas virtuales cada cierto número de iteraciones en nuevos *memeplexes*, utilizando un método similar a un algoritmo evolutivo conocido como *shuffled complex evolution algorithm*. Además, se introduce nueva información aleatoria gracias a nuevas ranas virtuales que se generan de forma aleatoria y sustituyen a otras ranas de la población.

Este proceso de saltos y reorganización se produce hasta que se alcance alguna condición de parada.

El buen funcionamiento de este algoritmo depende fundamentalmente del tamaño de la población, que está relacionado con la complejidad del problema. Una mayor población aumenta la probabilidad de obtener mejores soluciones, si bien aumenta el coste computacional.

La comparativa de rendimiento de *shuffled frog-leaping* con algoritmos genéticos en una serie de problemas teóricos y prácticos ha demostrado su mejor rendimiento, o al menos comparable, y mayor robustez en la búsqueda de la solución global (Eusuff et al., 2006).

En el ámbito de la planificación de proyectos de construcción, este algoritmo ha sido utilizado en RCPSP (Fang y Wang, 2012).

C. Otras técnicas básicas relevantes

Otras técnicas básicas de inteligencia artificial que han sido empleadas para la elaboración automática de cronogramas (Fazel et al., 2020) son el algoritmo de búsqueda tabú, que explora el espacio de búsqueda evitando soluciones repetidas a través de una "lista tabú", y la técnica de recocido simulado, que inspira su algoritmo de optimización en el proceso de recocido del acero, por el cual se calienta y luego enfría lentamente, de modo que los átomos varían su estado a posiciones vecinas aleatorias, para recristalizar en posiciones de mínima energía, asimilables a los mínimos locales de la función a optimizar.

2.2.5. Técnicas híbridas aplicables a la elaboración de cronogramas

La hibridación es el proceso por el que se combinan dos o más técnicas básicas, para integrar sus puntos fuertes y soslayar sus debilidades individuales (Seresht et al., 2018). Así, de la combinación de distintas versiones de las técnicas básicas resulta un amplio inventario de técnicas híbridas. La comparativa de Pellerin et al. (2020) y la investigación de Gil et al., (2021b) relacionan una muestra significativa de dichas técnicas híbridas. Debido a su extenso número, en este apartado se desarrollarán únicamente las principales líneas de estudio, que integran fundamentalmente lógica borrosa, redes neuronales y algoritmos evolutivos (Abraham, 2002), siguiendo la filosofía de complementariedad antes citada: los **sistemas borrosos** dotan de explicabilidad a las soluciones. El inconveniente que presentan es el ajuste de sus parámetros. Los **algoritmos evolutivos**, no tan precisos en la búsqueda local, permiten la optimización global de parámetros y el uso de **redes neuronales,** cuyos métodos de descenso de gradiente no garantizan la obtención de óptimos globales, permite optimizar la búsqueda local.

A. Neuroevolución

Partiendo de lo expuesto anteriormente, surgen las técnicas conocidas como neuroevolución. Estas consisten en la hibridación de redes neuronales con algoritmos genéticos, de tal modo que un algoritmo genético sea el responsable de la evolución de la red neuronal. Así, la neuroevolución realiza una búsqueda en un espacio de redes neuronales (Stanley y Miikkulainen, 2002). Se trata de una aproximación no estadística a la resolución de problemas

complejos, que ha devuelto soluciones más rápidas y más eficientes que otras heurísticas y metodologías. Dado que los algoritmos neuroevolutivos buscan un "comportamiento" en lugar de una función de valor, su resultado es bueno en problemas continuos y con un elevado número de dimensiones.

Las primeras técnicas de neuroevolución se basaban en una topología dada de la red neuronal, en la que el número de capas, neuronas y conexiones era fija. En estas técnicas, el algoritmo genético realiza una búsqueda de los pesos más adecuados para la red, a través de cruce, mutación y selección de vectores de pesos. Estos algoritmos neuroevolutivos tienen como objetivo, pues, la optimización de los pesos de la red.

Con posterioridad, las técnicas de neuroevolución ampliaron su alcance a la determinación de la topología de la red. En estos casos, el algoritmo genético realiza una búsqueda de la topología óptima. Algoritmos como NEAT (*NeuroEvolution of Augmenting Topologies*, Stanley y Miikkulainen, 2002) buscan incluso la topología de menor dimensión que optimice la tarea de la red. Algoritmos basados en redes *feed-forward* y *fully-connected* logran disminuir el tiempo de procesado para la búsqueda de una red óptima en torno a un 80% (Harvey, 2017).

Si bien su rendimiento en cuanto a métricas es muy bueno, estas técnicas tienen la desventaja de que se tratan de un modelo de "caja negra", no explicable.

En el área de la elaboración de cronogramas, estas técnicas han sido empleadas para la resolución de RCPSP por Agarwal et al. (2011), en un algoritmo en el que se aplican iteraciones en serie combinando algoritmos genéticos y redes neuronales.

B. Sistemas neurodifusos

Los sistemas neurodifusos (*neuro-fuzzy*) son modelos híbridos que combinan el poder de aprendizaje de las redes neuronales y la funcionalidad de la lógica borrosa, mejorando su capacidad de razonamiento e inferencia y manteniendo una representación explícita del conocimiento (Tiruneh et al., 2020). Existen diferentes formas de combinar sistemas difusos y redes neuronales, con un resultado que mejora el de los métodos individuales, al aprovechar la complementariedad existente entre las redes neuronales y la lógica borrosa. La potencia de los sistemas neurodifusos para resolver problemas reales viene dada por la representación explícita del conocimiento, el aprendizaje automático y la capacidad de utilizar variables lingüísticas para modelar las relaciones entrada-salida de un sistema. En particular, posee un gran potencial para varias aplicaciones relacionadas con la gestión de proyectos de construcción debido a sus características robustas, rápidas y efectivas para resolver problemas complejos.

Los sistemas neurodifusos se pueden distinguir en función de las tres siguientes características (Seresht et al., 2018):

1. La **estructura** de su componente borrosa, que en su mayoría es del tipo Mamdani o Takagi-Sugeno. Los sistemas de tipo Takagi-Sugeno se caracterizan fundamentalmente porque sus reglas devuelven el valor de una función, integrando así el proceso de defuzzificación (Guzmán y Castaño, 2006). Tienen un alto rendimiento, pero sus procedimientos de aprendizaje son complicados y son computacionalmente costosos. Los sistemas de tipo Mamdani utilizan reglas de tipo "SI antecedente ENTONCES consecuente", con una salida lingüística que requiere un proceso de defuzzificación. Estas heurísticas son más rápidas pero pueden comprometer su rendimiento.
2. El **algoritmo** de aprendizaje, p.ej., redes neuronales con algoritmo de retropropagación.
3. Las **características** del sistema que están determinadas por el algoritmo de aprendizaje, es decir, forma y/o número de funciones de pertenencia difusas, número de reglas difusas y/o pesos de reglas.

Los sistemas neurodifusos constan de cinco capas de procesamiento de datos:

1. Capa de entrada, donde se introducen las entradas del sistema.
2. Capa de fuzzificación, que determina los valores de pertenencia de las entradas a los conjuntos difusos.
3. Capa de inferencia, que asigna las entradas difusas a salidas difusas mediante un sistema de reglas borrosas.
4. Capa de defuzzificación, que transforma las salidas difusas de la capa de inferencia en valores numéricos concretos.
5. Capa de salida, que entrega la salida final del sistema.

Las metodologías para desarrollar sistemas neurodifusos utilizan el algoritmo de aprendizaje para obtener las capas de *fuzzificación* -por ejemplo, la forma y/o el número de funciones de pertenencia difusas- e inferencia -por ejemplo, el número de reglas difusas y/o pesos de las reglas en la base de reglas-.

ANFIS

Entre las técnicas neurodifusas que no incluyen algoritmos genéticos, la que mejores resultados ha conseguido (Seresht et al., 2018; Tiruneh et al., 2020) ha sido *adaptive neuro-*

fuzzy inference system o **ANFIS**, creada por Jang (1993). Dentro de este grupo, esta técnica ha sido, además, la más ampliamente utilizada.

ANFIS es un sistema borroso que ajusta los parámetros de las reglas con una red neuronal. Puede implementar diversos algoritmos de aprendizaje, como los basados en gradiente o en poblaciones, entre otros. ANFIS da forma automáticamente a las reglas que mapean los datos de entrada y salida, averiguando, además, las funciones de pertenencia óptimas. Para ello, utiliza la capacidad de aprendizaje de las redes neuronales. Se trata de una herramienta eficaz para reconocer patrones y generar modelos precisos de sistemas. Una ventaja de ANFIS es que no necesita la opinión de expertos para modelar y entrenar estos sistemas (Aengchuan y Phruksaphanrat, 2018).

Una arquitectura típica de ANFIS se muestra en la figura 12, como ejemplo del modelado de una función $f(x, y)$ de dos entradas y una única salida (Jang, 1993). Los nodos redondos son fijos, mientras que los nodos rectangulares son nodos que tienen parámetros que deben ser aprendidos (nodos adaptativos).

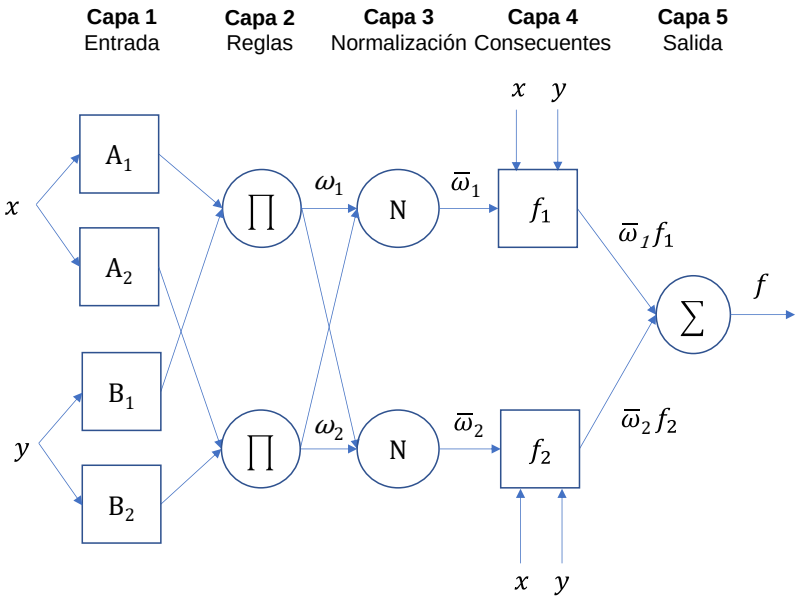


Figura 12. *Ejemplo de arquitectura de ANFIS.*

Este sistema borroso de ejemplo cuenta con dos reglas borrosas de tipo Takagi-Sugeno:

$$\text{Regla 1: SI } x \text{ es } \mathrm{A}_1 \text{ Y } y \text{ es } \mathrm{B}_1, \text{ ENTONCES } f_1 = p_1 x + q_1 y + r_1$$

$$\text{Regla 2: SI } x \text{ es } \mathrm{A}_2 \text{ Y } y \text{ es } \mathrm{B}_2, \text{ ENTONCES } f_2 = p_2 x + q_2 y + r_2 \qquad (4)$$

, donde $\mathrm{A_i}$ y $\mathrm{B_i}$ son los conjuntos borrosos de entrada y $p_i, q_i, r_i\ (i = 1{,}2)$ son los *parámetros consecuentes*, que es preciso obtener durante el entrenamiento. En la medida en que x e y cumplan los antecedentes de cada regla borrosa, se generará un consecuente $f_i\ (i = 1{,}2)$ por

cada una de las reglas. La salida f del sistema borroso será la agregación de estos dos consecuentes. A continuación, se detalla el funcionamiento de cada una de las capas:

- **Capa 1: nodos de entrada**. Su salida es el resultado de evaluar la función de pertenencia de x e y a cada uno de los conjuntos borrosos $\mathrm{A_i}$ y $\mathrm{B_i}$ $(i = 1{,}2)$, esto es, en qué grado x e y satisfacen cada cuantificador $\mathrm{A_i}$ y $\mathrm{B_i}$. La forma de cada uno de los conjuntos borrosos se encuentra definido por unos parámetros denominados *de premisa*, que deberán ser calculados.

- **Capa 2: nodos de reglas**. Sus salidas ω_i $(i = 1{,}2)$ expresan la denominada *fuerza de activación* de cada regla, esto es, el peso de cada regla. En este ejemplo, la conjunción Y de las salidas de la capa 1 se calcula como el producto de estas. Por ejemplo, si la variable de entrada x tiene un grado de pertenencia de 0,7 al conjunto $\mathrm{A_1}$ y la variable de entrada y tiene un grado de pertenencia de 0,5 a $\mathrm{B_1}$, el peso absoluto de la regla f_1 es de 0,35.

- **Capa 3: nodos de normalización**. una vez conocidos los pesos absolutos ω_i de las i reglas, este nodo calcula el peso relativo $\bar{\omega}_i$ de cada regla dentro de todo el conjunto de reglas. El peso relativo se conoce como *fuerza de activación normalizada*:

$$\bar{\omega}_i = \frac{\omega_i}{\omega_1 + \omega_2} \quad , i = 1{,}2 \tag{5}$$

- **Capa 4: nodos consecuentes**. Su salida viene dada por la función f_i correspondiente a cada regla, ponderada con su fuerza de activación normalizada:

$$\bar{\omega}_i f_i = \bar{\omega}_i (p_i x + q_i y + r_i) \quad , i = 1{,}2 \tag{6}$$

  Así, en esta salida aparece el conjunto de parámetros a calcular p_i, q_i, r_i $(i = 1{,}2)$ que corresponden a las reglas Takagi-Sugeno.

- **Capa 5: nodo de salida**. Este computa la suma de los nodos anteriores, que puede expresarse:

$$f = (\overline{\omega_1} x)p_1 + (\overline{\omega_1} y)q_1 + (\overline{\omega_1})r_1 + (\overline{\omega_2} x)p_2 + (\overline{\omega_2} y)q_2 + (\overline{\omega_2})r_2 \tag{7}$$

El método que habitualmente emplea ANFIS para entrenar los parámetros es, para los parámetros consecuentes, el de mínimos cuadrados, y para los parámetros de premisa, el de descenso de gradiente.

Tal y como se ha indicado con anterioridad, ANFIS ha obtenido muy buenos resultados en una amplia variedad de problemas, permitiendo además su interpretación. En la elaboración automática de cronogramas, también ha sido aplicado para agendar trabajos realizados por maquinaria (Azadeh et al., 2015) La principal desventaja de ANFIS es su coste computacional

y que puede generar modelos complejos para problemas relativamente sencillos (Tiruneh et al., 2020). Por otro lado, es preciso seleccionar una topología y unas funciones de pertenencia adecuadas en cada aplicación específica (Ko y Cheng, 2003).

C. Sistemas neurodifusos con algoritmos evolutivos

Aunque los sistemas neurodifusos suponen un avance con respecto a la lógica borrosa *per se* a la hora de simular las características y el proceso de inferencia humana, presentan problemas como los expuestos por Ko y Cheng, que requieren de una gran cantidad de tiempo para su resolución, y que aumentan con la complejidad del problema.

EFNIM y EFHNN

Según estos autores, los algoritmos genéticos son una técnica efectiva para resolver estos problemas. ***Evolutionary Fuzzy Neural Inference Model* (EFNIM)** es un modelo que va más allá de ANFIS al introducir un algoritmo evolutivo para decidir las formas más adecuadas de las funciones de pertenencia, la topología óptima y los parámetros óptimos del del sistema neurodifuso (Ko y Cheng, 2003).

EFNIM consiste en un sistema borroso en la capa de inferencia y un algoritmo de aprendizaje basado en redes neuronales que se utiliza para determinar los pesos optimizados de las reglas del sistema borroso. Finalmente, EFNIM emplea algoritmos genéticos para optimizar la estructura del sistema borroso, especificando el número y la forma de las funciones de pertenencia, así como el número de reglas (Ko y Cheng, 2003). La arquitectura general de EFNIM se muestra en la figura 13.

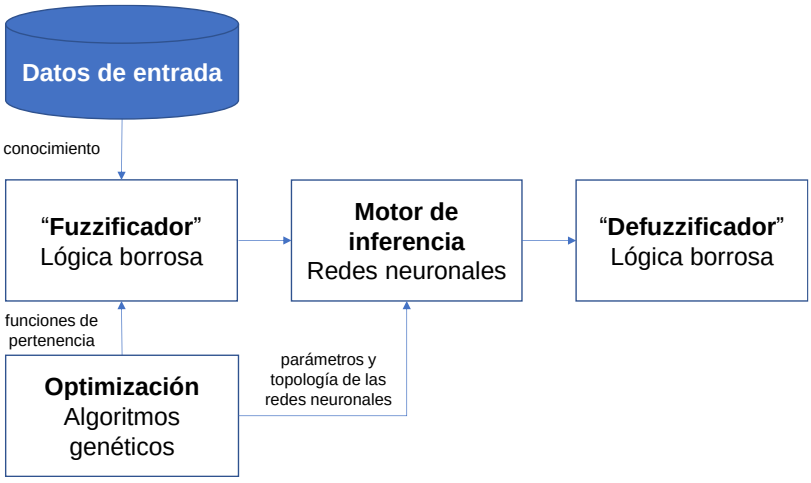


Figura 13. *Arquitectura general de EFNIM.*

EFNIM es capaz de autoadaptarse a través de un algoritmo genético, a través de las operaciones de selección, cruce y mutación. En el modelo de Ko y Cheng (2003), cada individuo codifica las variables del modelo en una cadena binaria, que simula un cromosoma

natural. Cada cadena comprende dos segmentos: uno con las funciones de pertenencia y otro con la "red neuronal borrosa", que integra el sistema borroso con la red neuronal, según se puede apreciar en la figura 14.

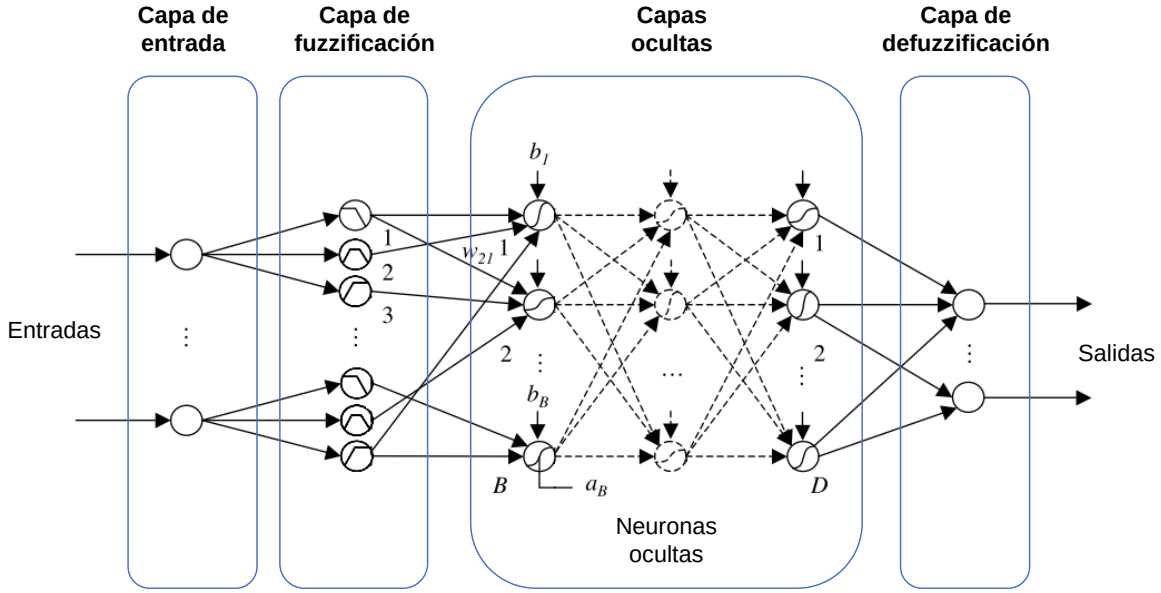


Figura 14. *Arquitectura de una red neuronal borrosa.*

Una evolución de EFNIM es la técnica conocida como ***Evolutionary Diffuse Hybrid Neuronal Network*** **(EFHNN**), propuesta por Cheng, Tsai y Sudjono (2009), que, en lugar de redes neuronales tradicionales, emplea redes neuronales híbridas (*hybrid neural networks*, HNN). Este tipo de redes híbridas combinan:

- Una red neuronal tradicional, compuesta por funciones de términos lineales:

$$y_j = f\left(\sum_i w_{ji}x_i + b_j\right) \tag{8}$$

- Una red neuronal de alto orden, compuesta por funciones de términos no lineales:

$$y_j = f\left(\prod_i x_i^{p_{ij}}\right) \tag{9}$$

Los términos de alto orden permiten capturar mejor las no linealidades y, con ello, mostrar un mejor desempeño que las redes neuronales tradicionales.

Dentro del ámbito de la elaboración automática de cronogramas, estos algoritmos han sido aplicados a la planificación de proyectos de plantas fotovoltaicas (Gil et al., 2021a).

D. Otras técnicas híbridas de optimización relevantes

Además de las técnicas mencionadas, a continuación se relacionan algunas de las más relevantes, como la hibridación de sistemas basados en reglas borrosas con algoritmos evolutivos o sistemas emergentes bioinspirados (Seresht et al., 2018), sin hibridación con

redes neuronales. Otra técnica que prescinde del uso de redes neuronales es *Evolutionary Fuzzy Support Vector Machines Inference Model* (EFSIM), propuesta por Cheng, Hoang, et al. (2012), cuyo principio es similar a EFNIM, utilizando máquinas de vector de soporte en lugar de la red neuronal. Por último, cabe mencionar las técnicas híbridas que combinan algoritmos de aprendizaje por refuerzo adaptativo con redes neuronales (Obayashi et al., 2010).

2.2.6. Resumen del estado del arte y su aplicación al problema

De lo expuesto en los apartados anteriores, se observa que existe un gran inventario de técnicas que aplican inteligencia artificial en la planificación de proyectos. Dada su diversidad, es preciso acotar el número de ellas que formarán parte este estudio comparativo para la elaboración de cronogramas de construcción de plantas fotovoltaicas. Para ello, es preciso señalar que el modelo correspondiente a este problema será **basado en casos**, debido a que se fundamentará para su elaboración en datos históricos.

Dentro de las posibles técnicas que permiten este tipo de modelado, se emplearán los siguientes **criterios** para seleccionar el grupo que formará parte de este estudio:

1. **Optimalidad:** este es el criterio básico de inclusión. Según Pellerin et al. (2020), las técnicas que devuelven mejores resultados contienen características comunes, entre las que destacan: la existencia de una lista de actividades, el equilibrio entre diversificación e intensificación en la búsqueda de soluciones, la existencia de aleatoriedad en la búsqueda, la existencia de poblaciones que doten de “memoria” al algoritmo de búsqueda y, por último, el autoajuste dinámico. De las 36 técnicas que consiguen mejores resultados, 16 de ellas incorporan **algoritmos evolutivos**. El segundo conjunto de técnicas con mejores resultados, con cuatro representantes, son las basadas en **enjambres de partículas**. Resulta procedente, pues, incorporar en la comparativa a algoritmos que representen a estas dos técnicas. Otra técnica que ha sido probada con grandes resultados en áreas muy diversas, y en concreto, en las relacionadas con la gestión de proyectos, es **ANFIS**, lo que la hace merecedora de formar parte de la comparativa.
2. **Explicabilidad:** se trata de otro factor importante, ya que permite explicar cómo se han alcanzado los resultados. La existencia de capas de lógica borrosa permite añadir este factor al modelo y, por ello, también se incorporará esta técnica en dos modelos de la comparativa. Se comparará su desempeño, asimismo, con el alcanzado por otro modelo básico explicable: los **árboles de decisión**. Su aportación a la comparativa es de interés, a la vista de que el modelo situado en octavo lugar en el ranking de

rendimiento de la revisión de Pellerin et al. (2020) emplea árboles de decisión y algoritmos genéticos. Además, técnicas como enjambre de partículas y evolución diferencial también pueden dotarse de explicabilidad sencilla si se realiza una definición adecuada de la función que representa el modelo.

3. **Variedad:** la comparativa debe realizarse entre técnicas que se encuadren en distintas tipologías. Por ese motivo se emplearán técnicas básicas y técnicas híbridas, todas ellas con características diversas. Dado el buen rendimiento demostrado por los algoritmos evolutivos, se utilizará la técnica de **evolución diferencial**, representando a este tipo de algoritmos en calidad de técnica básica.
4. **Novedad:** de entre las técnicas híbridas expuestas, EFHNN ya ha sido objeto de estudio por parte de (Gil et al., 2021a). En cambio, no se ha localizado en el estado del arte la aplicación de técnicas de **neuroevolución** al problema de la elaboración automática de cronogramas de construcción de plantas fotovoltaicas. La inclusión de estas técnicas permitirá determinar si el grado de complejidad introducido al hibridar lógica difusa con redes neuronales aporta una mejora relevante a la resolución del problema.

Las técnicas finalmente seleccionadas para la comparativa son las que figuran en la Tabla 2, que detalla asimismo los criterios fundamentales que cumplen para su inclusión.

Tabla 2. Criterios de selección de las técnicas empleadas en la comparativa.

Técnica	**Optimalidad**	**Explicabilidad**
Árboles de decisión		●
Enjambre de partículas	●	●
Evolución diferencial	●	●
Neuroevolución	●	
ANFIS	●	●

3. Objetivos y metodología de trabajo

Para este trabajo se planteará un objetivo general, del que derivarán una serie de objetivos específicos. A continuación de la definición de objetivos, se expondrá la metodología que se seguirá para su consecución.

3.1. Objetivo general

El **objetivo** del presente trabajo es realizar el **análisis comparativo** de **cinco técnicas de inteligencia artificial** basadas en **aprendizaje automático** para la **elaboración automática** de **cronogramas de construcción de grandes plantas fotovoltaicas** (> 1 MW). En dicho análisis se obtendrá un conjunto de métricas estándar (como MSE, MAE, R^2), a través de las que se determinará cuál de las técnicas ofrece la mejor solución, así como si estas técnicas superan una determinada "línea base". La fecha en la que se debe disponer de la citada solución es el 21 de julio de 2022.

Esta definición cumple los criterios "SMART" ampliamente aceptados como buena práctica para la fijación de objetivos.

- **S: *Specific*** (específico): comparar cinco técnicas de aprendizaje automático para obtener la mejor solución para elaborar cronogramas de construcción de grandes plantas fotovoltaicas (> 1 MW). Se observará si estas técnicas superan la línea base dada por la media de la variable a predecir.
- **M: *Measurable*** (medible): la comparativa se realizará a través de un conjunto de métricas estándar (como MSE, MAE, R^2). A la hora de elaborar y refinar cada uno de los modelos, se tomará como métrica de referencia un error medio absoluto de 7 días para la duración de las fases e intervalos, y un 20% de desvío máximo con respecto a la duración total de la fase de construcción.
- **A: *Attainable*** (alcanzable): se apoya en un estado del arte desarrollado y cuenta con un conjunto de datos notablemente más amplio que los estudios realizados hasta la fecha.
- **R: *Relevant*** (relevante): tal y como se ha mencionado en el capítulo primero, la mejora de las planificaciones en la construcción de plantas fotovoltaicas permite reducir el principal factor de riesgo en el éxito del cumplimiento de plazo de estos proyectos y, con ello, colaborar a la consecución en plazo de los objetivos de descarbonización, seguridad de suministro y reducción de costes del sistema eléctrico.

- **T: *Time-Related*** (con un tiempo determinado): se establece como fecha fin el 14 de julio de 2022.

3.2. Objetivos específicos

Los objetivos específicos de este trabajo son los siguientes:

1. **Selección de las técnicas** de inteligencia artificial más adecuadas para la comparativa.
2. **Elaboración de los modelos** conceptuales, implementación en software y optimización.
3. **Realización de la comparativa** entre los modelos y determinación de un posible modelo óptimo.
4. **Desarrollo de una herramienta** que permita obtener un cronograma de construcción a partir de unos datos de entrada a través del modelo óptimo.

3.3. Metodología del trabajo

Se seguirá una metodología **empírico-cuantitativa**. Tras haber planteado el problema objeto de estudio y haber revisado la información que existe sobre este, se establece la hipótesis de la investigación, a la que se le puede dar la siguiente forma negativa, a modo de hipótesis nula: "*ninguno de los modelos de la comparativa es capaz de obtener una predicción del cronograma de construcción de plantas fotovoltaicas mejor que las predicciones dadas por el resto de modelos, ni es capaz de superar la línea base establecida por el valor medio de cada variable*". Este estudio tratará de probar la falsedad de dicha hipótesis.

El siguiente punto en el diseño de la investigación es la elaboración conceptual de los modelos y métricas. Una vez completado este paso, es preciso disponer de los datos, para su análisis. Estos datos corresponden a una serie de planificaciones de construcción de plantas fotovoltaicas, elaboradas por expertos en el área.

Una vez analizados los datos, se someterán a contraste las distintas hipótesis. Como resultado del proceso anterior, se elaborarán las correspondientes conclusiones. Se trata en esta fase, pues, de un proceso hipotético-deductivo (Inche et al., 2014).

A modo de listado, para alcanzar los objetivos propuestos, se seguirán los siguientes **pasos**:

1. **Revisión del estado del arte** y trabajos relacionados.
2. **Determinación de los modelos** que formarán parte de la comparativa, en base al análisis del estado del arte.
3. **Elaboración de los distintos modelos y métricas** desde un punto de vista conceptual.
4. **Preparación del conjunto de datos** y **análisis exploratorio** de estos. El conjunto de datos estará formado por 25 cronogramas.
5. **Modelado e implementación** en software a través de Python.
6. **Entrenamiento** de los modelos. Para ello se separará un conjunto de entrenamiento, con 20 cronogramas, y un conjunto de test, con 5 cronogramas. El ajuste de los modelos (obtención de hiperparámetros y entrenamiento) se realizará, en lo posible, con técnicas de validación cruzada, de modo que pueda aprovecharse todo el conjunto de entrenamiento para elaborar los modelos. Asimismo, se procurará independizar la validación cruzada en la búsqueda de hiperparámetros de la validación cruzada en el entrenamiento (*nested cross-validation*).
7. **Obtención de las métricas** de los distintos modelos: MAE, MSE, R^2, tiempo de ejecución.
8. **Validación de los resultados** obtenidos.
9. **Optimización** de los distintos modelos, en función del resultado de la validación.
10. **Determinación de un posible modelo óptimo**, estrictamente mejor que el resto, o. identificación de otras casuísticas. Proponer el modelo óptimo o los modelos óptimos en función de las casuísticas.
11. **Desarrollo de una herramienta** que permita obtener un cronograma de construcción a partir de unos datos de entrada a través del modelo óptimo.
12. **Elaboración de conclusiones.**

Los pasos entre el entrenamiento y la validación -pasos 6 a 8- se repetirán de forma iterativa para optimizar los modelos, hasta alcanzar métricas comparables a la de referencia, indicada en el apartado 3.1. Asimismo, la optimización de los modelos puede conllevar un nuevo análisis exploratorio de los datos, esto es, reiniciar la metodología desde el paso 4.

4. Planteamiento de la comparativa

4.1. Conjunto de datos. Preprocesamiento

Los datos origen se encuentran en un archivo en formato Excel de tipo '*.xlsx*' y proceden de planificaciones elaboradas por expertos en el área de la gestión de proyectos de plantas fotovoltaicas.

			1.-PSFV50 MW_REV02			
			OK			
	PROGRAM SCHEDULE (OUTPUT DATA)		OK			
63	**Construction**		23/01/2020	20/09/2020	241 dias	
64	**Civil works**		23/01/2020	03/08/2020	193 dias	
65	Clear and grub		23/01/2020	28/02/2020	36 dias	0
66	Facilities Installation		23/01/2020	07/02/2020	15 dias	20
67	Earth moving	65	12/02/2020	28/03/2020	45 dias	29
68	Access and internal roads	65	12/03/2020	23/04/2020	42 dias	53
69	Drainage system	65,67	04/05/2020	25/05/2020	21 dias	-6
70	Perimetral fence	62,65,67	28/04/2020	02/06/2020	35 dias	20
71	Fixed structure foundations	65,67,4	18/05/2020	23/07/2020	66 dias	42
72	Inverter stations foundations	65,67	29/06/2020	03/08/2020	35 dias	-31
73	Trenches	65,67,4	29/05/2020	13/07/2020	45 dias	-62
74	MCR	65,67,4	28/03/2020	27/05/2020	60 dias	65
75	**Mechanical Works**		01/06/2020	17/08/2020	77 dias	
76	Marking	65,67,4	01/06/2020	01/07/2020	30 dias	4
77	Ramming /Drilling	65,67,4	05/06/2020	10/07/2020	35 dias	7
78	Structure assembly	49,50,65,67,4	12/06/2020	14/08/2020	63 dias	6
79	Modules installation	,50,51,65,67,4	18/06/2020	17/08/2020	60 dias	-16
80	**Electrical Works**		26/04/2020	31/08/2020	127 dias	
81	Installation of Inverter	56,71	01/06/2020	10/07/2020	39 dias	7
82	Electrical works (LV/DC)	53,54,65	09/06/2020	26/08/2020	78 dias	17
83	Electrical works (MV)	54,65	26/06/2020	31/08/2020	66 dias	-61
85	MCR	53,54,74	26/04/2020	20/06/2020	55 dias	75
84	CCTV installation	4,36,65,61,70	10/07/2020	09/08/2020	30 dias	0
86	Scada installation	60,85	10/07/2020	09/08/2020	30 dias	-11
87	Weather Stations Installation	65,67	29/06/2020	13/08/2020	45 dias	-128
88	**End of PV Plant construction**	64, 75,80	31/08/2020	31/08/2020	0 dias	-191
89	**PV plant substation and switching substation**	58,59	22/02/2020	24/07/2020	153 dias	5
90	**Transmission Line**	59	27/02/2020	04/06/2020	98 Days	143
91	**Precomissioning**	88,89,90	18/07/2020	25/07/2020	7 Days	7
92	**Commissioning**		26/07/2020	20/09/2020	56 Days	
93	**Mechanical and civil works**	88,89,90	26/07/2020	05/08/2020	10 Days	10
94	**Electrical Works**	88,89,90	05/08/2020	15/08/2020	10 Days	10
95	**Completion**	93,94	15/08/2020	15/08/2020	0 Days	-5
96	**Hot commisioning**	95	10/08/2020	20/08/2020	10 Days	5
97	**Check and review works**	96	15/08/2020	25/08/2020	10 Days	5
98	**Commercial Operation Date**	97	20/08/2020	20/08/2020	0 Days	0
99	**Performance Test**	98	20/08/2020	20/09/2020	31 Days	31
100	**Provisional Acceptance**	99	20/09/2020	20/09/2020	0 Days	

Figura 15. *Ejemplo de planificación detallada de una planta.*

Los datos de origen se estructuran por cada una de las plantas. Estos contienen, para cada planta:

1. **Información general** sobre las características de la planta y su planificación, en la que se incluyen datos relativos a las dimensiones de la obra civil, mecánica, eléctrica y la línea de transmisión para su conexión a la red.
2. **Planificación detallada** de la construcción de la planta, desde el inicio de la obra civil hasta la aceptación provisional de la planta, una vez ha entrado en operación. En esta planificación figuran, pues, las fases de obra civil, mecánica, eléctrica, línea de transmisión y puesta en servicio. Se muestra un ejemplo de planificación en la figura 15, que incluye para cada fase:
    - Codificación.
    - Descripción.

- Tareas predecesoras.
- Fecha inicio y fecha fin.
- Duración de la tarea.
- Intervalo entre el inicio de la tarea y el inicio de la siguiente.

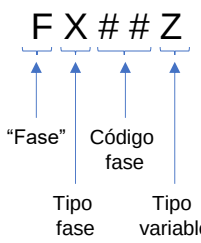
De ese modo, la **información general** se corresponderá con las **variables de entrada** al modelo y la **planificación detallada** corresponderá a las **variables de salida**, esto es, a predecir. La tabla 3 muestra el detalle del conjunto de datos una vez clasificado en variables de entrada y salida, con los códigos numerados entre 65 y 100, que corresponden a las variables de salida.

Tabla 3. Relación de variables de entrada y de salida.

VARIABLES DE ENTRADA				
Variables generales	**Obra civil**	**Obra mecánica**	**Obra eléctrica**	**Línea de transmisión**
Fecha inicio	Área	**Seguidores**	**Cableado**	Distancia
Potencia total	Perímetro	Nº seguidores	Línea BT	Nº apoyos
Tipo planificación	Nº cimientos	Nº paneles/seguidor	Línea MT	Tensión
Ratio MW/días deseado	Carreteras	**Paneles solares**	**Inversores**	
Fecha fin	Zanjas	Nº paneles	Nº inversores	
Duración total	Nº pilares	Potencia unitaria	Potencia unitaria	

VARIABLES DE SALIDA									
Obra civil		**Obra mecánica**		**Obra eléctrica**		**Línea de transmisión**		**Puesta en marcha**	
65	Limpieza del terreno	76	Marcado	81	Instalación de inversores	89	Subestación	91	Pre-puesta en marcha
66	Instalación de infraestructuras	77	Perforación	82	Obra eléctrica BT y CC	90	Línea de transmisión	93	Obra civil y mecánica
67	Movimiento de tierras	78	Montaje de estructuras	83	Obra eléctrica MT			94	Obra eléctrica
68	Accesos y vías interiores	79	Instalación de módulos	85	Sala de medida			95	Fin trabajos
69	Sistema de drenaje			84	Instalación CCTV			96	Puesta en marcha en caliente
70	Cierre perimetral			86	Instalación SCADA			97	Revisión de trabajos
71	Pilares de estructuras fijas			87	Estación meteorológica			98	Fecha de operación comercial
72	Pilares de inversores							99	Test de pruebas
73	Zanjas							100	Aceptación provisional
74	Sala de medida								

Este conjunto de datos se procesa para poder trabajar sobre él en forma tabular. Para ello, se construye un registro por cada planta, en el que se almacenen todas las variables de entrada y dos variables de salida por cada una de las fases: su **duración** y el **intervalo** entre el inicio de la fase en cuestión y el inicio de la siguiente. No es preciso guardar fechas absolutas: el modelo cuenta con un número muy reducido de ejemplos, por lo que en el alcance de este trabajo no se pretende inferir variaciones en el cronograma en función de la época del año en que se desarrollan cada una de las fases de los trabajos.
Cada una de las variables de salida sigue la siguiente notación:



- El **tipo de fase** representa la clase de trabajos en los que se engloba dicha fase: obra civil (C), mecánica (C), eléctrica (E), de la línea de transmisión para la conexión a la red (N) o puesta en marcha (P).

- El **código de fase** consta de dos dígitos y es exactamente el asignado a la fase en la tabla 3.
- Por último, el **tipo de variable** representa si esta corresponde a la duración de la fase (D) o al intervalo entre el inicio de esta fase y la siguiente (I).

Por ejemplo, FC65D representa la duración de la fase de construcción 65 que, según se puede apreciar en la tabla 3, corresponde a la limpieza del terreno. El valor FC65I representa el intervalo desde el inicio de la fase 65 al inicio de la fase 66, 'Instalación de infraestructuras'. Así, las variables "de intervalo" definen el número de días que transcurren hasta el inicio de la **fase posterior**, contados desde el inicio de la fase a la que están asignadas.

4.2. Análisis exploratorio de los datos

En este paso se realiza un análisis estadístico-descriptivo de los datos. Este análisis tiene como objetivo:

- Detectar valores faltantes.
- Localizar errores, datos atípicos, o datos fuera de rango.
- Analizar el comportamiento de las variables y observar correlaciones.
- Reducción de dimensionalidad, para detectar la posibilidad de eliminar alguna característica que aporte ruido y afecte al rendimiento del modelo.

El resultado de este análisis permitirá formular las hipótesis necesarias para modelar el cálculo automático de cronogramas de construcción de plantas fotovoltaicas.

4.2.1. Análisis descriptivo

El conjunto de datos resultante consta de **25 registros** y **85 variables**. De estas 85 variables, **23 son de entrada y 62 de salida**. Se observa, así, que el problema puede padecer de "maldición de la dimensión" (*curse of dimensionality*) dado el elevado número de variables y el reducido número de ejemplos, lo que provoca una elevada distancia entre puntos en el hiperespacio. Así, algunas de las variables que no aportan información añadida deben ser eliminadas, para favorecer la reducción de dimensionalidad. El Anexo II detalla el proceso seguido para la selección de variables de entrada. El conjunto resultante de 13 variables de entrada se muestra en la tabla 4.

Dentro de esta fase de análisis descriptivo se analiza el **histograma** correspondiente a las variables de entrada, a fin de observar si existen ejemplos en todo el rango de entrada de las variables, que permitan al modelo ser entrenado adecuadamente.

Tabla 4. Variables de entrada incorporadas al modelo.

VARIABLES DE ENTRADA				
Variables generales	**Obra civil**	**Obra mecánica**	**Obra eléctrica**	**Línea de transmisión**
Potencia total	Área	**Seguidores**	**Cableado**	Distancia
	Perímetro	Nº seguidores	Línea BT	
	Nº pilares	**Paneles solares**	Línea MT	
	Carreteras	Nº paneles	**Inversores**	
	Zanjas		Nº inversores	
			Potencia unitaria	

En la figura 16 es posible observar cómo aproximadamente la mitad del conjunto de entrada tiene valores muy similares para dos de las variables a priori más importantes: 'Potencia' y 'Área'. Un análisis en mayor detalle revela que 9 de las plantas son prácticamente idénticas en cuanto a potencia, área, perímetro, carreteras, zanjas, pilares y redes de baja y media tensión. Este hecho tendrá consecuencias importantes a la hora de seleccionar los conjuntos de entrenamiento y test.

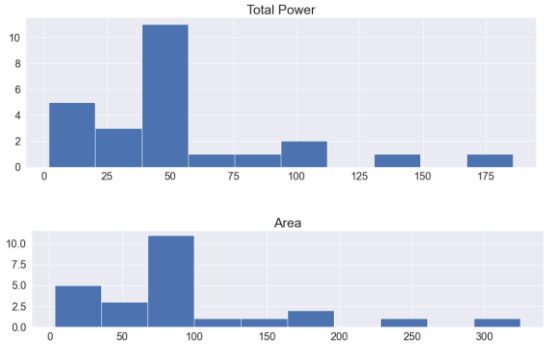


Figura 16. *Histogramas de potencia y área.*

También se procede a analizar los histogramas de las variables de salida, que se muestran en el Anexo II.1. Se observa, en general, una gran dispersión en los valores que pueden adoptar estas variables. En las siguientes secciones se analizará si esta dispersión resulta significativa, a través de la detección de valores anómalos, y se discutirá si es explicable a través de los valores de entrada, gracias al análisis de correlaciones.

4.2.2. Valores anómalos. Depuración de datos.

La estrategia de búsqueda de valores anómalos puede realizarse a través de la identificación de valores más allá del rango definido por la desviación típica de una variable normal (habitualmente 3 veces el valor de la desviación típica σ). En este caso, esta estrategia no parece adecuada a la vista del pequeño número de ejemplos con los que establecer la distribución normal, así como el hecho de que los histogramas no muestran de forma clara que los ejemplos adopten este tipo de distribución. Así, se considerarán **posibles valores anómalos** de una variable x aquellos que no cumplan:

$$\bar{x} - 1{,}5[Q_3(x) - Q_1(x)] < x < \bar{x} + 1{,}5[Q_3(x) - Q_1(x)] \quad (10)$$

, donde $\bar{x}$ es el valor medio de la variable y $Q_3(x) - Q_1(x)$ es el rango intercuartílico entre el primer y el tercer cuartil de la variable.

Del resultado de ese análisis se obtiene una serie de valores anómalos injustificados. Para su tratamiento se plantean las dos siguientes alternativas:

- **Incorporar las variables de entrada** que puedan justificar los valores anómalos. Dado que en el alcance de este estudio no se contempla la posibilidad de incorporar nuevas variables, la alternativa que se utilizará es la que se emplea a continuación.
- **Sustituir los valores anómalos por un valor extrapolado**, a través de los métodos de *k*-vecinos e interpolación lineal, en función de las variables de entrada más representativas y considerando con el conocimiento del dominio.

Si bien el tratamiento de estos valores anómalos ha constituido una parte importante del trabajo, se trata únicamente de un paso previo necesario para alcanzar el núcleo de la contribución. Por ello, el detalle de este apartado se desarrolla en la parte final de este documento, en el Anexo II.

4.2.3. Análisis de correlaciones

Para facilitar el estudio de las correlaciones entre las 13 variables de entrada y las 58 de salida, se ha realizado un estudio por grupos de variables. Cada grupo se ha definido según figura en la tabla 3.

A título de ejemplo, la figura 17 muestra las correlaciones existentes entre dos de los grupos de variables: las variables de entrada generales y las variables de salida de obra civil.

La tabla 5 resume aquellas correlaciones entre grupos de variables de entrada y grupos de variables de salida que pueden considerarse altas, tomando como tales aquellas con valores en torno a 0,7, o superiores.

Tabla 5. Variables de entrada y salida con alta correlación.

Tipo variables entrada	Variables salida
General	Duración obra civil, duración obra eléctrica, duración puesta en servicio
Obra civil y mecánica	Duración obra civil, duración obra eléctrica, duración puesta en servicio
Obra eléctrica	Duración obra civil, duración obra eléctrica, duración puesta en servicio
Línea de transmisión	Duración construcción de la infraestructura de conexión a la red

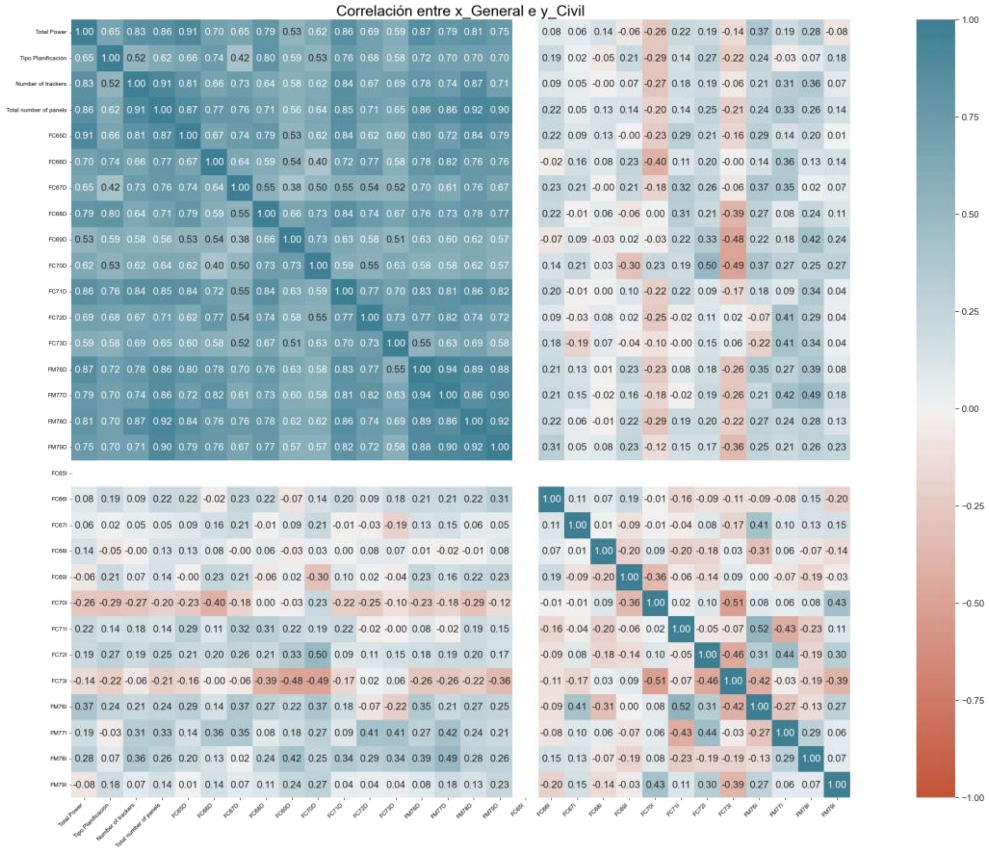


Figura 17. *Correlación entre las variables de entrada generales y de salida de obra civil.*

4.3. Modelado e implementación

En base al análisis de detalle de las correlaciones, al conocimiento del dominio y a un modelado previo, se pueden establecer **dos grupos de variables** que se pueden asumir como **independientes entre sí**:

1. Las **variables relativas a la infraestructura de conexión a la red**:
    - *Entrada*. Se trata de dos variables: 'Distancia' y 'Potencia'. Esta última variable presenta una correlación de 0,51 con la duración de la fase de construcción de las subestaciones. Este valor es inferior al 0,7 fijado como referencia para considerar alta correlación, pero suficiente para tener una influencia sensible en esa variable de salida.
    - *Salida.* Son dos variables: duración de la construcción de las subestaciones (FN89D) y duración de la construcción de la línea de conexión (FN90D).
2. Las **variables relativas a la planta**:
    - *Entrada*. Se trata de las siguientes 12 variables: 'Potencia total', 'Área', 'Perímetro', 'Número de pilares', 'Carreteras', 'Zanjas', 'Número de seguidores',

'Número de paneles', ' Línea BT', 'Línea MT', ' Nº inversores', 'Potencia unitaria' (de cada inversor).

- *Salida*. 53 variables, correspondientes a: duraciones e intervalos de obra civil (FC65 a FC73), obra mecánica (FM76 a FM79), obra eléctrica (FE81 a FE87, con la salvedad de FE87I, tal y como se explicará a continuación) y puesta en marcha (FP91 a FE99).

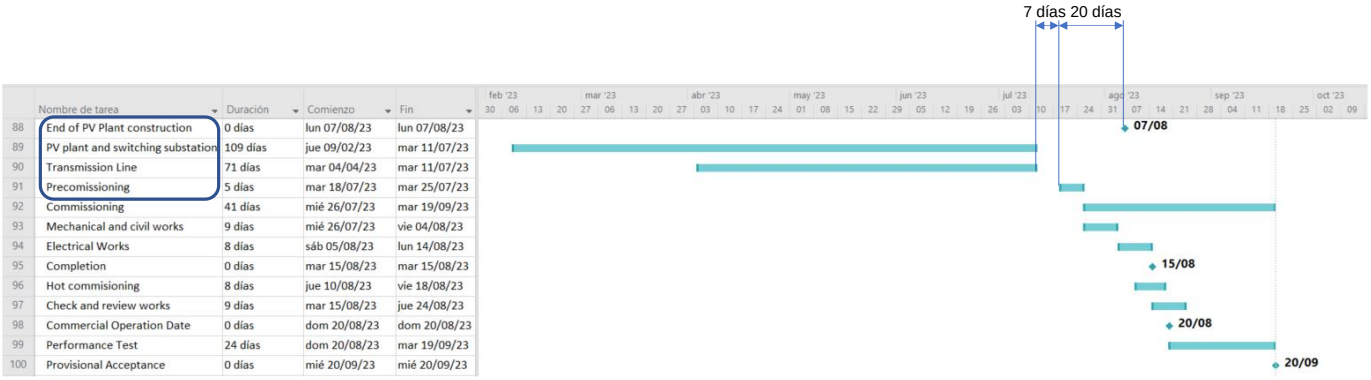
Así, con **cada uno de estos grupos de variables** se elaborará el correspondiente **modelo**: uno para la **conexión con la red** y otro para la **planta.**

Realmente, ambos modelos constituyen **dos problemas independientes**, con la única salvedad de la relación temporal entre ambos, que en el cronograma se encuentra representada por FE87I (que determina la fecha de inicio de construcción de las subestaciones a partir de las fechas de inicio de construcción de la estación meteorológica). Esta variable no puede obtenerse directamente a partir de las variables de entrada, ya que no forman parte del modelo algunas tareas predecesoras necesarias para la construcción de la infraestructura de conexión con la red. Esto es, tanto la construcción de la línea de conexión como las subestaciones dependen de fases previas, como pueden ser la tramitación de los respectivos proyectos específicos, de las que no se dispone información.

Así, a fin de elaborar un cronograma de construcción completo aun sin la información correspondiente a las fases previas que condicionan el inicio de la ejecución de la línea de conexión y las subestaciones, se establece la siguiente relación temporal entre ambos problemas, a partir de la observación de los datos y del conocimiento del dominio:

- La fase de **precomisionado** se inicia 20 días antes de la finalización del montaje de la planta, entendiendo como tal la finalización de la obra civil, eléctrica y mecánica. Esto se debe a que es posible solapar ciertas fases de la puesta en servicio con la fase final del montaje.
- La **línea de transmisión** y las **subestaciones** deben estar finalizadas 7 días antes del inicio del precomisionado. Si bien podría iniciarse el precomisionado sin haberse concluido la línea y las subestaciones, estas dos fases son, en general, las de mayor duración, por lo que es preferible dotar de un margen de seguridad a su finalización.

La figura 18 permite observar estas relaciones sobre un cronograma concreto.



	Nombre de tarea	Duración	Comienzo	Fin
88	End of PV Plant construction	0 días	lun 07/08/23	lun 07/08/23
89	PV plant and switching substation	109 días	jue 09/02/23	mar 11/07/23
90	Transmission Line	71 días	mar 04/04/23	mar 11/07/23
91	Precomissioning	5 días	mar 18/07/23	mar 25/07/23
92	Commissioning	41 días	mié 26/07/23	mar 19/09/23
93	Mechanical and civil works	9 días	mié 26/07/23	vie 04/08/23
94	Electrical Works	8 días	sáb 05/08/23	lun 14/08/23
95	Completion	0 días	mar 15/08/23	mar 15/08/23
96	Hot commisioning	8 días	jue 10/08/23	vie 18/08/23
97	Check and review works	9 días	mar 15/08/23	jue 24/08/23
98	Commercial Operation Date	0 días	dom 20/08/23	dom 20/08/23
99	Performance Test	24 días	dom 20/08/23	mar 19/09/23
100	Provisional Acceptance	0 días	mié 20/09/23	mié 20/09/23

Figura 18. *Relación entre los modelos de construcción de la planta y conexión a la red.*

La arquitectura del modelo se encuentra representada en la figura 19. En ella se muestra cómo el conjunto de variables de entrada -características de la planta y de la conexión con la red- alimenta al modelo de la planta y al modelo de conexión con la red, para obtener las correspondientes salidas: duraciones de fases e intervalos entre fases. Estas, conjuntamente con la fecha de inicio de construcción, permiten generar el cronograma de construcción de la planta y de las instalaciones de conexión con la red, considerando para ello los intervalos definidos para el inicio de la puesta en servicio y para la finalización de la línea de conexión y las subestaciones.

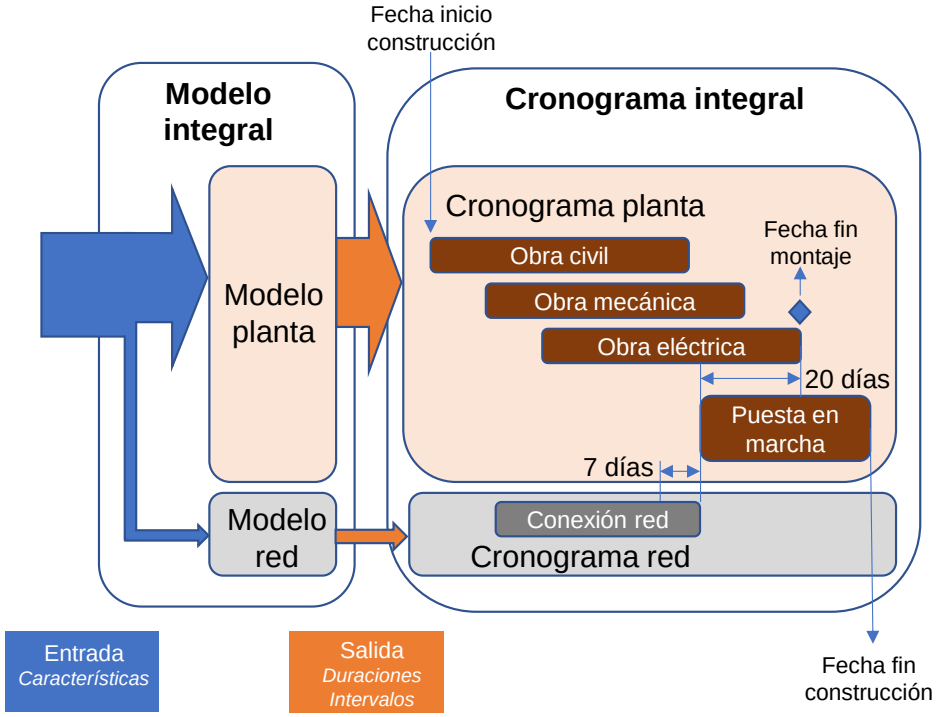


Figura 19. *Arquitectura del modelo integral.*

El **modelado de la infraestructura de conexión a la red** se reduce a un problema con dos variables de entrada y dos variables de salida, entre las que existe una importante linealidad. Esto permite un modelado sencillo, a través de un algoritmo de **regresión lineal**. Con este

planteamiento, se consigue predecir la duración de la construcción de la línea sin errores (se trata, por lo tanto, de un dato calculado). Para la duración de la construcción de las subestaciones, el error medio relativo es inferior a un 13%. Estos resultados implican que este modelo de la infraestructura de conexión a la red se considere **suficientemente bueno**, por lo que **no será preciso compararlo frente a otros modelos de mayor complejidad**. Este modelo se desarrolla con mayor detalle en la parte final del documento, en concreto, en el apartado 1 del Anexo I.

El **núcleo del problema** se encuentra, por lo tanto, en la elaboración del **cronograma de la construcción de la propia planta fotovoltaica**, con las mencionadas 12 variables de entrada y 53 de salida. La complejidad de este problema motiva la realización de la comparativa entre los distintos algoritmos de inteligencia artificial.

El conocimiento del dominio para este problema permite presuponer, en general, una **relación monótona entre las variables de entrada y salida**. Esto es, un cambio en una variable de entrada en un sentido dado producirá cambios en una variable de salida siempre en un mismo sentido. Por ejemplo, una mayor longitud de zanja implicará siempre un mayor tiempo en la fase de construcción de zanjas. En aquellos modelos donde sea necesario especificar dicha relación, de inicio, se tomará una dependencia lineal, al objeto de no partir de grados de complejidad que puedan resultar innecesarios.

4.4. Estrategia de entrenamiento

La estrategia general es la siguiente:

1. **División de los datos** en un conjunto de entrenamiento, del que forman parte el 80% de los datos (20 registros), y un conjunto de test, constituido por el 20% de los datos (5 registros). Los conjuntos de test se generan de forma aleatoria. Sin embargo, tal y como se ha indicado, existe un subconjunto de 9 plantas de características muy similares, por lo que, para garantizar que las métricas del conjunto de test sean representativas, solamente se considerarán aquellos conjuntos de test que contengan menos de un 50% de las plantas de las del citado subconjunto, esto es, un máximo de dos plantas de dicho subconjunto.

2. **Búsqueda de los mejores hiperparámetros**. Debe realizarse sobre el conjunto de entrenamiento. Esta búsqueda se particulariza para cada algoritmo específico:

    - *Árboles de decisión*: se realiza búsqueda en una red de hiperparámetros (*grid search*), con validación cruzada para reducir el sobreajuste y mejorar el rendimiento de las predicciones sobre el conjunto de test.

- *Optimización de enjambre de partículas*: se realiza búsqueda en una red de hiperparámetros, así como búsqueda aleatoria, para tratar de mejorar el rendimiento maximizando la exploración de regiones del conjunto de parámetros.
- *Evolución diferencial*: se realiza búsqueda en una red de hiperparámetros.
- *Neuroevolución*: en este caso, la búsqueda de los hiperparámetros de la red neuronal se realiza a través de un algoritmo genético.
- *ANFIS*: las funciones de pertenencia y las reglas son calculadas por parte del algoritmo, a partir de unas funciones de pertenencia que se definen inicialmente.

3. **Entrenamiento**. Al igual que en el caso anterior, se aplican diferentes estrategias de entrenamiento en función del algoritmo. En árboles de decisión, optimización de enjambre de partículas, evolución diferencial y ANFIS se realiza un entrenamiento independiente para cada una de las variables de salida. En neuroevolución, la red neuronal se construye de tal modo que permite generar simultáneamente todas las salidas con un único entrenamiento.

4.5. Criterios de éxito y métricas. Validación

Una vez generado el modelo a partir del entrenamiento, se mide su bondad contra los datos de test. A partir de estos, con cada modelo es posible obtener las predicciones de duraciones e intervalos de fechas que construyen el cronograma.

Con el error de la predicción de cada una de las variables de salida es posible construir una matriz, con número de filas igual al número de registros de test, y número de columnas igual al número de variables de salida. A partir de las predicciones, los valores reales y la matriz de error es posible obtener las métricas de referencia: **MAE** -media del error absoluto-, **MSE** -media del error cuadrático- y $\boldsymbol{R^2}$ -coeficiente de determinación-, que se definen como sigue (Scikit-learn, s. f.-b):

$$MAE(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \qquad MSE(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{11}$$

, donde n es el número total de muestras, $\hat{y}_i$ es el valor predicho para la *i*-ésima muestra, y_i es el valor real de la *i*-ésima muestra, $\hat{y}$ el vector que representa los valores predichos, y el vector que representa los valores reales e $\bar{y}$ el valor promedio del valor real de las muestras:

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{12}$$

A la hora de evaluar el R^2 total, se promediará por la varianza de cada variable, para así compensar su asimetría, dado que los valores de esta métrica van desde $-\infty$ hasta 1, siendo tanto más probables los valores menores cuanto más próxima esté la distribución a la media, esto es, cuanto menor varianza exista.

Para evaluar la bondad de los distintos modelos, se definirá una **línea base**, en la que la predicción vendrá dada por el valor medio de cada una de las variables de salida, y sobre la que se obtendrán las mencionadas métricas.

También se considerará en la comparativa el **tiempo** de entrenamiento de cada uno de los modelos, dado que existen diferencias relevantes entre ellos. El procesador con el que han sido entrenados es una CPU Intel Core i5-7300U, a 2.60GHz, y una memoria RAM de 8 GB.

Para la **validación de cada modelo**, se realizará una batería de ocho test, de tal modo que todos los registros existentes formen parte de alguno de los conjuntos de test. Con ello se consigue llevar a cabo una validación cruzada, cumpliendo asimismo la restricción impuesta para evitar conjuntos de test que contengan más de la mitad de registros muy similares. Las métricas finales que serán objeto de la comparativa se obtendrán como promedio de las métricas obtenidas en cada uno de estos test.

5. Desarrollo de la comparativa

La comparativa se realizará en el siguiente orden:

1. Árboles de decisión: para esta técnica se empleará tanto un árbol aislado como dos algoritmos *ensemble*: *XGBoost* y *Random forest*.
2. Optimización de enjambre de partículas.
3. Evolución diferencial.
4. Neuroevolución.
5. ANFIS.

En cada técnica se presenta su implementación, así como los resultados obtenidos tras su evaluación en los ocho test que cubren todo el conjunto de datos, con una comparativa con la

línea base. En el último apartado, se realiza una comparativa directa entre los resultados obtenidos por los distintos métodos.

5.1. Árboles de decisión

La implementación de este algoritmo se ha realizado empleando un árbol de regresión para cada una de las variables de salida. Cada árbol se alimenta con todas las variables de entrada, según se muestra en la figura 20. Esta estrategia dota de suficiente flexibilidad e independencia entre variables de salida a la regresión, lo cual es relevante dado el reducido tamaño del conjunto de datos (en el Anexo I se discute la posibilidad de utilizar un único árbol "multivariable" y se observa el bajo rendimiento dado en las pruebas realizadas).

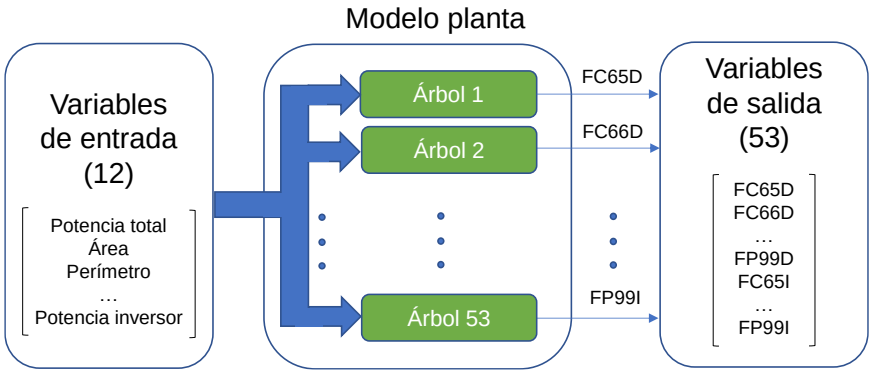


Figura 20. *Modelo basado en árboles de decisión*.

En primer lugar, se realiza una búsqueda en red (*grid search*) de los mejores hiperparámetros de cada árbol, utilizando para ello el conjunto de entrenamiento. Se busca la combinación óptima de parámetros (profundidad máxima y número mínimo de elementos por hoja) a través de la técnica de *k-fold cross validation*, con *k*= 5. A fin de evitar sobreajuste, se limita la profundidad máxima de cada árbol a 6 niveles y un número de ejemplos por hoja de 2. Hay que considerar que, con 3 ejemplos por hoja, el árbol permitiría, a lo sumo, 6 divisiones en hojas a partir de los 20 ejemplos del conjunto de test.

Con los hiperparámetros obtenidos para cada árbol, se realiza el entrenamiento de cada uno de ellos por separado, con la implementación de Scikit-learn (n.d.-a).

Una vez entrenado el modelo, se evalúa su bondad en el conjunto de entrenamiento y en el conjunto de test, con las métricas establecidas: MAE, MSE y R^2. Asimismo, se evalúa el tiempo de cálculo.

La figura 21 muestra un ejemplo de uno de estos árboles, en concreto, para la variable FC65D, que representa la duración de la fase de limpieza del terreno.

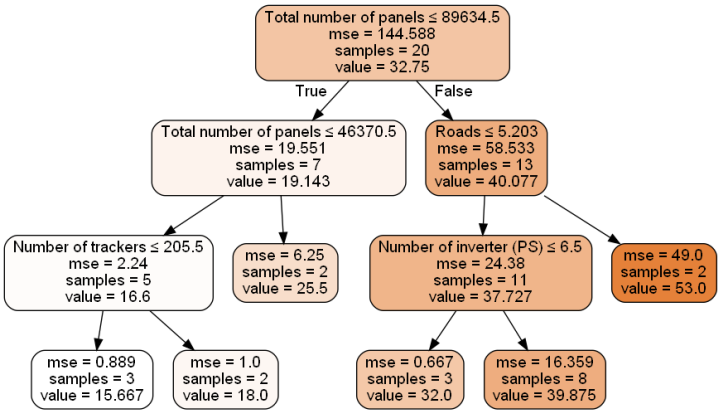


Figura 21. *Árbol de regresión para la duración de la fase de limpieza del terreno.*

Asimismo, los resultados de estos algoritmos son explicables de forma muy sencilla. En la figura 22 se representa, a modo de ejemplo, cuáles son las variables de entrada más importantes para una variable de salida concreta, así como una cuantificación de dicha importancia. Mediante esta herramienta ha sido posible confirmar o descartar las hipótesis establecidas en la fase de análisis exploratorio de los datos. Se observa, además, que la calidad de la predicción empeora para valores extremos de las variables de salida.

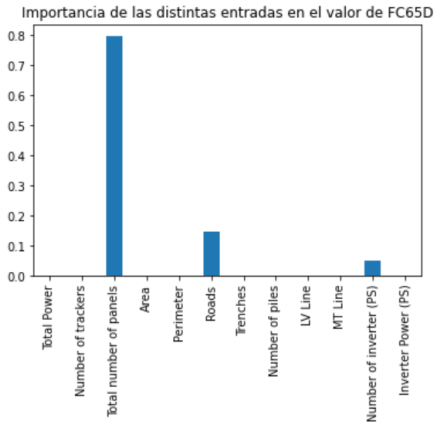


Figura 22. *Representación de la importancia de las variables de entrada.*

El **rendimiento promedio** alcanzado en los ocho test con esta técnica se muestra en la tabla 6. Se observa cómo se produce una mejora clara con respecto a la línea base en todas las métricas. La mejora obtenida se puede comprender de forma sencilla a través del error medio absoluto (MAE) alcanzado, de 5,57 días en test, frente a los 9,17 de la línea base. También se produce una mejora clara en MSE y R^2. En cuanto al coste computacional de obtención del modelo, es pequeño, con un promedio de 25,93 segundos para la ejecución del algoritmo.

Se puede observar la existencia de sobreajuste a los datos de entrenamiento. El MAE en entrenamiento es de solamente 2,38 días y el coeficiente de determinación R^2 alcanza, en entrenamiento, un valor muy elevado (0,8748, frente a 0,3650 en test).

Tabla 6. Rendimiento del árbol de regresión.

		Línea base	Árbol regresión
MAE entrenamiento	días	8,48	2,38
MSE entrenamiento	días^2	259,12	32,89
R^2 entrenamiento	-	0,0000	0,8748
MAE test	días	9,17	5,57
MSE test	días^2	275,86	129,29
R^2 test	-	-0,4201	0,3650
Tiempo	s	0	26

Para tratar de mejorar este rendimiento, se realizan pruebas con *ensembles*: un modelo basado en *boosting -XGBoost-* y otro basado en *bagging -Random Forest-*. La contrapartida de la mejora de rendimiento prevista con estos modelos será la pérdida de una representación sencilla, ya que el modelo estará ahora constituido por un conjunto de árboles, lo que le resta explicabilidad.

XGBoost

Tal y como se ha mencionado en el apartado 2.2.4, esta técnica consiste en elaborar un modelo formado por sucesivos árboles de regresión, dando en cada una de las sucesivas iteraciones un mayor peso a aquellos ejemplos que hayan sido incorrectamente clasificados con anterioridad. El modelo resultante es la suma de todos los árboles calculados. Esta estrategia permite resolver al menos conceptualmente los problemas detectados con los valores extremos de las variables de salida. Para poder evaluar la bondad de las predicciones durante el entrenamiento, y así realizar el reajuste, es necesario definir un conjunto de validación dentro del propio conjunto de entrenamiento. Se ha tomado para este conjunto de validación un total de 5 registros, esto es, un 25% de los datos de entrenamiento. La implementación utilizada es la de XGBoost (n.d.).

Con la salvedad fundamental de la necesidad de disponer de un conjunto de validación, el proceso de entrenamiento y test es análogo al descrito para árboles de decisión. En este caso, *XGBoost* no permite definir el mínimo número de ejemplos por hoja, si bien el control de la profundidad del árbol -limitado a 6 niveles- se muestra suficiente en la práctica.

Los **resultados** obtenidos por este modelo (tabla 7) mejoran levemente en todas las métricas a los de un árbol de regresión aislado (MAE de 5,38 días frente a 5,57). El sobreajuste es también ligeramente menor, como se puede deducir del hecho de un mayor MSE en los datos de entrenamiento, lo que implica que el modelo funciona mejor aun permitiendo desviaciones extremas de mayor entidad sobre los datos de entrenamiento.

Tabla 7. Rendimiento de *XGBoost.*

		Línea base	XGBoost
MAE entrenamiento	días	8,48	1,80
MSE entrenamiento	días^2	259,12	46,74
R^2 entrenamiento	-	0,0000	0,8166
MAE test	días	9,17	5,38
MSE test	días^2	275,86	117,45
R^2 test	-	-0,4201	0,4354
Tiempo	s	0	157

Random forest

Tal y como se ha introducido en el apartado 2.2.4, este algoritmo se basa en la generación un conjunto de árboles de decisión. Para cada uno de ellos se toma únicamente una parte de las variables de entrada (habitualmente $\sqrt{p}$, siendo p el número de variables de entrada) y una parte del conjunto de entrenamiento (habitualmente 2/3). El resultado de la predicción se toma como el promedio del resultado de cada uno de los árboles individuales. Con ello se pretende evitar el sobreajuste que suele existir al modelar con un único árbol, a costa de perder una explicabilidad sencilla, al encontrarse "diluida" entre todos los modelos.

Tras diversas pruebas, el número óptimo de árboles se ha establecido en 500. Con un mayor número de árboles, se produce un sobreajuste excesivo, sin mejorar las métricas de test. Con un número menor de árboles, aparece un problema de inestabilidad en las soluciones, ya que la variabilidad en su calidad aumenta notablemente, aunque el rendimiento medio solo decae ligeramente. El segundo hiperparámetro que se fija es la profundidad máxima de cada árbol que, tras un proceso análogo al realizado para definir el número de árboles, se ha establecido en 6 niveles.

Con ello es suficiente para generar el modelo a partir del conjunto de entrenamiento. La implementación en la que se basa es, al igual que para el árbol de regresión, la de Scikit-learn.

Los **resultados** obtenidos se muestran en la tabla 8. Se produce una ligera mejora frente a XGBoost en MAE (5,26 frente a 5,38), si bien tanto MSE como R^2 se degradan. Esto supone que el modelo basado en *Random forest*, si bien tiene un buen comportamiento medio, es más susceptible de dar soluciones que difieren de forma extrema -por lo cual aumenta su MSE- y es peor a la hora de explicar las variaciones -dado su menor coeficiente de determinación-. Se observa que los modelos resultantes de esta técnica se ajustan de una forma muy precisa a los datos de entrenamiento, lo que en la práctica se traduce en un sobreajuste que impide una inferencia eficiente sobre los datos de test.

Tabla 8. Rendimiento de *Random forest*.

		Línea base	Random forest
MAE entrenamiento	días	8,48	0,06
MSE entrenamiento	días^2	259,12	0,96
R^2 entrenamiento	-	0,0000	0,9963
MAE test	días	9,17	5,26
MSE test	días^2	275,86	137,65
R^2 test	-	-0,4201	0,3225
Tiempo	s	0	47

Como resumen, las técnicas basadas en árboles de decisión asignan un valor promedio o mediano a las variables de salida en función de determinados rangos en las variables de entrada. Por ello, producirán un sesgo sistemático en la predicción. Este sesgo es tanto más importante cuanto menor sea el tamaño del conjunto de datos con el que se ha elaborado el modelo. Puede observarse que en este conjunto de datos existe una dispersión relevante de las variables de salida frente a las variables de entrada identificadas como de mayor importancia, según se muestra en el ejemplo de la figura 23 (el Anexo II también contiene un ejemplo similar).

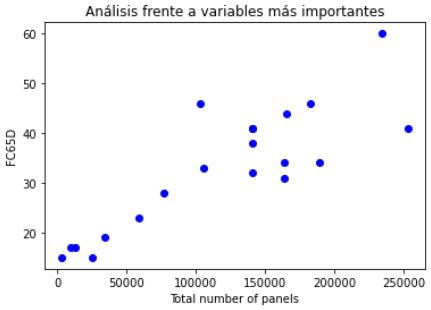


Figura 23. *Ejemplo de dispersión en una variable de salida.*

Así, las técnicas basadas en árboles de decisión no permiten modelar adecuadamente algunas de las variables de salida, especialmente los intervalos entre inicio de fechas. En estas, las muestras se encuentran muy separadas en algunas de las variables de salida del hiperespacio. La división en un número elevado de hojas produce sobreajuste y, la alternativa opuesta, con un pequeño número de hojas, asigna un mismo valor a un conjunto de valores en un mismo rango de entradas relativamente grande.

5.2. Optimización de enjambre de partículas

La respuesta conceptual al problema planteado anteriormente se encuentra tanto en esta como en las siguientes técnicas de optimización. Todas ellas permiten obtener variables de

salida continuas. Esto supone que las variables de salida se obtengan a partir de las de entrada a través de funciones paramétricas. El problema será, pues, la obtención de dichos parámetros. Su cálculo se resuelve de modo más eficiente acotando el hiperespacio de búsqueda. Ello se consigue a través de la **normalización** de las variables de entrada y salida dentro de un entorno próximo al origen (aproximadamente entre -1 y 1), del siguiente modo:

- *Variables de entrada*: se escalan dividiendo por el valor máximo correspondiente.
- *Variables de salida (días)*: se divide su valor entre 100. De este modo, quedan aproximadamente en el entorno de (-1,1). Este escalado permite que las métricas MSE y MAE sean comparables de forma sencilla con las obtenidas con los árboles de regresión anteriores y permite dar sentido físico a estas métricas, al ser interpretables en días.

El modelo empleado para la optimización de enjambre de partículas utiliza, de modo análogo al modelo de árboles, un enjambre por cada una de las variables de salida. Cada enjambre se alimenta con todas las variables de entrada, según muestra la figura 24.

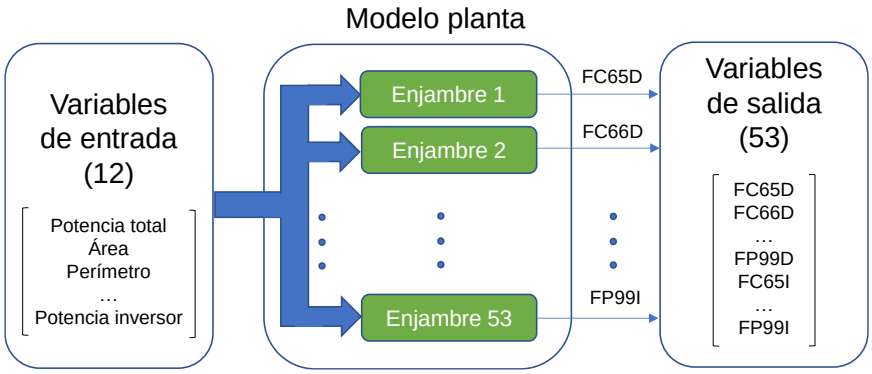


Figura 24. *Modelo basado en optimización de enjambre de partículas.*

Cada una de las partículas de un enjambre busca la posición óptima en el hiperespacio. Las dimensiones de la posición de una partícula representan los parámetros que relacionan a la correspondiente variable de salida con las variables de entrada.

Tal y como se ha indicado en el apartado 4.3, “Modelado e implementación”, inicialmente se han supuesto dependencias lineales entre las variables de entrada y las variables de salida. Así, la estimación de cada variable de salida y_i puede representarse en función de las n variables de entrada x_j como

$$\hat{y}_i = \sum_{j=1}^{n} (w_j x_j) + b_i \qquad (13)$$

, siendo w_j, b_i los parámetros a calcular, esto es, los parámetros que definen la posición de la partícula, un total de 13 en este problema: un parámetro lineal w_j por cada una de las 12 variables de entrada más un término independiente b_i.

La función de coste, que será el objetivo a minimizar, es la suma de los cuadrados de los errores entre el valor calculado de la variable de salida $\hat{y}_i$ y su valor real y_i, para las m muestras del conjunto de entrenamiento:

$$Coste\ (y_i, \hat{y}_i) = \frac{1}{m}\sum_{i=1}^{\mathrm{m}} \|y_i - \hat{y}_i\|^2 \tag{14}$$

Esta función de coste se evalúa para cada una de las partículas del enjambre. El desplazamiento de las partículas se realiza, tal y como se ha expuesto en el apartado 2.2.4, en función de su velocidad. Esta se calcula considerando:

- El peso de la mejor solución histórica para una partícula dada (c_1).
- El peso de la mejor solución histórica para todo el conjunto de partículas (c_2).
- El peso de la velocidad anterior de la partícula (w).

Los mejores resultados de la implementación se han obtenido con un total de 1000 partículas para el cálculo de cada una de las variables. De entre las distintas posiciones iniciales del enjambre, han devuelto resultados similares una aleatoria en el primer cuadrante y una aleatoria centrada en el origen. Los enjambres que se inicializan de forma homogéneamente distribuida han devuelto peores resultados.

Para establecer los hiperparámetros c_1, c_2 y w se han seguido dos estrategias:

1. Búsqueda aleatoria.
2. Búsqueda en red (*grid search*).

Se ha observado que la búsqueda en red es la que consigue mejores resultados. Esta búsqueda en red se realiza en una región acotada de los hiperparámetros, en torno a $c_1 = 0{,}8$, $c_2 = 0{,}3$ y $w = 0{,}8$, y se lleva a cabo para cada una de las variables de salida.

El algoritmo converge de forma muy rápida, tal y como se aprecia en la figura 25. Su implementación se basa en Miranda (2017). La figura 11 (apartado 2.2.4) muestra la trayectoria seguida por el enjambre de partículas, restringida su representación a únicamente dos variables de entrada, para poder visibilizarla en un plano.

Sin embargo, los **resultados** devueltos, que se muestran en la tabla 9, no son buenos. En el caso de MAE (9,80 días), es incluso inferior a la dada por la línea base (9,17 días). El bajo rendimiento de este algoritmo se debe a un problema conocido: su tendencia a converger a

soluciones subóptimas. Si bien el promedio de las predicciones no es bueno, sí produce una mejora sobre la línea base en cuanto a los valores extremos (véase el MSE) y en su capacidad de explicar las variaciones en la salida (dada por R^2).

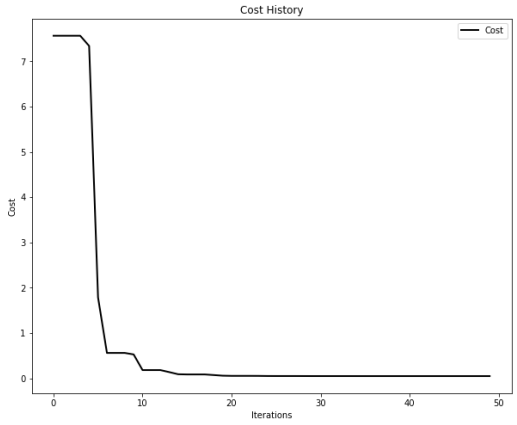


Figura 25. *Convergencia de la función de coste del enjambre de partículas.*

Tabla 9. Rendimiento de optimización de enjambre de partículas.

		Línea base	PSO
MAE entrenamiento	días	8,48	7,71
MSE entrenamiento	días^2	259,12	105,28
R^2 entrenamiento	-	0,0000	0,6150
MAE test	días	9,17	9,80
MSE test	días^2	275,86	193,34
R^2 test	-	-0,4201	0,0383
Tiempo	s	0	572

Una de las soluciones propuestas para resolver los problemas de convergencia subóptima es el empleo de técnicas de optimización local, en las que se definen una serie de "subenjambres". Con ello se evita que un único enjambre caiga en un mínimo local. Las pruebas realizadas con este algoritmo local han obtenido un rendimiento muy bajo y se describen aparte, en el Anexo III.2.

5.3. Evolución diferencial

En este método también se obtendrá un submodelo independiente por cada una de las variables de salida. Cada submodelo consiste en una población de vectores que se alimenta con todas las variables de entrada, según muestra la figura 26.

El algoritmo requiere definir los siguientes hiperparámetros básicos:

- Tamaño de la población.
- Grado de mutación.

- Constante de recombinación.

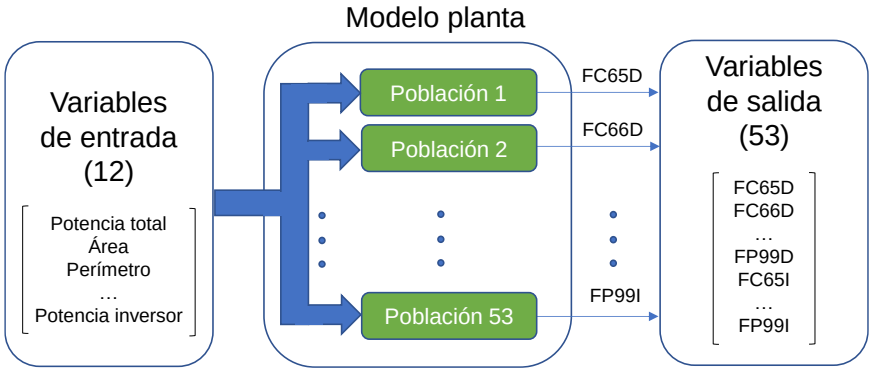


Figura 26. *Modelo basado en evolución diferencial*.

Este algoritmo puede plantearse a través de distintas estrategias. La finalmente seleccionada es la conocida como *'best1bin'*, que generalmente suele devolver buenos resultados. En ella, la diferencia entre los dos vectores seleccionados aleatoriamente se utiliza para mutar el mejor miembro de la población. Este vector mutado será el que transfiera algunos de sus parámetros al vector a evolucionar, en función de la constante de recombinación (ver 2.2.4). Las posibilidades de encontrar un óptimo global son mayores cuanto mayor sea el tamaño de la población y el grado de mutación y cuanto menor sea la constante de recombinación. Esto tiene el efecto de ampliar el radio de búsqueda, pero ralentizar la convergencia (SciPy, 2020).

La función de coste, a minimizar, es análoga a la descrita en el algoritmo de optimización de enjambre de partículas. La implementación se realiza sobre la base de SciPy.

Los mejores **resultados** se han obtenido con un parámetro de población de 25 vectores, un grado de mutación entre 0,5 y 1 y un grado de recombinación de 0,5. Este algoritmo requiere acotar el espacio de búsqueda de los parámetros a través de un conjunto de límites, que se han definido como (-2, 2) para cada parámetro -una vez normalizadas las variables-.

Los mejores resultados devueltos por este algoritmo son muy próximos al valor medio de cada variable y, por tanto, las métricas son muy similares a las dadas por la línea base, tal y como se puede comprobar en la tabla 10. Así, este modelo **no aporta información adicional significativa sobre la dada por el valor medio**. En cuanto al tiempo de ejecución (212 s de promedio), no es excesivo, si bien resulta inútil al no aportar valor alguno sobre la línea base.

Para tratar de mejorar su rendimiento, se han explorado regiones más amplias del hiperespacio, aumentando la población y la mutación y reduciendo la recombinación, así como incrementando el valor de los límites. Estas pruebas han devuelto rendimientos inferiores a los dados por la línea base. También se ha empleado la estrategia '*rand1bin*', que utiliza un vector aleatorio en lugar del "mejor vector", sin superar los resultados dados por '*best1bin*'.

Tabla 10. Rendimiento de evolución diferencial.

		Línea base	Evolución diferencial
MAE entrenamiento	días	8,48	8,48
MSE entrenamiento	días^2	259,12	259,13
R^2 entrenamiento	-	0,0000	0,0000
MAE test	días	9,17	9,19
MSE test	días^2	275,86	276,16
R^2 test	-	-0,4201	-0,4216
Tiempo	s	0	213

En definitiva, este algoritmo adolece en este caso del problema ya descrito en el estado del arte, consistente en el estancamiento en soluciones inferiores a las calificables como subóptimas.

5.4. Neuroevolución

La implementación realizada en esta técnica busca definir la topología de la red neuronal a través del algoritmo genético. Se parte de una capa de entrada con un número de neuronas igual al número total de variables de entrada (12) y una capa de salida con un número de neuronas igual al número total de variables de salida (53).

Las aproximaciones basadas en NEAT resultan interesantes desde un punto de vista teórico, pero su implementación práctica no resulta atractiva, dada la complejidad de las topologías resultantes, en las que eventualmente cada neurona puede estar conectada con cualquier otra. Así, se ha optado por una implementación basada en Keras (Harvey, 2017). En ella, cada individuo de la población es una red neuronal alimentada hacia delante, completamente conectada y con varias capas ocultas. Cada individuo se encuentra definido por los siguientes parámetros o “genes”:

- *Número de neuronas por capa oculta*. Se han conseguido buenos resultados cuando este gen puede adoptar 8 valores predefinidos (en una lista de valores comprendidos entre 120 y 370 neuronas).
- *Número de capas ocultas*. Tras las pruebas realizadas, se consigue un buen rendimiento cuando el valor de este gen puede evolucionar entre 2, 3 ó 4 capas.
- *Función de activación*: ReLU, eLU, sigmoide o tangente hiperbólica.
- *Optimizador*: *Adam, SGD, Adagrad, Adadelta, Nadam, RMSprop, Adamax*.

Como hiperparámetros del algoritmo genético, deben definirse:

- *Número de generaciones*: el algoritmo se aproxima a su asíntota de rendimiento a las 20 generaciones.
- *Población de cada generación*: se obtiene un buen resultado con 20 redes por generación.
- *Ratio de la población* que debe permanecer tras cada generación. Las pruebas dan un buen resultado para un valor de 0,4.
- *Selección aleatoria*: probabilidad de que una red no seleccionada por su *fitness* continúe en la población. Se emplea un valor de 0,1.
- *Grado de mutación*: representa la probabilidad de que una red sufra alguna mutación. Se emplea un valor de 0,2.

Una vez llevada a cabo la selección por parte del algoritmo genético, el modelo obtenido será el individuo (red) con mejor *fitness* de la última generación. La función de *fitness* debe ser maximizada. Para ello, esta función ha sido definida como el MSE cambiado de signo, de modo que cuanto menor sea el MSE, mayor será el *fitness*.

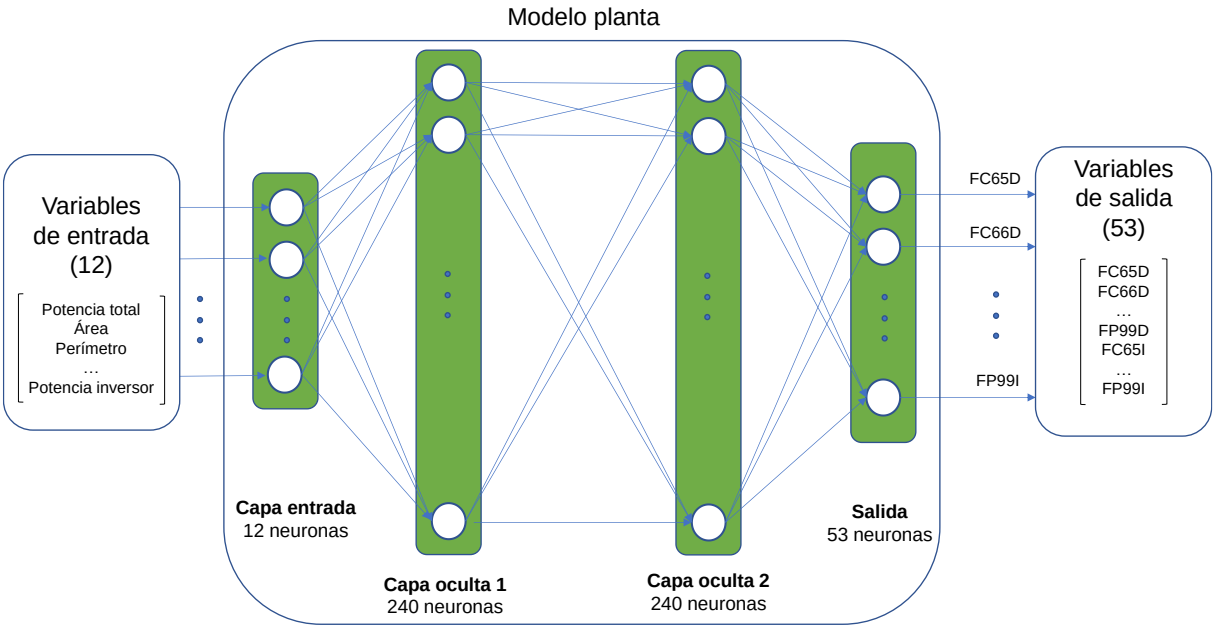


Figura 27. *Arquitectura del modelo obtenido como resultado de neuroevolución.*

La figura 27 representa la arquitectura del modelo obtenido por neuroevolución para una de las soluciones de los test. La red neuronal consta de dos capas ocultas y 240 neuronas por capa. La función de activación en las capas ocultas que ha devuelto mejores resultados ha sido, en la práctica totalidad de los casos, ReLU. Cada capa, excepto la de salida, lleva a continuación un *dropout* de 0,2, para regularizar el comportamiento de la red. Se han realizado pruebas con capas *batch normalization* antes de la función de activación, sin que se hayan

producido mejoras en los resultados. Por último, la capa de salida también incorpora función de activación ReLU. Dado que esta función solamente devuelve valores positivos, ha sido preciso escalar las variables de salida de modo que todos sus valores sean positivos.

El entrenamiento se ejecuta con 100 *epoch* con condición de parada temprana en caso de que tras 10 iteraciones no se produzcan mejoras en el rendimiento de la red.

Los parámetros que habitualmente devuelven mejores resultados son 2 ó 3 capas y un número de neuronas por capa en torno a 240.

Los **resultados** obtenidos se presentan en la tabla 11. Puede observarse que el rendimiento obtenido es muy superior al establecido por la línea base. La mejora sustancial de este método no se observa en el MAE (5,36 días), sino en la reducción del valor del MSE (75,41), lo que significa un mejor comportamiento ante posibles errores grandes, y un incremento relevante en el coeficiente de determinación R^2, muy próximo a 0,6. El tiempo de ejecución promedio es de 703 segundos. Puede parecer elevado pero, tal y como se ha indicado en el apartado 2.2.5, esta estrategia reduce sobre un 80% el tiempo empleado por algoritmos de fuerza bruta.

Tabla 11. Rendimiento de neuroevolución.

		Línea base	Neuro evolución
MAE entrenamiento	días	8,48	4,34
MSE entrenamiento	días^2	259,12	49,41
R^2 entrenamiento	-	0,0000	0,8079
MAE test	días	9,17	5,36
MSE test	días^2	275,86	75,41
R^2 test	-	-0,4201	0,5909
Tiempo	s	0	704

5.5. ANFIS

La implementación de este algoritmo se basa en Meggs (2020). En este algoritmo la necesidad de recursos de computación crece exponencialmente al incrementar el número de variables de entrada que, en la práctica, se ven limitadas a un número de cinco por cada modelo.

Esta restricción implica la necesidad de reformular el problema, para tratar de expresar cada una de las variables de salida a través de un máximo de cinco variables de entrada. En base al conocimiento del dominio, el análisis de los datos y la realización de diversas pruebas, se ha configurado la relación entre variables de entrada y variables de salida según se representa en la tabla 12 (la codificación de las variables de salida es la dada por la tabla 3).

Tabla 12. Modelado para ANFIS: grupos de variables.

	Variables de salida	Variables de entrada
Obra civil	FC65 a FC73	*Área, Carreteras, Zanjas, Nº cimientos, Nº inversores*
Obra mecánica	FM76 a FM79	*Potencia total, Nº seguidores, Nº paneles, Nº pilares, Nº inversores*
Obra eléctrica	FE81 a FE87	*Línea BT, Línea MT, Nº inversores, Potencia inversor*
Puesta en marcha	FE91 a FP99	*Potencia total, Área, Nº paneles, Nº inversores*

Con ello, es posible crear un conjunto de modelos ANFIS para cada grupo de variables de la planta, tal y como se representa en la figura 28. Como también se puede observar en la tabla 12, algunas de las variables de entrada son comunes a más de un modelo.

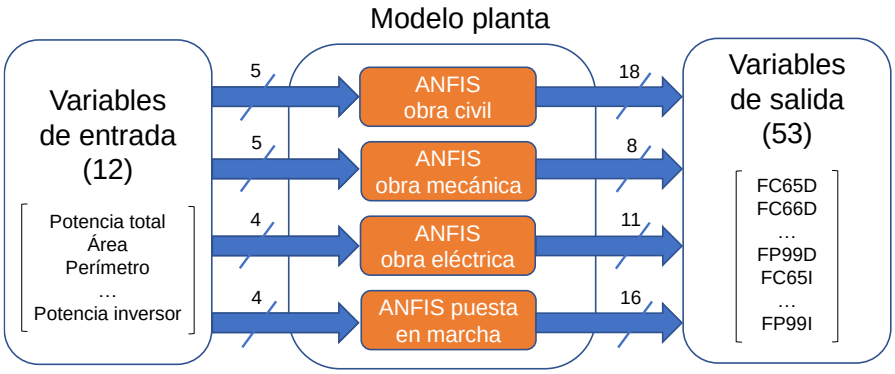


Figura 28. *Modelo global de la planta basado en ANFIS.*

La figura 29 detalla cómo se realiza la descomposición de cada conjunto de modelos ANFIS, utilizando como ejemplo el grupo de variables de obra eléctrica. Se observa cómo para este caso existen 11 modelos ANFIS (uno por variable de salida), cada uno de ellos alimentado por las mismas 4 variables de entrada.

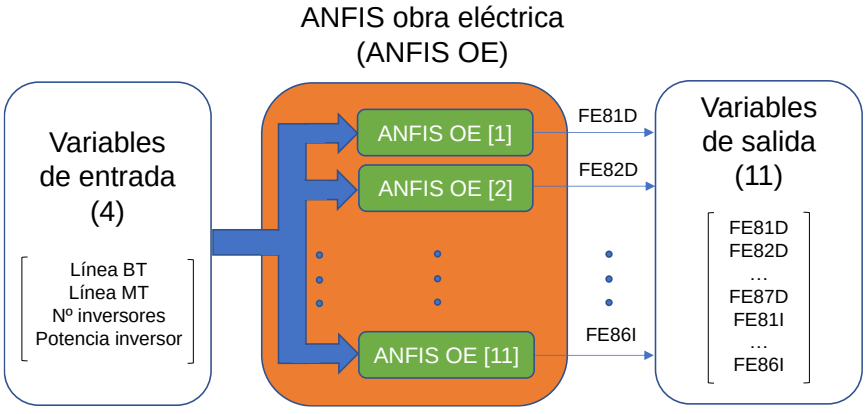


Figura 29. *Modelado ANFIS del grupo de variables de obra eléctrica.*

A su vez, cada uno de estos modelos ANFIS implementa un sistema neurodifuso, representado en la figura 30 para la variable FE81D (duración de la fase de tendido de la línea de BT). Para ello, ANFIS requiere de la definición inicial de un conjunto de funciones de

pertenencia borrosas $\mu(x)$ para cada una de las variables de entrada x. Tras la realización de diversas pruebas, se ha encontrado que la definición que minimiza el sobreajuste es la que se muestra en la figura 31: tres funciones de pertenencia gaussianas por variable, con σ=0,45 y μ =0, 0,5 y 1. Cada una de ellas representa el grado de pertenencia a un nivel "bajo", "medio" o "alto" de x, respectivamente. Nótese que representaciones iniciales a priori más naturales (como la dada por σ=0,25, en la que el grado de pertenencia medio tiende a 0 para valores próximos a 0 ó a 1) han conducido a peores resultados, con sobreajuste. Estas 3 funciones de pertenencia generan 3^n reglas del tipo Takagi-Sugeno, siendo n el número de variables de entrada que modelan cada variable de salida. Así, una variable de salida como FE81D, que depende de 4 variables de entrada, estará definida por 81 reglas borrosas. Dichas reglas, una vez normalizadas, ponderan las funciones consecuentes Takagi-Sugeno. Cada uno de ellas es del tipo $f_i = p_1x_1 + p_2x_2 + \ \dots \ + p_nx_n + r_i$, siendo n el número de variables de entrada. Así, una variable de salida que dependa, como en el ejemplo, de 4 variables de entrada, dispondrá de 81 reglas y el consecuente de cada una de estas reglas se encontrará definido, a su vez, por 5 parámetros ($p_1, \ \dots, p_4, \ r_i$), esto es, un total de 405 parámetros.

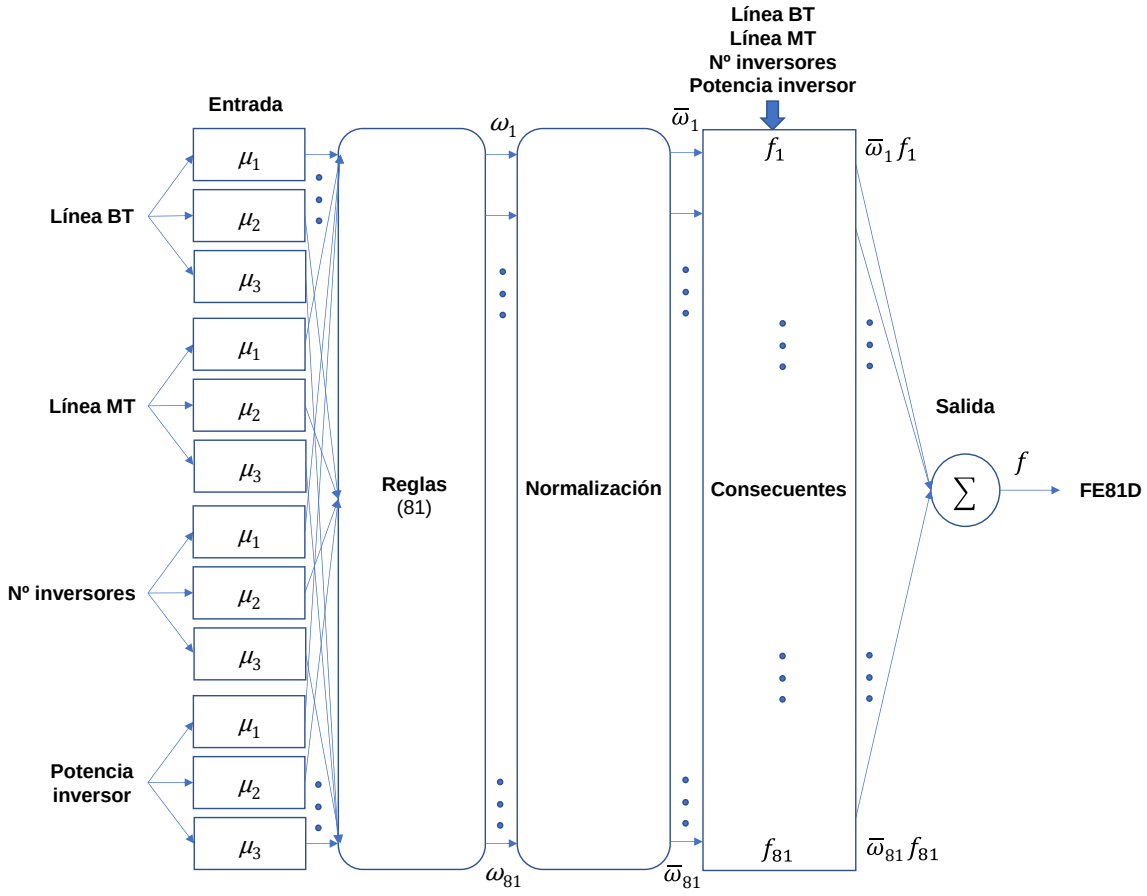


Figura 30. *Detalle del sistema neurodifuso de un modelo ANFIS.*

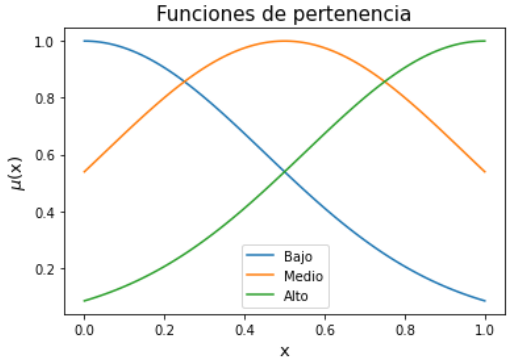


Figura 31. *Funciones de pertenencia definidas para ANFIS.*

ANFIS procesa la información de entrada a través del número de *epochs* que se establezca. Se observa que un número mayor a 4 no conduce necesariamente a mejores resultados y en algún caso muestra inestabilidad. El resultado de cada modelo devuelve la evolución de la pérdida en cada iteración y el ajuste de la predicción a los datos de entrenamiento. La figura 32 muestra el caso concreto del modelo de duración de la fase de limpieza del terreno.

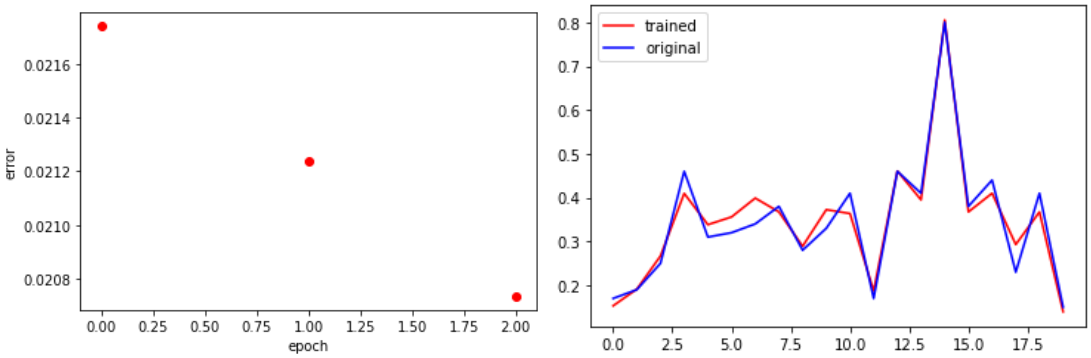


Figura 32. *Resultados gráficos del entrenamiento con ANFIS.*

Tras el entrenamiento, ANFIS devuelve:

- *Los valores de σ y μ de las funciones de pertenencia ajustadas*. Una variable de salida está representada por n variables de entrada, cada una con 3 funciones de pertenencia, por lo que el resultado son $3n$ funciones de pertenencia (típicamente 15, ya que el valor más común para n es igual a 5). Las modificaciones en los valores de σ y μ dados inicialmente no han sido, en general, significativos (<10%). La representación de las funciones de pertenencia resultantes, por tanto, no varían de forma sensible con respecto a las de la figura 31. Estas funciones de pertenencia son las que permiten establecer el peso de cada una de las reglas borrosas.
- *Los consecuentes p_n, r_i, de las reglas borrosas Takagi-Sugeno*. Nótese que la mayoría de las variables de salida dependen de 5 variables de entrada, lo que supone 3^5x 6 = 1458 parámetros para cada variable de salida.

Los **resultados** obtenidos con ANFIS se muestran en la tabla 13. El rendimiento promedio es en todas las métricas muy superior al de la línea base. También mejora el rendimiento del resto de algoritmos sobre el conjunto de test, no habiendo incurrido en un grado importante de sobreajuste. La única desventaja de ANFIS es su elevado tiempo de ejecución, que en promedio se demora más de 35 minutos.

Tabla 13. Rendimiento de ANFIS.

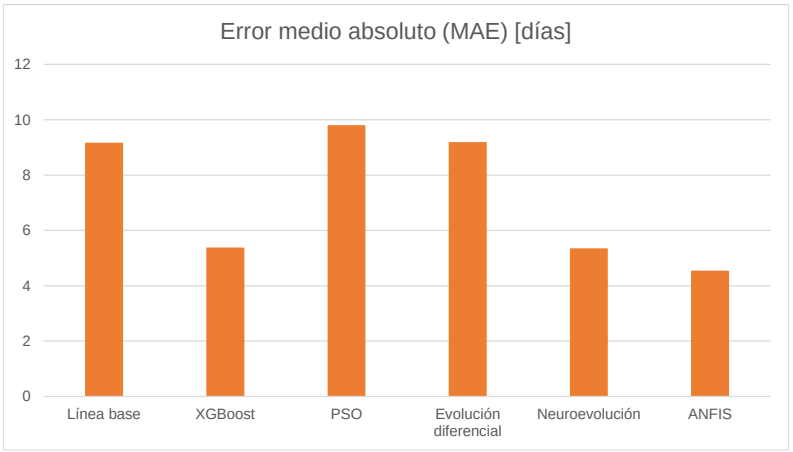
		Línea base	ANFIS
MAE entrenamiento	días	8,48	2,10
MSE entrenamiento	días^2	259,12	20,35
R^2 entrenamiento	-	0,0000	0,9242
MAE test	días	9,17	4,54
MSE test	días^2	275,86	67,19
R^2 test	-	-0,4201	0,6545
Tiempo	s	0,0000	2176,1557

5.6. Comparativa entre los distintos métodos

La comparativa en sí se inicia con un resumen básico que muestra las métricas principales obtenidas en promedio en los test (tabla 14 y figura 33). Para ello, del grupo de técnicas de árboles de regresión se ha seleccionado su representante con mejor rendimiento, *XGBoost*.

Tabla 14. Resumen de la comparativa entre las distintas técnicas.

		Línea base	XGBoost	PSO	Evolución diferencial	Neuro evolución	ANFIS
MAE test	días	9,17	5,38	9,80	9,19	5,36	4,54
MSE test	días^2	275,86	117,45	193,34	276,16	75,41	67,19
R^2 test	-	-0,4201	0,4354	0,0383	-0,4216	0,5909	0,6545



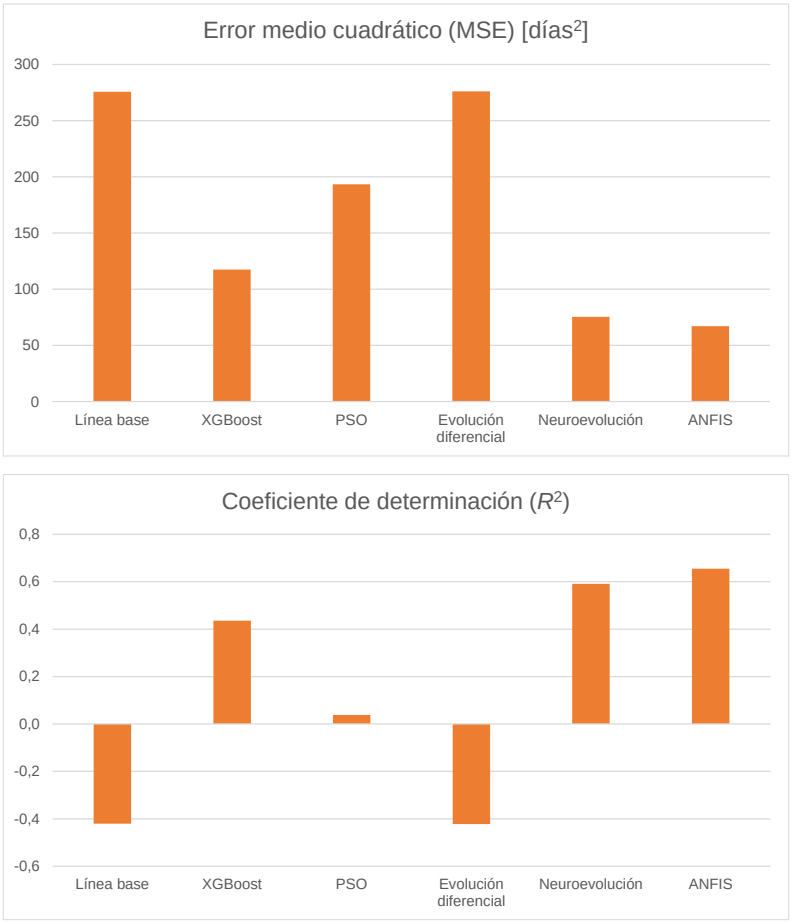


Figura 33. *Gráfico resumen de la comparativa entre las distintas técnicas.*

De este resumen básico, se pueden extraer que las técnicas que presentan un **mejor rendimiento** son, **por este orden, ANFIS, neuroevolución, *XGBoost*, optimización de enjambre de partículas** y **evolución diferencial.** ANFIS destaca por su buen rendimiento en todas las métricas, seguido por neuroevolución. *XGBoost* presenta un resultado de MAE casi al nivel de estas dos técnicas, si bien en la comparativa de MSE y R^2 ofrece un rendimiento inferior. Ello significa que sus predicciones son más sensibles a sufrir desvíos de gran magnitud y que el modelo explica en menor medida las variaciones en la salida. El siguiente modelo en cuanto a rendimiento es la optimización de enjambre de partículas. Aunque su MSE y R^2 mejoran la línea base, su rendimiento en MAE es peor, lo que significa que, en la práctica, no resulta de utilidad predictiva. Por último, los resultados de evolución diferencial son prácticamente análogos a los dados por la línea base, esto es, por la media.

La tabla 15 y la figura 34 muestran una comparativa más detallada en la que se incluyen todas las técnicas basadas en árboles de decisión, así como:

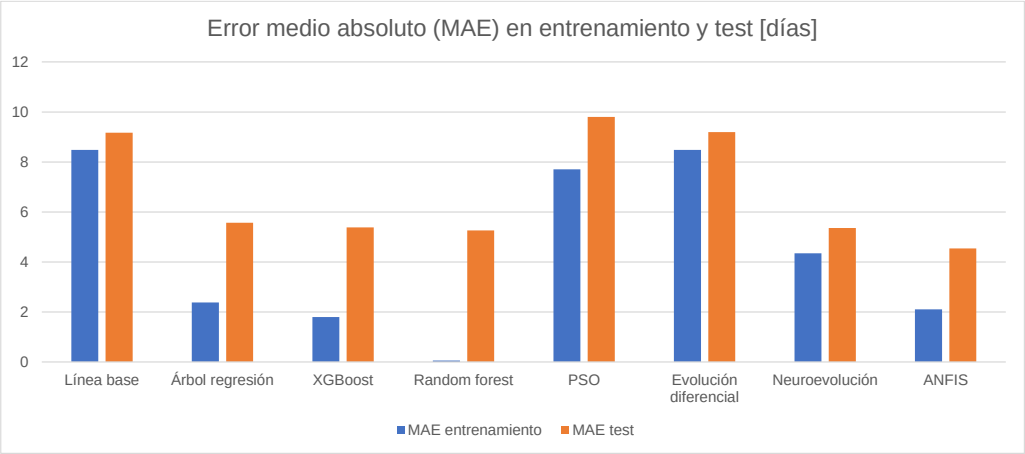
- **Métricas en el entrenamiento**. Su diferencia con las métricas del conjunto de test permiten observar el grado de sobreajuste de cada técnica. En concreto, las técnicas basadas en árboles son las que sufren de un mayor sobreajuste, especialmente más

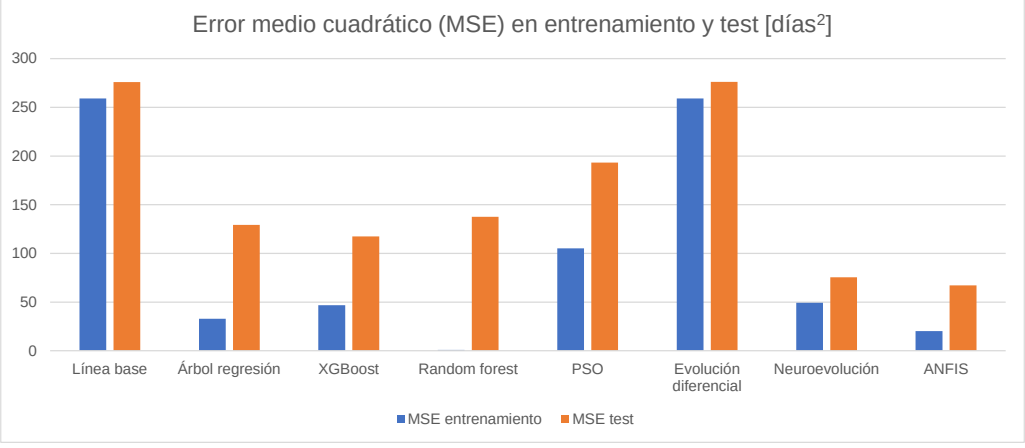
acusado en *Random forest*, con unos MAE y MSE en entrenamiento de dos órdenes de magnitud inferiores a los de test, y un R^2 en entrenamiento muy próximo a 1. En el resto de técnicas no se observa un grado tan importante de sobreajuste, si bien ANFIS obtiene unos resultados en entrenamiento claramente superiores a los de test.

- **Tiempo de ejecución**. El tiempo de ejecución de las técnicas basadas en árboles es relativamente rápido, variando entre los 26 segundos en promedio del árbol de regresión hasta los 157 segundos de *XGBoost*. La optimización de enjambre de partículas y evolución diferencial presentan tiempos medios de ejecución de 572 y 213 segundos. Los algoritmos con un mayor tiempo de ejecución son neuroevolución (704 segundos) y, especialmente, ANFIS (2176 segundos).

Tabla 15. Detalle de la comparativa entre las distintas técnicas.

		Línea base	Árbol regresión	XGBoost	Random forest	PSO	Evolución diferencial	Neuro evolución	ANFIS
MAE entrenamiento	días	8,48	2,38	1,80	0,06	7,71	8,48	4,34	2,10
MSE entrenamiento	días^2	259,12	32,89	46,74	0,96	105,28	259,13	49,41	20,35
R^2 entrenamiento	-	0,0000	0,8748	0,8166	0,9963	0,6150	0,0000	0,8079	0,9242
MAE test	días	9,17	5,57	5,38	5,26	9,80	9,19	5,36	4,54
MSE test	días^2	275,86	129,29	117,45	137,65	193,34	276,16	75,41	67,19
R^2 test	-	-0,4201	0,3650	0,4354	0,3225	0,0383	-0,4216	0,5909	0,6545
Tiempo	s	0	26	157	47	572	213	704	2176





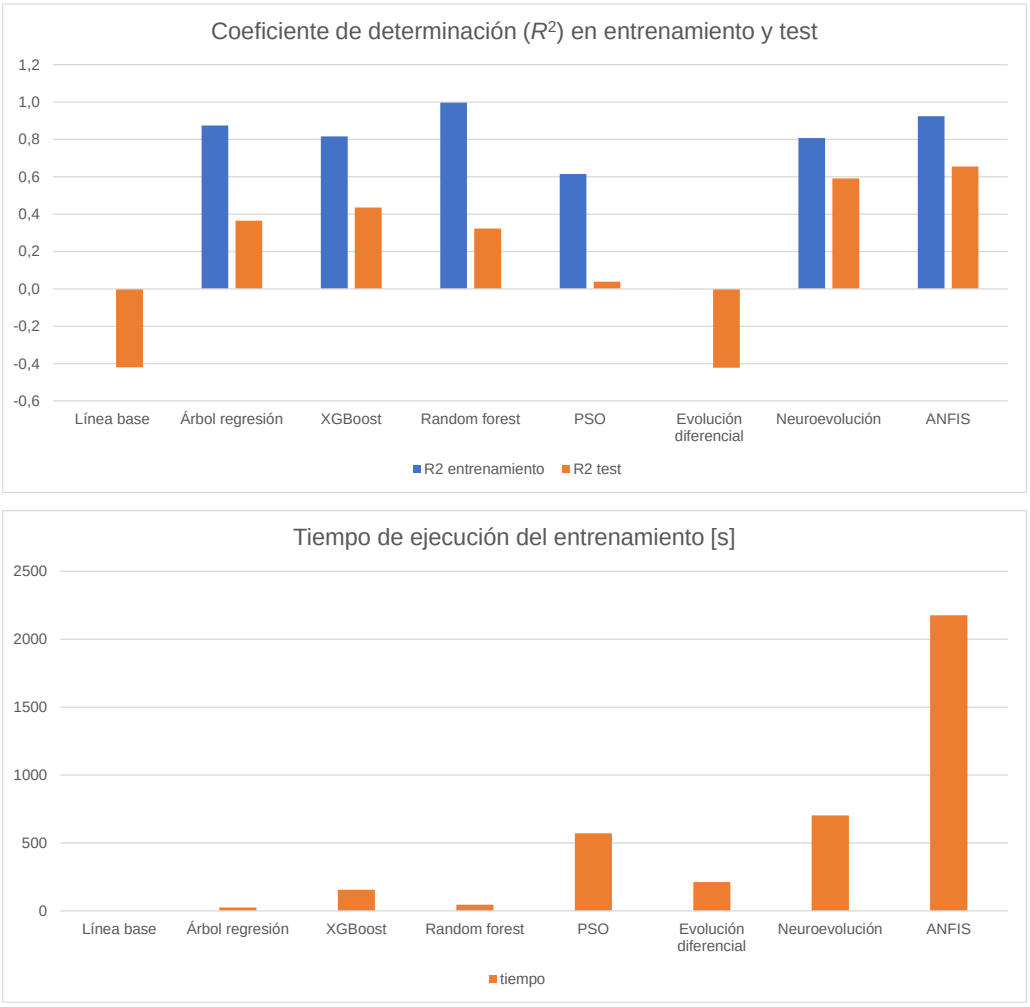


Figura 34. *Gráficos de detalle de la comparativa entre las distintas técnicas.*

Por último, se muestran los resultados obtenidos en las pruebas realizadas con cada uno de los ocho conjuntos de test, para observar la consistencia en el desempeño de cada algoritmo, así como posibles puntos ciegos. En este caso, dado el elevado volumen de datos, solamente se muestra la información gráfica (figura 35), para mayor claridad de interpretación de la información agregada.

Los resultados muestran consistencia en el orden de desempeño de los algoritmos. Cabe destacar que en dos de los test (número 4 y 6) el mejor resultado en MAE es obtenido por *Random forest*. Es precisamente en estos dos test, así como en el test número 5, donde neuroevolución obtiene el mejor rendimiento de entre todas las técnicas en las métricas MSE y R^2. Si bien en estos test concretos ANFIS no es la técnica con mejor desempeño en todas las métricas, sí ofrece el rendimiento más equilibrado en todas ellas.

Por último, en estas gráficas se observa la práctica coincidencia de los resultados de evolución diferencial con la línea base. Asimismo, queda de manifiesto que el rendimiento de esta técnica en especial y la de optimización de enjambre de partículas son insatisfactorios y claramente inferiores al resto de técnicas.

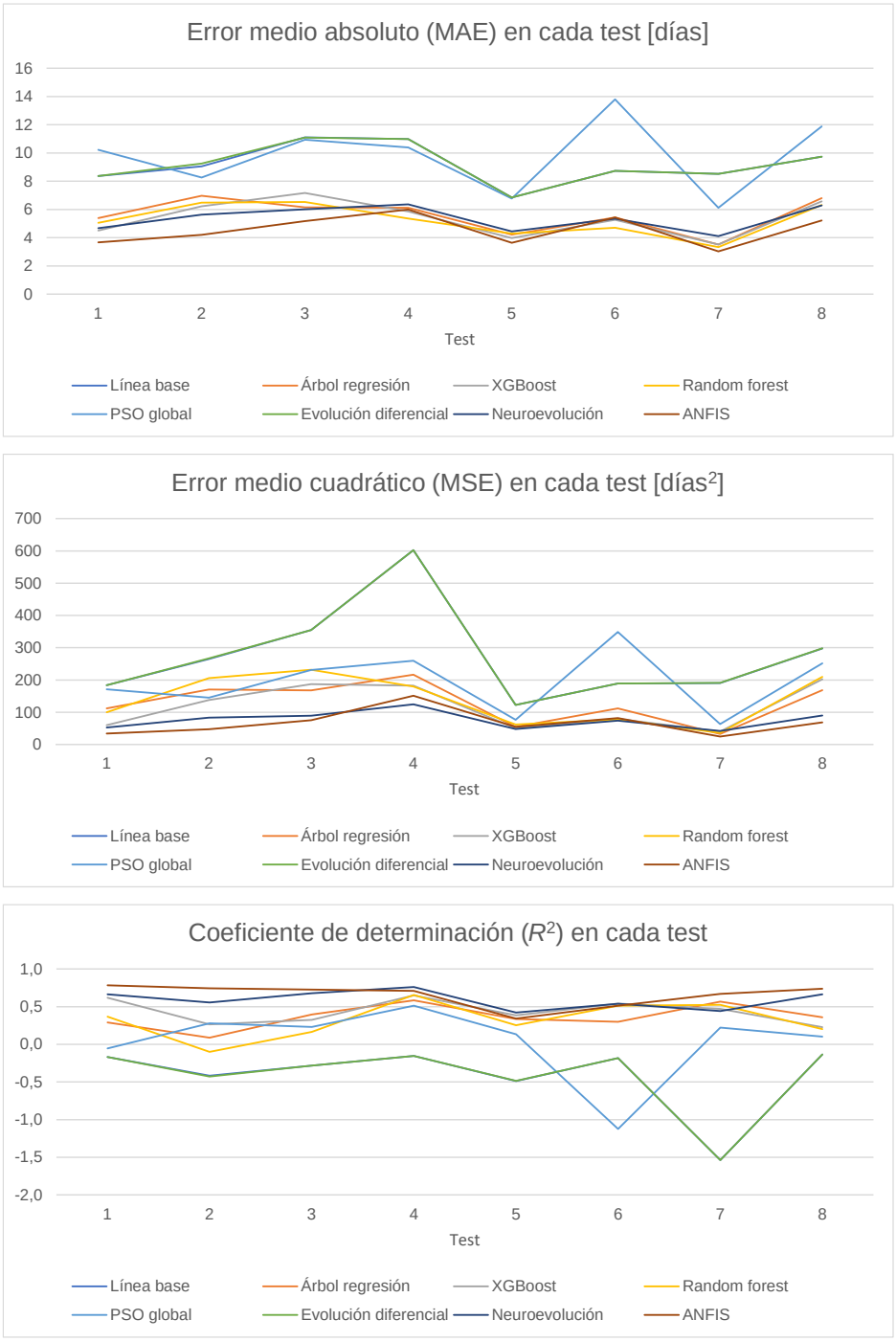


Figura 35. *Métricas obtenidas en cada uno de los test.*

6. Discusión y análisis de resultados

A continuación, se analizará el significado de los resultados presentados en el anterior capítulo, mencionando las ventajas e inconvenientes de las soluciones evaluadas. Esta discusión se realizará por orden de optimalidad de los modelos: ANFIS, neuroevolución, árboles de decisión -en todas sus variantes-, optimización de enjambre de partículas y evolución diferencial. También se discutirá el valor de los resultados alcanzados desde un punto de vista práctico, más allá de la comparativa entre las distintas técnicas.

Tal y como se ha expuesto, la técnica que ha devuelto mejor rendimiento en las métricas de error (MAE, MSE, R^2) ha sido **ANFIS**. Este buen rendimiento se obtiene de forma consistente en todos los conjuntos de entrada, sin incurrir en sobreajustes importantes. ANFIS es capaz de valorar adecuadamente la importancia de cada una de las entradas al modelo a través del conjunto de funciones de pertenencia y reglas borrosas y, en consecuencia, devolver una salida ajustada al valor y peso de cada una de las entradas. Este buen rendimiento en los resultados tiene su contrapartida en el coste computacional, tres veces superior al de la siguiente técnica más costosa (neuroevolución), con un tiempo promedio de cómputo de 35 minutos. En especial, el cálculo matricial de la parte del algoritmo de optimización basada en mínimos cuadrados aumenta el consumo de recursos de forma exponencial. Una manera de disminuir esta complejidad de cálculo es reducir el conjunto de reglas, eliminando aquellas con menor sentido físico (p.ej. aquellas en las que figure 'Potencia' grande y 'Área' pequeña).

Otra de las ventajas de ANFIS es la explicabilidad de sus salidas, que se obtienen operando las variables de entrada a través de relaciones lingüísticas (funciones de pertenencia y reglas) y lineales (consecuentes). Sin embargo, la trazabilidad de este modelo concreto no resulta inmediata, debido a que cada salida se compone del resultado ponderado de 3^n reglas, siendo n el número de variables de entrada (así, la salida de un modelo basado en 5 variables de entrada es el resultado ponderado de aplicar 243 reglas). El peso relativo de cada una de las reglas resulta un primer criterio eficiente para seleccionar aquellas con mayor influencia en la salida. No obstante, una vez más, la simplificación del conjunto de reglas puede ser de ayuda en este aspecto.

La siguiente técnica en cuanto a rendimiento es **neuroevolución**. El rendimiento en las métricas de error es inferior a ANFIS, en un 10% para MSE y R^2 y en un 20% para MAE. Este rendimiento es consistente en todos los test, por lo que los resultados obtenidos pueden calificarse como buenos, en especial habida cuenta de la importante reducción en el coste computacional con respecto a ANFIS (3 veces menor). Su implementación en Keras, además, permite dotar a este modelo de una gran flexibilidad a la hora de incorporar nuevos parámetros

de entrada que permitan afinar el modelo -cuestión que, en ANFIS, aumenta extraordinariamente la capacidad de cómputo necesaria-. Asimismo, Keras permite trabajar con los modelos obtenidos de una forma sencilla, tanto para su recuperación como para su puesta en producción.

Otra de las ventajas de este método es que es aquel con menor sobreajuste. Esto puede observarse en el hecho de que dispone de las ratios más favorables en las métricas de error en test frente a entrenamiento.

La principal desventaja de este método es su falta de explicabilidad. Un modelo como el presentado en la figura 27 cuenta con más de 70.000 parámetros, de los que ninguno tiene un significado físico, lo que, en la práctica, dificulta enormemente entender qué criterios subyacen tras un valor de salida determinado. Asimismo, las soluciones encontradas son menos óptimas que las obtenidas a través de ANFIS, aun cuando se ha empleado un coste computacional todavía considerable (con un promedio de 12 minutos por cada test).

En consecuencia, se aprecia un **rendimiento superior de las técnicas híbridas** frente a las técnicas básicas. En base a los test realizados, es posible responder afirmativamente a la cuestión planteada en el apartado 2.2.6 sobre si la inclusión de lógica borrosa, con ANFIS, es capaz de superar a un modelo basado en redes neuronales.

Dentro de las técnicas básicas, aquellas que han ofrecido mejores resultados son las basadas en **árboles de decisión**. Su clasificación por orden de rendimiento en cuanto a métricas de error en test es: *XGBoost*, *Random forest* y árbol de regresión aislado, sin que existan diferencias muy significativas entre todas ellas. Su MAE (≈5,4) es muy similar a la lograda con neuroevolución, llegando incluso a ser ligeramente mejor en el caso de *Random forest*, si bien los resultados en R^2 (≈0,4, frente a ≈0,6 de las técnicas híbridas) y MSE (≈120, frente a ≈70 de las técnicas híbridas) son claramente inferiores. Esto viene a significar que estas técnicas presentan un comportamiento peor en cuanto a errores grandes, una menor explicabilidad del modelo y la existencia de posibles sesgos (ya expuesta en el apartado 5.1). Pese a ello, el resultado es razonablemente bueno para obtener una primera aproximación, considerando que los tiempos de ejecución oscilan entre los 26 y los 157 segundos para un árbol aislado y para *XGBoost*, respectivamente. Se observa que todas estas técnicas adolecen de un importante sobreajuste, aun a pesar de haber diseñado los algoritmos de forma específica para minimizarlos. Este sobreajuste es especialmente claro en el caso de *Random forest*, con un R^2 promedio en entrenamiento de 0,9963, y menor en el caso de un árbol aislado.

Por último, solamente la técnica basada en un árbol aislado ofrece buena explicabilidad. En este caso, además, resulta clara y sencilla de interpretar (figura 21).

Los resultados obtenidos con **optimización de enjambre de partículas** han sido inferiores a los dados por árboles de decisión y pueden calificarse de insatisfactorios. MSE y R^2 mejoran a los dados por la línea base. Sin embargo, MAE, con un valor próximo a 10 días, ofrece un rendimiento inferior a la línea base, y que supone en torno al doble de la MAE obtenida en las técnicas hasta ahora comentadas. Tal y como se ha indicado en el apartado 5.2, esta técnica converge rápidamente a soluciones que, en general, son subóptimas. Aun habiendo optado por soluciones que permitan ampliar el conjunto de soluciones (como la generación de varios enjambres para cada variable de salida, de forma iterativa o con optimización local), no se han conseguido buenos resultados. El coste computacional para esta técnica ha sido relativamente elevado (en torno a 10 minutos). Una eventual ventaja de esta técnica, en caso de haber alcanzado resultados más satisfactorios, es su bajo sobreajuste, hecho que puede observarse del análisis de la diferencia entre las métricas de entrenamiento y test.

En cuanto a su explicabilidad, no es una propiedad de esta técnica *per se*. Sin embargo, la implementación realizada, basada en hipótesis de linealidad, sí permitiría valorar de forma sencilla cuáles de los parámetros de entrada tienen mayor influencia en una salida. De todos modos, una optimización lineal no ha supuesto una solución adecuada a este problema. Tal y como se ha observado, las técnicas que introducen mayor complejidad en su modelado permiten obtener mejores soluciones.

La técnica de la comparativa que ha devuelto peores resultados ha sido **evolución diferencial**. Las métricas de error son muy similares a las dadas por la línea base. Esto se debe a que el resultado de la predicción dada por este modelo es muy aproximadamente el valor medio de la variable en el conjunto de entrenamiento. Un posible motivo de la mala calidad de las predicciones es, una vez más, la hipótesis de linealidad adoptada entre las variables de entrada y las de salida. Esta hipótesis, si bien favorece la explicabilidad de las soluciones, impide explorar el conjunto de mejores soluciones, por lo que el algoritmo se estanca en regiones subóptimas, como la dada por el promedio de las variables de entrada. Tal y como se ha indicado previamente, el tiempo de ejecución es de los menores (entre 3 y 4 minutos, solamente superior al de los árboles de regresión), lo que supondría una ventaja en el caso de poder obtener soluciones satisfactorias.

Una perspectiva adicional en el análisis de resultados es su aportación en términos absolutos, más allá de la comparativa, para su **aplicación práctica**. Se tomará, para ello, el modelo que mejores resultados ha devuelto, ANFIS. Una primera medida de la bondad de los resultados alcanzados puede tomarse contrastando el valor de MAE en test (4,54 días) frente al valor absoluto medio de las variables de salida (29,17 días). Esto supone un error relativo promedio de un **16%** en el cálculo de las duraciones y los intervalos entre las distintas fases de construcción. Por otro lado, un coeficiente de determinación R^2 de 0,6545 puede considerarse

como muy bueno ante un conjunto de datos muy pequeño y de gran variabilidad. Sirva como referencia que la línea base dada por la media de los datos de entrenamiento obtiene un R^2 negativo, de -0,4201, cuando en conjuntos de test y entrenamiento homogéneos, la R^2 dada por una estimación basada en la media es 0.

La prueba final de la bondad del modelo consiste en **cotejar cronogramas reales de test frente a los cronogramas que elaboraría el modelo**. Para que la prueba sea lo más verosímil posible, se utilizará el modelo ANFIS obtenido con aquel test -de entre los ocho realizados- donde las métricas de error han sido las más similares a las métricas promedio. Dicho modelo es el obtenido con el test realizado con el número 2 (ver figura 35). Los resultados obtenidos con este modelo son notablemente buenos. Observando la **duración total** de los cronogramas obtenidos (tabla 16), su **error relativo** es de un **8%**, con un máximo de un 10% y sin que se observe un sesgo significativo al alza o a la baja (hecho que se ha confirmado sobre cronogramas realizados sobre otros conjuntos de test). Puede concluirse que los errores existentes en cada una de las fases individuales (16%) no contienen sesgos relevantes, lo que redunda en su cancelación mutua parcial, para alcanzar el citado error en la duración total de un 8%. El Anexo I.2 desarrolla la implantación práctica de este modelo. Es preciso mencionar que los resultados mostrados en este punto consideran únicamente el modelo objeto de comparativa, que es el de construcción de la planta. El modelo sencillo elaborado para la infraestructura de la conexión a la red (ver Anexo I.1) ha conseguido, debido a la simplicidad de este sistema, un error relativo y un coeficiente de determinación R^2 que son incluso inferiores a los del modelo de la planta.

Tabla 16. Resultado de los cronogramas obtenidos a partir del modelo ANFIS.

		Planta 1	**Planta 21**	**Planta 11**	**Planta 3**	**Planta 5**
		96 MW	146 MW	65 MW	50 MW	37 MW
Duración real	días	253	340	220	226	162
Duración predicha	días	278	311	242	215	175
Error total	días	25	-29	22	-11	13
Error relativo	%	10%	9%	10%	5%	8%

7. Conclusiones y trabajo futuro

Esta sección final discutirá las conclusiones del trabajo llevado a cabo, desde su enfoque hasta las soluciones obtenidas, contrastando si se han alcanzado los objetivos específicos planteados en el capítulo 3. Por último, se sugerirán posibles líneas de investigación futuras surgidas como consecuencia de este trabajo.

7.1. Conclusiones

La necesidad de impulsar la construcción de infraestructuras de generación renovable sirve como motivación para tratar de resolver, a través de técnicas de inteligencia artificial, la principal dificultad para cumplir el plazo de construcción de estos proyectos: la elaboración de una planificación adecuada. Una herramienta capaz de generar cronogramas de forma automática, sin influencia de sesgos humanos, puede resultar de gran ayuda a la hora de establecer planificaciones iniciales de referencia. Este trabajo concreto ha profundizado en la generación de cronogramas de construcción de plantas de gran potencia basadas en tecnología fotovoltaica a partir de datos históricos.

Para ello, se ha establecido el **contexto** de los problemas de planificación y la aplicación de técnicas de inteligencia artificial para su resolución. Se ha partido de una perspectiva general, en la que se plantean tanto los modelos generales como los de aplicación a este problema, basados en casos. Se ha realizado, asimismo, una revisión completa del estado del arte, tratando diferentes técnicas básicas e híbridas, e incidiendo de manera especial en aquellas que ofrecen un resultado más prometedor. Como resultado de dicha revisión, se han incluido en la comparativa cinco técnicas de carácter complementario, incorporando criterios de optimalidad, explicabilidad y novedad sobre el estado del arte: técnicas básicas, como árboles de decisión, optimización de enjambre de partículas y evolución diferencial, y técnicas híbridas, como neuroevolución y ANFIS. Con ello, se ha cumplido el *primer objetivo específico*: la selección de las técnicas de inteligencia artificial más adecuadas para la comparativa.

Un trabajo preliminar importante ha consistido en el **análisis exploratorio**, revisión y depuración de los datos de origen. Este trabajo ha sido crucial para plantear un modelo efectivo, a la vista del reducido número de registros -25- y el elevado número de variables -23 de entrada y 62 de salida-. De este estudio se han obtenido las correlaciones más significativas, que han permitido desglosar el modelo en dos partes (modelo de la planta y modelo de la conexión con la red) y establecer las variables de entrada más significativas para cada una de las variables de salida.

La **implementación** de cada uno de estos modelos en software se ha llevado a cabo en lenguaje Python. La existencia de librerías científicas en este lenguaje, de las que se aportan las correspondientes referencias, ha permitido simplificar el desarrollo del código. Una parte importante de la investigación ha consistido en el **ajuste** de estos modelos, en concreto, hiperparámetros, funciones (coste, pertenencia) e incluso estrategias de optimización. Esto supone el cumplimiento del *segundo objetivo específico*: la elaboración de los modelos conceptuales, su implementación en software y optimización.

El entrenamiento de los modelos se ha aplicado a un conjunto de test llevados a cabo de forma pseudoaleatoria, a fin de garantizar la inclusión de todo el conjunto de entrada minimizando los sesgos. El **resultado** obtenido en estos test muestra un rendimiento superior de las **técnicas híbridas** frente a las técnicas básicas. El grado de complejidad añadido de estas técnicas se traduce en unos mejores resultados en cuanto a métricas de error, si bien supone un coste computacional más importante. La técnica que mejores resultados ha devuelto es **ANFIS**, observándose que su error medio absoluto es de en torno a 4,5 días y su coeficiente de determinación R^2, de aproximadamente 0,65. El algoritmo de **neuroevolución** ha conseguido, asimismo, unos resultados satisfactorios, ligeramente inferiores a los de ANFIS, con un menor coste computacional. En cuanto a las técnicas básicas, solamente las técnicas basadas en **árboles de decisión** consiguen mejorar claramente la predicción dada por la media del conjunto de entrenamiento, que ha servido como línea base. Estas técnicas se han implementado con árboles de regresión aislados, *Random forest* y *XGBoost*, habiendo obtenido este último el mejor rendimiento. Estas técnicas consiguen un error medio absoluto bajo, si bien contienen sesgos importantes, una variabilidad explicada inferior a los modelos híbridos y un importante sobreajuste. En cambio, su tiempo de ejecución es bajo y, en el caso del árbol de regresión aislado, permiten un modelado explicable, lo que resulta interesante para realizar análisis previos de modelos y observar el comportamiento de las variables de salida en función de las de entrada. Tanto la **optimización de enjambre de partículas** como **evolución diferencial** han sufrido de problemas de estancamiento en soluciones subóptimas, motivadas tanto por la naturaleza de los algoritmos, como por la definición de un modelo lineal que no permite capturar la complejidad del sistema. En el caso de evolución diferencial, las mejores soluciones obtenidas son iguales a la dada por la línea base. Esto supone el cumplimiento del *tercer objetivo específico*: la realización de la comparativa entre los modelos y determinación de un posible modelo óptimo. La obtención de un modelo con resultado superior al resto y mejor que la línea base ha permitido asimismo descartar la hipótesis negativa del experimento formulada en el apartado 3.3, consistente en que "*ninguno de los modelos de la comparativa es capaz de obtener una predicción del cronograma de*

construcción de plantas fotovoltaicas mejor que las predicciones dadas por el resto de modelos, ni es capaz de superar la línea base establecida por el valor medio de cada variable".

Por último, se ha desarrollado una **herramienta** capaz de establecer planificaciones de referencia en base al modelo y a partir de 14 variables de entrada -una de las cuales es la fecha de inicio de la construcción-. Los resultados obtenidos por esta herramienta, con un error relativo de un 8% en la duración total de la planificación, han sobrepasado el objetivo del 20% establecido en el apartado 3.1, lo que da cumplimiento al *cuarto objetivo específico*: el desarrollo de una herramienta que permita obtener un cronograma de construcción a partir de unos datos de entrada gracias al modelo óptimo. Queda así de manifiesto que el trabajo desarrollado supone una contribución relevante sobre los resultados alcanzados en el estado del arte (Gil et al., 2021a), abriendo una serie de líneas futuras de investigación, que se discuten en el próximo apartado.

7.2. Líneas de trabajo futuro

A continuación, se presentan las posibles líneas de trabajo que han surgido como consecuencia de este estudio:

1. **Testear nuevos modelos híbridos**, a la vista del superior resultado alcanzado por este tipo de modelos. EFHNN forma parte de líneas de investigación actualmente en curso, si bien modelos más sencillos como EFNIM o los basados en NEAT pueden conseguir resultados satisfactorios. Como consecuencia del prometedor resultado obtenido por los árboles de decisión, una posible línea de investigación futura pasaría por su hibridación con modelos que permitan establecer una salida continua y que disminuyan su elevado sobreajuste. Los modelos de optimización de enjambre de partículas y evolución diferencial también podrían beneficiarse de su hibridación, a fin de explorar modelos de mayor complejidad que los lineales. Por último, tal y como se ha mencionado, una simplificación de la base de reglas de ANFIS en base a criterios expertos permitiría incluir un mayor número de variables de entrada por cada variable de salida. Para este tipo de test también se propone el empleo de sistemas específicos de cómputo, con mayor capacidad que el utilizado para el desarrollo de este trabajo.
2. **Desarrollar modelos básicos** que permitan superar las limitaciones de los que han formado parte de la comparativa. *Support Vector Machine* es un algoritmo apto para problemas como el considerado, con un reducido número de registros y un elevado número de variables. Estas propiedades lo han incluido en actuales líneas de investigación. Los modelos bioinspirados desarrollados en mayor profundidad en el

estado del arte y que no han sido implementados (colonia de hormigas y, especialmente, *shuffled frog-leaping*) podrían superar la tendencia a caer en soluciones subóptimas que se ha observado en los algoritmos testeados en esta tipología.

3. **Modelar el problema de forma paramétrica, con dependencias no lineales.** Esta línea de investigación requiere un conocimiento experto del dominio, a fin de establecer los parámetros necesarios (p.ej. potencias o productos de las variables de entrada).
4. **Enriquecer el conjunto de entrada.** Esta línea de investigación dispone de diferentes vertientes:
    a. Un mayor número de registros, que permita mejorar la inferencia.
    b. Una mayor diversidad en los datos de las variables de entrada, que permita mejorar la inferencia. El actual conjunto de datos solo permite predecir planificaciones de “tipo 0”, esto es, no optimizadas. Otra restricción se encuentra en que todas las instalaciones se encuentran conectadas a una tensión de 132 kV, impidiendo la inferencia en tensiones diferentes. Asimismo, tal y como se ha mencionado, 9 de las 25 planificaciones son sobre instalaciones muy similares.
    c. Incorporar variables adicionales que expliquen valores aparentemente contradictorios, esto es, por qué un intervalo de construcción es muy diferente entre dos plantas, cuando las características que lo determinan son iguales (p.ej. el plazo de construcción del cierre perimetral en dos plantas con mismo perímetro). Esta línea de trabajo se complementa con una depuración y homogeneización profunda de los datos, a realizar por parte de expertos.
5. Una línea de investigación más profunda permitiría **ajustar la duración de las tareas en función de la época del año**, dado que puede afectar sensiblemente al avance de los trabajos. Sirvan como ejemplos no exhaustivos el número de horas de luz diaria disponibles para trabajar, o la climatología, que puede condicionar la ejecución de accesos, o el acondicionamiento de determinados tipos de terreno, en temporadas de alta pluviosidad.

8. Bibliografía

Abraham, A. (2002). EvoNF: A framework for optimization of fuzzy inference systems using neural network learning and evolutionary computation. *IEEE International Symposium on Intelligent Control - Proceedings*, 327-332. https://doi.org/10.1109/isic.2002.1157784

Adeli, H., y Karim, A. (1997). Scheduling/Cost Optimization and Neural Dynamics Model for Construction. *Journal of Construction Engineering and Management*, *123*(4), 450-458. https://doi.org/10.1061/(asce)0733-9364(1997)123:4(450)

Aengchuan, P., y Phruksaphanrat, B. (2018). Comparison of fuzzy inference system (FIS), FIS with artificial neural networks (FIS + ANN) and FIS with adaptive neuro-fuzzy inference system (FIS + ANFIS) for inventory control. *Journal of Intelligent Manufacturing*, *29*(4), 905-923. https://doi.org/10.1007/s10845-015-1146-1

Agarwal, A., Colak, S., y Erenguc, S. (2011). A Neurogenetic approach for the resource-constrained project scheduling problem. *Computers and Operations Research*, *38*(1), 44-50. https://doi.org/10.1016/j.cor.2010.01.007

Agyekum-Mensah, G., y Knight, A. D. (2017). The professionals' perspective on the causes of project delay in the construction industry. *Engineering, Construction and Architectural Management*, *24*(5). https://doi.org/10.1108/ECAM-03-2016-0085

Ashuri, B., y Tavakolan, M. (2021). Fuzzy enabled hybrid genetic algorithm particle swarm optimization approach to solve TCRO problems in construction project planning. *Journal ofConstruction Engineering and Management*, *138*(9), 1065-1074.

Avila Rondon, R., Carvalho, A., y Hernandez, G. (2008). Neural network modelling and simulation of the scheduling. *Eighth IFIP international conference on information technology for balanced automation systems, Porto, Portuga*, 231-238, Springer USA.

Azadeh, A., Hosseini, N., Abdolhossein Zadeh, S., y Jalalvand, F. (2015). A hybrid computer simulation-adaptive neuro-fuzzy inference system algorithm for optimization of dispatching rule selection in job shop scheduling problems under uncertainty. *The International Journal of Advanced Manufacturing Technology*, *79*. https://doi.org/10.1007/s00170-015-6795-x

Blomquist, T., Hällgren, M., Nilsson, A., y Söderholm, A. (2010). Insights and Trends: Current Portfolio, Programme, and Project Management Practices. *Project Management Journal*,

Vol. 41(No. 1), 5-16.

Bowen, P. A., Cattel, K. S., Hall, K. A., Edwards, P. J., y Pearl, R. G. (2012). Perceptions of Time, Cost and Quality Management on Building Projects. *Construction Economics and Building*, *2*(2), 48-56. https://doi.org/10.5130/ajceb.v2i2.2900

Chen, S., Fang, S., y Tang, R. (2019). A reinforcement learning based approach for multi-projects scheduling in cloud manufacturing. *International Journal of Production Research*, *57*(10), 3080-3098. https://doi.org/10.1080/00207543.2018.1535205

Chen, W., y Ni, X. (2014). Chaotic differential evolution algorithm for resource constrained project scheduling problem. *International Journal of Computing Science and Mathematics*, *5*(1), 81-93.

Cheng, M. Y., Chang, Y. H., y Korir, D. (2019). Novel Approach to Estimating Schedule to Completion in Construction Projects Using Sequence and Nonsequence Learning. *Journal of Construction Engineering and Management*, *145*(11), 04019072. https://doi.org/10.1061/(asce)co.1943-7862.0001697

Cheng, M. Y., Hoang, N. D., Roy, A. F. V., y Wu, Y. W. (2012). A novel time-depended evolutionary fuzzy SVM inference model for estimating construction project at completion. *Engineering Applications of Artificial Intelligence*, *25*(4), 744-752. https://doi.org/10.1016/j.engappai.2011.09.022

Cheng, M. Y., Lien, L. C., Tsai, H. C., y Chen, P. H. (2012). Artificial Intelligence Approaches to Dynamic Project Success Assessment Taxonomic. *Life Science Journal 2012*, *9(4)*, 49-56.

Cheng, M. Y., Tsai, H. C., y Hsieh, W. S. (2009). Web-based conceptual cost estimates for construction projects using Evolutionary Fuzzy Neural Inference Model. *Automation in Construction*, *18*(2), 164-172. https://doi.org/10.1016/j.autcon.2008.07.001

Cheng, M. Y., Tsai, H. C., Ko, C. H., y Chang, W. Te. (2008). Evolutionary Fuzzy Neural Inference System for Decision Making in Geotechnical Engineering. *Journal of Computing in Civil Engineering*, *22*(4), 272-280. https://doi.org/10.1061/(asce)0887-3801(2008)22:4(272)

Cheng, M. Y., Tsai, H. C., y Sudjono, E. (2009). Evolutionary fuzzy hybrid neural network for conceptual cost estimates in construction projects. *2009 26th International Symposium on Automation and Robotics in Construction, ISARC 2009*, *2005*, 512-519.

https://doi.org/10.22260/isarc2009/0040

Cheng, M. Y., Tsai, H. C., y Sudjono, E. (2011). Evaluating subcontractor performance using evolutionary fuzzy hybrid neural network. *International Journal of Project Management*, *29*(3), 349-356. https://doi.org/10.1016/j.ijproman.2010.03.005

Chua, D. K. H., Loh, P. K., Kog, Y. C., y Jaselskis, E. J. (1997). Neural networks for construction project success. *Expert Systems with Applications*, *13*(4), 317-328.

Comisión Europea. (2019). El Pacto Verde Europeo. *Comunicación de la Comisión al Parlamento Europeo, al Consejo Europeo, al Consejo, al Comité Económico y Social Europeo y al Comité de las Regiones.*, 28.

Das, S., y Suganthan, P. N. (2011). Differential evolution: A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, *15*(1), 4-31. https://doi.org/10.1109/TEVC.2010.2059031

Dorigo, M., Birattari, M., y Stützle, T. (2019). Ant colony optimisation. *IEEE Computational Intelligence Magazine*, *1*(4), 28-39. https://doi.org/10.1007/978-3-319-93025-1_3

Ershadi, M. J., Qhanadi Taghizadeh, O., y Hadji Molana, S. M. (2021). Selection and performance estimation of Green Lean Six Sigma Projects: a hybrid approach of technology readiness level, data envelopment analysis, and ANFIS. *Environmental Science and Pollution Research*, *28*(23), 29394-29411. https://doi.org/10.1007/s11356-021-12595-5

Eusuff, M., Lansey, K., y Pasha, F. (2006). Shuffled frog-leaping algorithm: A memetic meta-heuristic for discrete optimization. *Engineering Optimization*, *38*(2), 129-154. https://doi.org/10.1080/03052150500384759

Faghihi, V., Nejat, A., Reinschmidt, K. F., y Kang, J. H. (2015). Automation in construction scheduling: a review of the literature. *International Journal of Advanced Manufacturing Technology*, *81*(9-12), 1845-1856. https://doi.org/10.1007/s00170-015-7339-0

Fang, C., y Wang, L. (2012). An effective shuffled frog-leaping algorithm for resource-constrained project scheduling problem. *Computers and Operations Research*, *39*(5), 890-901. https://doi.org/10.1016/j.cor.2011.07.010

Fayek, A. R., y Lourenzutti, R. (2018). Introduction to Fuzzy Logic in Construction Engineering and Management. En *Fuzzy Hybrid Computing in Construction Engineering and*

Management. https://doi.org/10.1108/978-1-78743-868-220181001

Fazel, M. H., Sadat, A. A., Sotudian, S., y Castillo, O. (2020). A state of the art review of intelligent scheduling. En *Artificial Intelligence Review* (Vol. 53, Número 1). Springer Netherlands. https://doi.org/10.1007/s10462-018-9667-6

Ge, H.-W., Sun, L., Liang, Y.-C., y Qian, F. (2008). An effective PSO and AIS-based hybrid intelligent algorithm for job-shop scheduling. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, *38*(2), 358-368.

Georgy, M. E., Chang, L. M., y Zhang, L. (2005). Prediction of engineering performance: A neurofuzzy approach. *Journal of Construction Engineering and Management*, *131*(5), 548-557.

Gil, J., Martínez, J., y González, R. (2021a). *Planificación Y Gestion De Proyectos De Plantas Fotovoltaicas Aplicando Inteligencia Artificial. August*, 1-15.

Gil, J., Martínez, J., y González, R. (2021b). The application of artificial intelligence in project management research: A review. *International Journal of Interactive Multimedia and Artificial Intelligence*, *6*(6), 54-66. https://doi.org/10.9781/ijimai.2020.12.003

Gingnell, L., Franke, U., Lagerström, R., Ericsson, E., y Lilliesköld, J. (2014). Quantifying Success Factors for IT Projects—An Expert-Based Bayesian Model. *Inf. Syst. Manag.*, *31*(21-36).

Guo, W., Vanhoucke, M., Coelho, J., y Luo, J. (2021). Automatic detection of the best performing priority rule for the resource-constrained project scheduling problem. *Expert Systems with Applications*, *167*(October), 114116. https://doi.org/10.1016/j.eswa.2020.114116

Guzmán, D., y Castaño, V. M. (2006). La Lógica difusa en ingeniería: Principios, aplicaciones y futuro. *Ciencia y Tecnología*, *24*(2), 22.

Harvey, M. (2017). *Let's evolve a neural network with a genetic algorithm*. https://blog.coast.ai/lets-evolve-a-neural-network-with-a-genetic-algorithm-code-included-8809bece164

Hashemi Golpayegani, S. A., y Emamizadeh, B. (2007). Designing work breakdown structures using modular neural networks. *Decision Support Systems*, *44*(1), 202-222. https://doi.org/10.1016/J.DSS.2007.03.013

Inche, J., Andía, Y., Huamanchumo, H., López, M., Vizcarra, J., y Flores, G. (2014). PARADIGMA CUANTITATIVO: Un Enfoque Empírico y Analítico. *Industrial Data*, *6*(1), 023. https://doi.org/10.15381/idata.v6i1.5938

IRENA. (2020). *Renewable capacity statistics 2020 International Renewable Energy Agency (IRENA), Abu Dhabi*.

Jang, J. S. R. (1993). ANFIS: Adaptive-Network-Based Fuzzy Inference System. *IEEE Transactions on Systems, Man and Cybernetics*, *23*(3), 665-685. https://doi.org/10.1109/21.256541

Kachitvichyanukul, V. (2012). Comparison of Three Evolutionary Algorithms : *Industrial Engineering y Management Systems*, *11*(3), 215-223.

Kahneman, D., y Tversky, A. (1982). Intuitive prediction: Biases and corrective procedures. En *Judgment under Uncertainty* (pp. 414-421). Cambridge University Press. https://doi.org/10.1017/CBO9780511809477.031

Kartam, N. A., Levitt, R. E., y Wilkins, D. E. (1991). Extending Artificial Intelligence Techniques for Hierarchical Planning. *Journal of Computing in Civil Engineering*, *5*(4), 464-477. https://doi.org/10.1061/(ASCE)0887-3801(1991)5:4(464)

Ko, C. H., y Cheng, M. Y. (2003). Hybrid use of AI techniques in developing construction management tools. *Automation in Construction*, *12*(3), 271-281. https://doi.org/10.1016/S0926-5805(02)00091-2

Ko, C. H., y Cheng, M. Y. (2007). Dynamic Prediction of Project Success Using Artificial Intelligence. *Journal of Construction Engineering and Management*, *133*(4), 316-324. https://doi.org/10.1061/(asce)0733-9364(2007)133:4(316)

Ko, C. H., Cheng, M. Y., y Wu, T. K. (2007). Evaluating sub-contractors performance using EFNIM. *Automation in Construction*, *16*(4), 525-530. https://doi.org/10.1016/j.autcon.2006.09.005

Lewis, F. L., y Vrabie, D. (2009). Reinforcement learning and adaptive dynamic programming for feedback control. *IEEE Circuits and Systems Magazine*, *9*(3), 32-50. https://doi.org/10.1109/MCAS.2009.933854

Lien, L. C., y Cheng, M. Y. (2014). Particle bee algorithm for tower crane layout with material quantity supply and demand optimization. *Automation in Construction*, *45*, 25-32.

https://doi.org/10.1016/j.autcon.2014.05.002

Magaña, D., y Fernández, J. C. (2015). Artificial Intelligence Applied to Project Success: A Literature Review. *International Journal of Interactive Multimedia and Artificial Intelligence*, *3*(5), 77. https://doi.org/10.9781/IJIMAI.2015.3510

Meggs, T. (2020). *Python implementation of an Adaptive neuro fuzzy inference system*. GitHub. https://github.com/twmeggs/anfis

Merkle, D., Middendorf, M., y Schmeck, H. (2002). Ant colony optimization for resource-constrained project scheduling. *IEEE Transactions on Evolutionary Computation*, *6*(4), 333-346.

Miranda, L. J. V. (2017). *PySwarms*. PySwarms 1.3.0 documentation. https://pyswarms.readthedocs.io/en/latest/

Neri, F., y Tirronen, V. (2010). Recent advances in differential evolution: A survey and experimental analysis. En *Artificial Intelligence Review* (Vol. 33, Números 1-2). https://doi.org/10.1007/s10462-009-9137-2

Norvig, P., y Russell, S. (2004). *Inteligencia artificial. Un enfoque moderno* (2ª edición). PEARSON EDUCACIÓN, S.A.

Obayashi, M., Nishida, T., Kuremoto, T., Kobayashi, K., y Feng, L. B. (2010). A reinforcement learning system embedded agent with neural network-based multi-valued pattern memory structure. *ICCAS 2010 - International Conference on Control, Automation and Systems*, *May 2014*, 176-181.

Pellerin, R., Perrier, N., y Berthaut, F. (2020). A survey of hybrid metaheuristics for the resource-constrained project scheduling problem. En *European Journal of Operational Research* (Vol. 280, Número 2). https://doi.org/10.1016/j.ejor.2019.01.063

Portoleau, T., Artigues, C., y Guillaume, R. (2020). *Robust Predictive-Reactive Scheduling : An Information-Based Decision Tree Model* (Vol. 2). Springer International Publishing. https://doi.org/10.1007/978-3-030-50153-2

Prieto, R. (2019). *Impacts of Artificial Intelligence on Management of Large Complex Projects*.

Project Management Institute Inc. (2017). *Guía de los Fundamentos para la Dirección de Proyectos, (Guía del PMBOK®)* (6.ª ed.).

Rahmani, N., Zeighami, V., y Akbari, R. (2015). A study on the performance of differential search algorithm for single mode resource constrained project scheduling problem. *Decision Science Letters*, *4*(4), 537-550.

Rondon, F. (2015). *Sistemas fotovoltaicos. Consideraciones técnicas*. https://feriaexposolar.com/wp-content/uploads/2018/12/ABB-SKP-Consideraciones-técnicas-y-Dimensionamiento.pdf

Sallam, K. M., Chakrabortty, R. K., y Ryan, M. J. (2021). A reinforcement learning based multi-method approach for stochastic resource constrained project scheduling problems. *Expert Systems with Applications*, *169*(September 2020), 114479. https://doi.org/10.1016/j.eswa.2020.114479

Scikit-learn. (s. f.-a). *Decision tree regressor*. Scikit-learn 1.1.1 documentation. https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html

Scikit-learn. (s. f.-b). *Metrics and scoring: quantifying the quality of predictions*. Scikit-learn 1.1.1 documentation. Recuperado 1 de julio de 2022, de https://scikit-learn.org/stable/modules/model_evaluation.html

SciPy. (2020). *Differential evolution*. SciPy. https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.differential_evolution.html

Seresht, N. G., Lourenzutti, R., Salah, A., y Fayek, A. R. (2018). Overview of Fuzzy Hybrid Techniques in Construction Engineering and Management. En *Fuzzy Hybrid Computing in Construction Engineering and Management*. https://doi.org/10.1108/978-1-78743-868-220181002

Shou, Y. Y. (2006). An ant colony algorithm for improving the net present values of resource constrained projects. *Journal of the Chinese Institute of Industrial Engineers*, *23*(6), 478-483. https://doi.org/10.1080/10170660609509344

Song, X., Xu, J., Shen, C., Peña-Mora, F., y Zeng, Z. (2017). A decision making system for construction temporary facilities layout planning in large-scale construction projects. *International Journal of Civil Engineering*, *15*, 333-353.

Soto, G., Hernández, J., Almarza, D., Jofré, I., y Ukar, A. (2018). *Guía de operación y mantenimiento de sistemas fotovoltaicos*. 54.

Stanley, K. O., y Miikkulainen, R. (2002). Evolving Neural Networks through Augmenting Topologies. *Evolutionary Computation*, *10*(2), 99-127. https://doi.org/10.1162/106365602320169811

Tiruneh, G. G., Fayek, A. R., y Sumati, V. (2020). Neuro-fuzzy systems in construction engineering and management research. *Automation in Construction*, *119*(July), 103348. https://doi.org/10.1016/j.autcon.2020.103348

UNEF. (2021). *Annual report. 2021. Energía solar fotovoltaica. Nuestra Energía cambia el mundo*.

Vamvakeridou-Lyroudia, L. S., Walters, G. A., y Savic, D. A. (2005). Fuzzy multiobjective optimization of water distribution networks. *Journal of Water Resources Planning and Management*, *131*(6), 467-476.

XGBoost. (s. f.). *XGBoost Python package*. XGBoost. https://xgboost.readthedocs.io/en/stable/python/index.html

Zhang, H., Li, H., y Tam, C. M. (2006). Particle swarm optimization for resource-constrained project scheduling. *International Journal of Project Management*, *24*(1), 83-92. https://doi.org/10.1016/j.ijproman.2005.06.006

Anexos

Anexo I. Implementación del modelo

I.1. Modelo de la infraestructura de conexión a la red

Tal y como se puede observar en la figura 36, en el modelo de la infraestructura de la conexión a la red existe una dependencia lineal entre las variables de entrada -distancia a la red y potencia total de la planta- y salida -duración de la construcción de las subestaciones y duración de la construcción de la línea de conexión-:

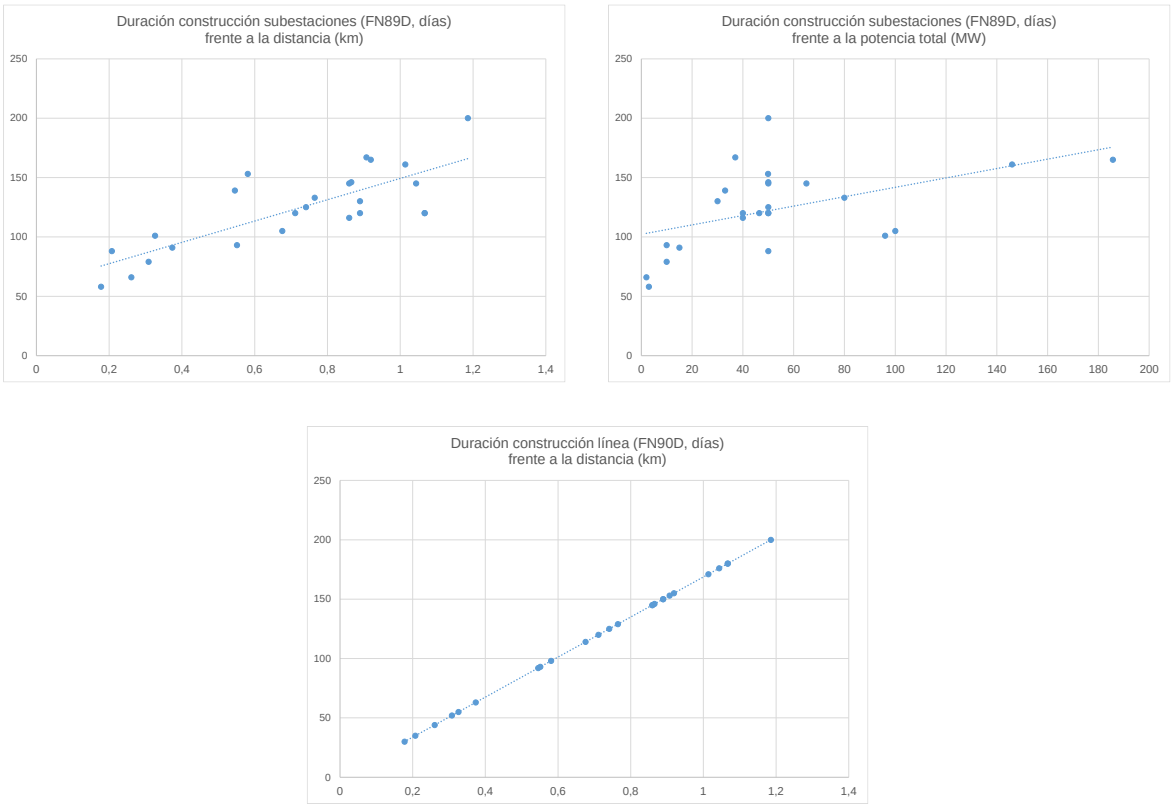


Figura 36. *Relaciones entre variables de la infraestructura de conexión a la red.*

Así, se establecen los siguientes modelos lineales:

- Duración de la construcción de la línea de conexión: depende de su longitud.
- Duración de la construcción de la planta: depende de la longitud de la línea de conexión y de la potencia total de la planta.

Ambos problemas se resuelven de forma simple con un algoritmo de regresión lineal. Los resultados así alcanzados son buenos: el modelo de la línea no contiene errores, lo que

permite observar que es un dato calculado. El modelo de la construcción de las subestaciones devuelve las métricas que se muestran en la tabla 17. Si bien aparentemente el valor de MAE parece elevado, no es así en términos relativos, dado que la duración media de esta fase es de unos 125 días, por lo que el error relativo es inferior al 13%.

En conjunto, el modelo lineal de la infraestructura de conexión consigue unas **métricas mejores que el modelo de la planta** (MAE relativo de un 6%, R^2 de 0,8400), lo que se encuentra motivado por la clara relación lineal existente entre las variables. Esto justifica la innecesariedad de su modelado a través de técnicas más complejas.

Tabla 17. Métricas del modelo para la fase de construcción de las subestaciones.

		Modelo subestaciones
MAE entrenamiento	días	16,52
MSE entrenamiento	días^2	457,97
R^2 entrenamiento	-	0,6800
MAE test	días	16,01
MSE test	días^2	418,97
R^2 test	-	0,6460
Tiempo	s	0,24

I.2. Generación automática de cronogramas

Como consecuencia del resultado de la comparativa, para la implementación práctica se ha seleccionado el modelo obtenido a través de ANFIS, debido a su mayor rendimiento a la hora de predecir salidas en registros con los que no ha sido entrenado. Ahora bien, de cara a la puesta en producción, se debe entrenar un modelo a partir de todo el conjunto de datos disponible (25 registros), para así mejorar la capacidad predictiva ante nuevas entradas de datos desconocidas.

El modelo así obtenido permite generar planificaciones de construcción de plantas fotovoltaicas de forma muy sencilla, a partir de un conjunto de 14 variables de entrada. Estas se introducen en Microsoft Excel en la pantalla que se muestra en la figura 37:

Descripción	Valor
Fecha de inicio	23/01/2023
Potencia total [MW]	49,91
Número de seguidores	1559
Número total de paneles	140280
Área [ha]	80
Perímetro [km]	3,577
Carreteras [km]	3
Zanjas [km]	23,845
Número de pilares	40030
Línea BT [m]	213.773
Línea MT [m]	23.023
Número de inversores	7
Potencia inversor [MW]	7,130
Distancia a la red [km]	0,581

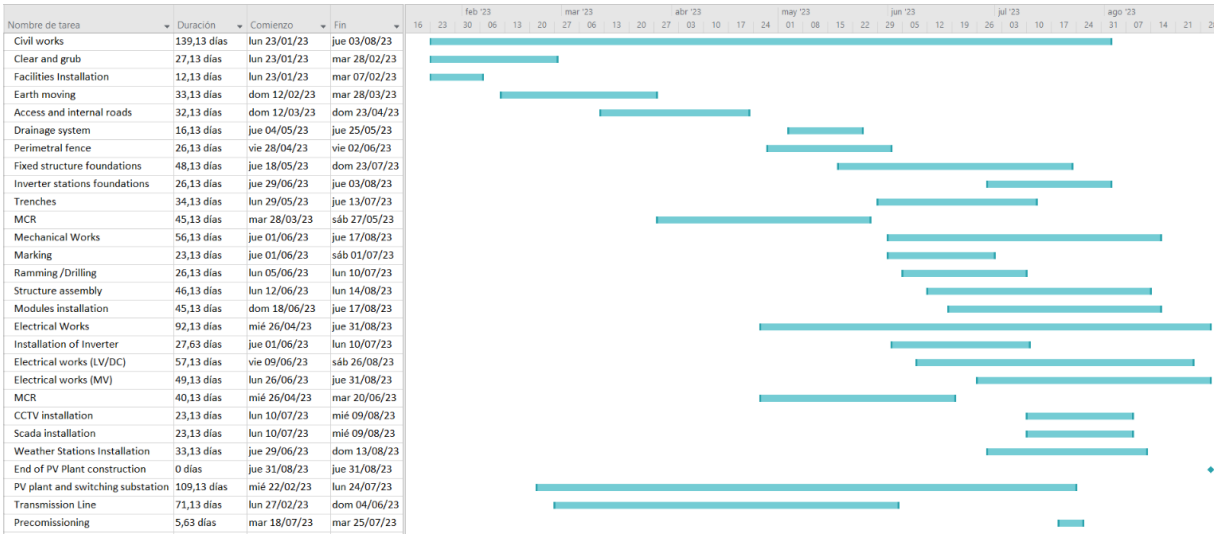
Figura 37. *Pantalla con las variables de entrada necesarias para generar un cronograma.*

Las variables de salida, que son las duraciones e intervalos entre fases, se obtienen a través de la aplicación del modelo a las variables de entrada indicadas. La fecha de inicio se emplea como referencia para generar el cronograma de construcción. A partir de ella, y de las variables de salida obtenidas, se calculan las fechas de inicio y fin de cada fase, tal y como se muestra en la figura 38:

Id	Nombre	Duración	Comienzo	Fin	Predecesoras
64	**Civil works**	**193**	23/01/2023	03/08/2023	
65	Clear and grub	36	23/01/2023	28/02/2023	
66	Facilities Installation	15	23/01/2023	07/02/2023	65
67	Earth moving	45	12/02/2023	28/03/2023	65
68	Access and internal roads	42	12/03/2023	23/04/2023	65;67
69	Drainage system	21	04/05/2023	25/05/2023	62;65;67
70	Perimetral fence	35	28/04/2023	02/06/2023	65;67;4
71	Fixed structure foundations	66	18/05/2023	23/07/2023	65;67
72	Inverter stations foundations	35	29/06/2023	03/08/2023	65;67;4
73	Trenches	45	29/05/2023	13/07/2023	65;67;4
75	**Mechanical Works**	77	01/06/2023	17/08/2023	65;67;4
76	Marking	30	01/06/2023	01/07/2023	65;67;4
77	Ramming /Drilling	35	05/06/2023	10/07/2023	49;50;65;67;4
78	Structure assembly	63	12/06/2023	14/08/2023	49;50;51;65;67;4
79	Modules installation	60	18/06/2023	17/08/2023	
80	**Electrical Works**	127	26/04/2023	31/08/2023	56;71
81	Installation of Inverter	39	01/06/2023	10/07/2023	53;54;65
82	Electrical works (LV/DC)	78	09/06/2023	26/08/2023	54;65
83	Electrical works (MV)	66	26/06/2023	31/08/2023	53;54;74
84	CCTV installation	30	10/07/2023	09/08/2023	60;85
86	Scada installation	30	10/07/2023	09/08/2023	65;67
87	Weather Stations Installation	45	29/06/2023	13/08/2023	64; 75;80
88	**End of PV Plant construction**	0	31/08/2023	31/08/2023	**58;59**
89	PV plant substation and switching substa	153	22/02/2023	24/07/2023	59
90	Transmission Line	98	27/02/2023	04/06/2023	88;89;90
91	Precomissioning	7	18/07/2023	25/07/2023	
92	**Commissioning**	56	26/07/2023	20/09/2023	**88;89;90**
93	Mechanical and civil works	10	26/07/2023	05/08/2023	88;89;90
94	Electrical Works	10	05/08/2023	15/08/2023	93;94
95	Completion	0	15/08/2023	15/08/2023	95
96	Hot commisioning	10	10/08/2023	20/08/2023	96
97	Check and review works	10	15/08/2023	25/08/2023	97
98	Commercial Operation Date	0	20/08/2023	20/08/2023	98
99	Performance Test	31	20/08/2023	20/09/2023	99
100	Provisional Acceptance	0	20/09/2023	20/09/2023	

Figura 38. *Ejemplo de cronograma generado a partir del modelo.*

Este cronograma puede exportarse a Microsoft Project, como se muestra en la figura 39:



Nombre de tarea	Duración	Comienzo	Fin
Civil works	139,13 días	lun 23/01/23	jue 03/08/23
Clear and grub	27,13 días	lun 23/01/23	mar 28/02/23
Facilities Installation	12,13 días	lun 23/01/23	mar 07/02/23
Earth moving	33,13 días	dom 12/02/23	mar 28/03/23
Access and internal roads	32,13 días	dom 12/03/23	dom 23/04/23
Drainage system	16,13 días	jue 04/05/23	jue 25/05/23
Perimetral fence	26,13 días	vie 28/04/23	vie 02/06/23
Fixed structure foundations	48,13 días	jue 18/05/23	dom 23/07/23
Inverter stations foundations	26,13 días	jue 29/06/23	jue 03/08/23
Trenches	34,13 días	lun 29/05/23	jue 13/07/23
MCR	45,13 días	mar 28/03/23	sáb 27/05/23
Mechanical Works	56,13 días	jue 01/06/23	jue 17/08/23
Marking	23,13 días	jue 01/06/23	sáb 01/07/23
Ramming /Drilling	26,13 días	lun 05/06/23	lun 10/07/23
Structure assembly	46,13 días	lun 12/06/23	lun 14/08/23
Modules installation	45,13 días	dom 18/06/23	jue 17/08/23
Electrical Works	92,13 días	mié 26/04/23	jue 31/08/23
Installation of Inverter	27,63 días	jue 01/06/23	lun 10/07/23
Electrical works (LV/DC)	57,13 días	vie 09/06/23	sáb 26/08/23
Electrical works (MV)	49,13 días	lun 26/06/23	jue 31/08/23
MCR	40,13 días	mié 26/04/23	mar 20/06/23
CCTV installation	23,13 días	lun 10/07/23	mié 09/08/23
Scada installation	23,13 días	lun 10/07/23	mié 09/08/23
Weather Stations Installation	33,13 días	jue 29/06/23	dom 13/08/23
End of PV Plant construction	0 días	jue 31/08/23	jue 31/08/23
PV plant and switching substation	109,13 días	mié 22/02/23	lun 24/07/23
Transmission Line	71,13 días	lun 27/02/23	dom 04/06/23
Precomissioning	5,63 días	mar 18/07/23	mar 25/07/23

Figura 39**.** *Cronograma exportado a Microsoft Project.*

Desde esta herramienta es posible continuar con la gestión del proyecto, por ejemplo, incorporando condiciones como las tareas predecesoras, o incluyendo nuevas fases, como la instalación de la sala de medida (MCR), tal y como se muestra en el ejemplo de la figura 40:

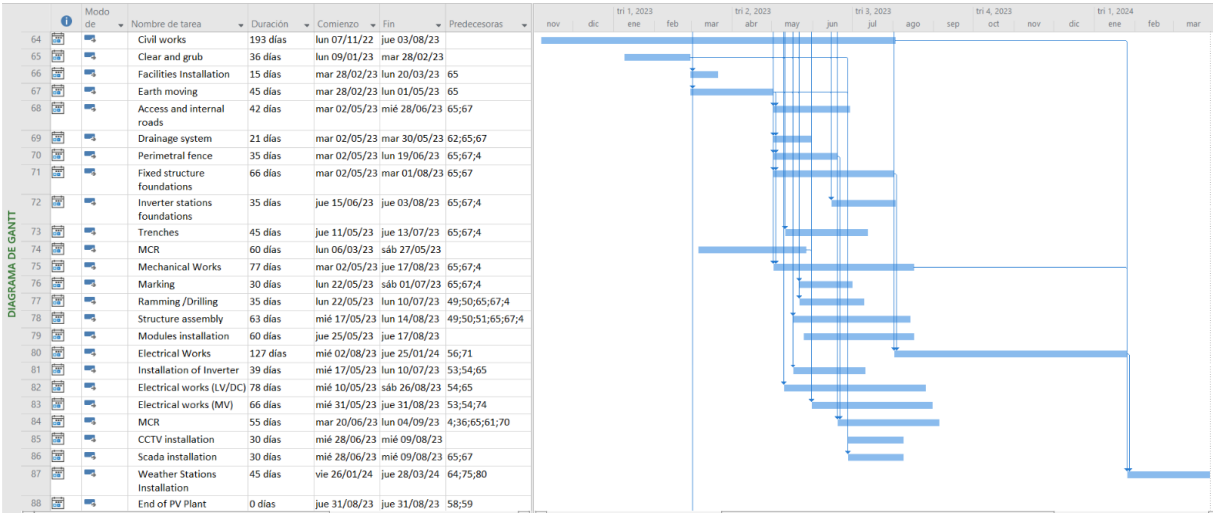


DIAGRAMA DE GANTT

	Modo de	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
64		Civil works	193 días	lun 07/11/22	jue 03/08/23	
65		Clear and grub	36 días	lun 09/01/23	mar 28/02/23	
66		Facilities Installation	15 días	mar 28/02/23	lun 20/03/23	65
67		Earth moving	45 días	mar 28/02/23	lun 01/05/23	65
68		Access and internal roads	42 días	mar 02/05/23	mié 28/06/23	65;67
69		Drainage system	21 días	mar 02/05/23	mar 30/05/23	62;65;67
70		Perimetral fence	35 días	mar 02/05/23	lun 19/06/23	65;67;4
71		Fixed structure foundations	66 días	mar 02/05/23	mar 01/08/23	65;67
72		Inverter stations foundations	35 días	jue 15/06/23	jue 03/08/23	65;67;4
73		Trenches	45 días	jue 11/05/23	jue 13/07/23	65;67;4
74		MCR	60 días	lun 06/03/23	sáb 27/05/23	
75		Mechanical Works	77 días	mar 02/05/23	jue 17/08/23	65;67;4
76		Marking	30 días	lun 22/05/23	sáb 01/07/23	65;67;4
77		Ramming /Drilling	35 días	lun 22/05/23	lun 10/07/23	49;50;65;67;4
78		Structure assembly	63 días	mié 17/05/23	lun 14/08/23	49;50;51;65;67;4
79		Modules installation	60 días	jue 25/05/23	jue 17/08/23	
80		Electrical Works	127 días	mié 02/08/23	jue 25/01/24	56;71
81		Installation of Inverter	39 días	mié 17/05/23	lun 10/07/23	53;54;65
82		Electrical works (LV/DC)	78 días	mié 10/05/23	sáb 26/08/23	54;65
83		Electrical works (MV)	66 días	mié 31/05/23	jue 31/08/23	53;54;74
84		MCR	55 días	mar 20/06/23	lun 04/09/23	4;36;65;61;70
85		CCTV installation	30 días	mié 28/06/23	mié 09/08/23	
86		Scada installation	30 días	mié 28/06/23	mié 09/08/23	65;67
87		Weather Stations Installation	45 días	vie 26/01/24	jue 28/03/24	64;75;80
88		End of PV Plant	0 días	jue 31/08/23	jue 31/08/23	58;59

Figura 40. *Cronograma en Microsoft Project con tareas predecesoras.*

Anexo II. Tratamiento de datos

Este anexo detalla la selección de las variables de entrada al modelo y explica el proceso seguido para la depuración de los valores anómalos.

II.1. Selección de variables de entrada

El conjunto de datos inicial se compone de 25 registros y 85 variables. Estas se dividen en 23 variables de entrada y 62 de salida, tal y como muestra la tabla 3 del capítulo 4. A continuación, se explicará cuáles han sido las variables finalmente seleccionadas y el criterio empleado para su elección.

Una revisión en profundidad de las variables y la relación de estas con el modelo permite prescindir de las siguientes:

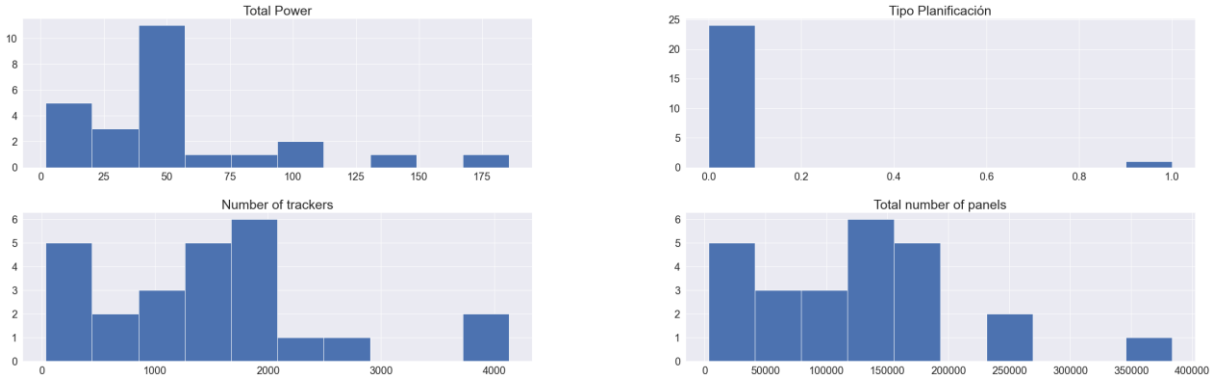
- **‘Fecha inicio’** y ‘**Fecha fin’**: no se trabajará con fechas absolutas, sino con duraciones de fases e intervalos entre inicio de fases.
- **‘Duración total’**: es un dato calculado, en concreto, la duración total del proyecto, incluyendo fases previas a la construcción.
- **‘Ratio MW/días deseado’**: es un dato calculado como el cociente entre la potencia total, 'Total Power', y 'Total duration'.
- **‘Nº paneles/seguidor’**: es un dato calculado como el cociente entre dos datos de los que se dispone: el número total de paneles y el número de seguidores.
- **‘Potencia unitaria’** por panel: es el cociente entre la potencia total y el número total de paneles. Se observará, además, en el análisis de correlaciones, que aquella variable con mayor interdependencia de esta, la ‘Obra eléctrica BT y CC’, tiene una correlación todavía mayor con la potencia total y el número total de paneles, por lo que resulta más significativa para el modelado.
- **‘Nº cimientos’**: coincide con 'Número de inversores'. Una vez observadas las correlaciones, se optará por un modelo que integrará conjuntamente ambas variables, por lo que una de ellas resulta redundante.
- **‘Nº de apoyos’**: es un dato calculado a partir de ‘Distancia’, resultando un apoyo por cada 50 m. Es preferible mantener la ‘Distancia’ ya que el 'Nº de apoyos’ aporta una información menos precisa, que ha sido redondeada a un número entero.

La selección efectuada reduce a 15 el número de variables de entrada. El análisis de su histograma (figura 41) permite prescindir, asimismo, de las siguientes variables:

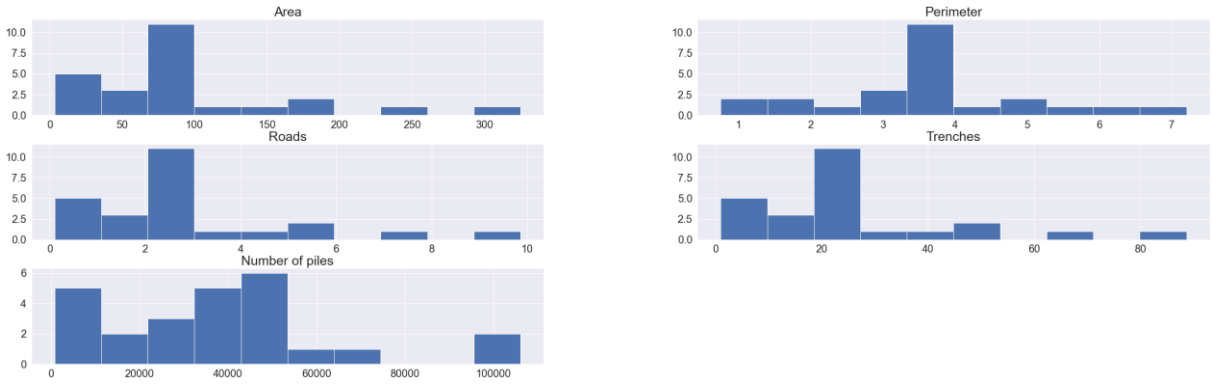
- **‘Tipo planificación’**: entero que, cuanto mayor sea su valor, más ajustado será el cronograma. De todo el conjunto de datos, 24 registros tienen valor 0 y solamente un registro tiene valor 1. No se aprecian diferencias sensibles en el cronograma de este registro con respecto al resto de plantas, por lo que se decide prescindir de esta variable.
- **‘Tensión’**: todas las plantas del conjunto de datos se conectan a una tensión de red de 132 kV. Así, esta variable no aporta información al modelo, por lo que se prescindirá de ella.

Tras esta selección, el número de variables de entrada se reduce a las **13 variables** mostradas en la tabla 4.

Variables generales



Variables obra civil



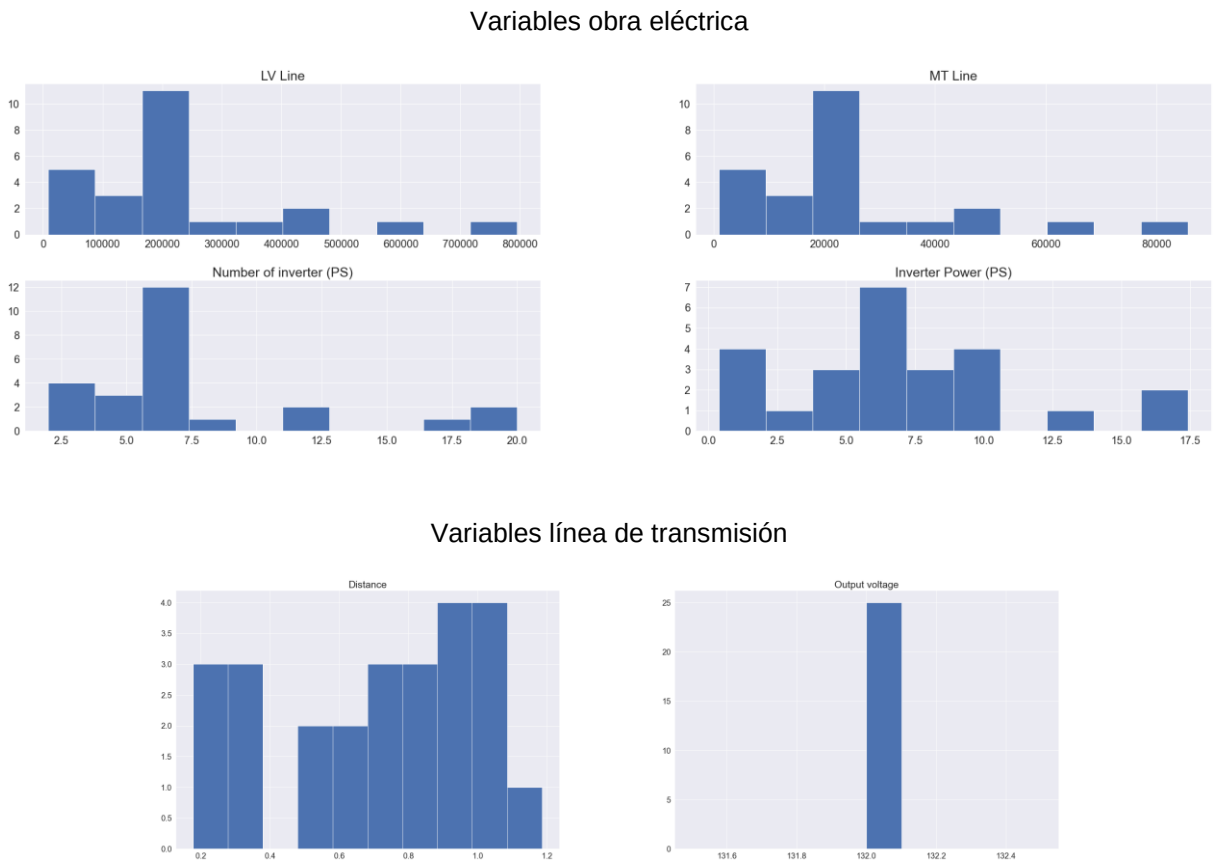
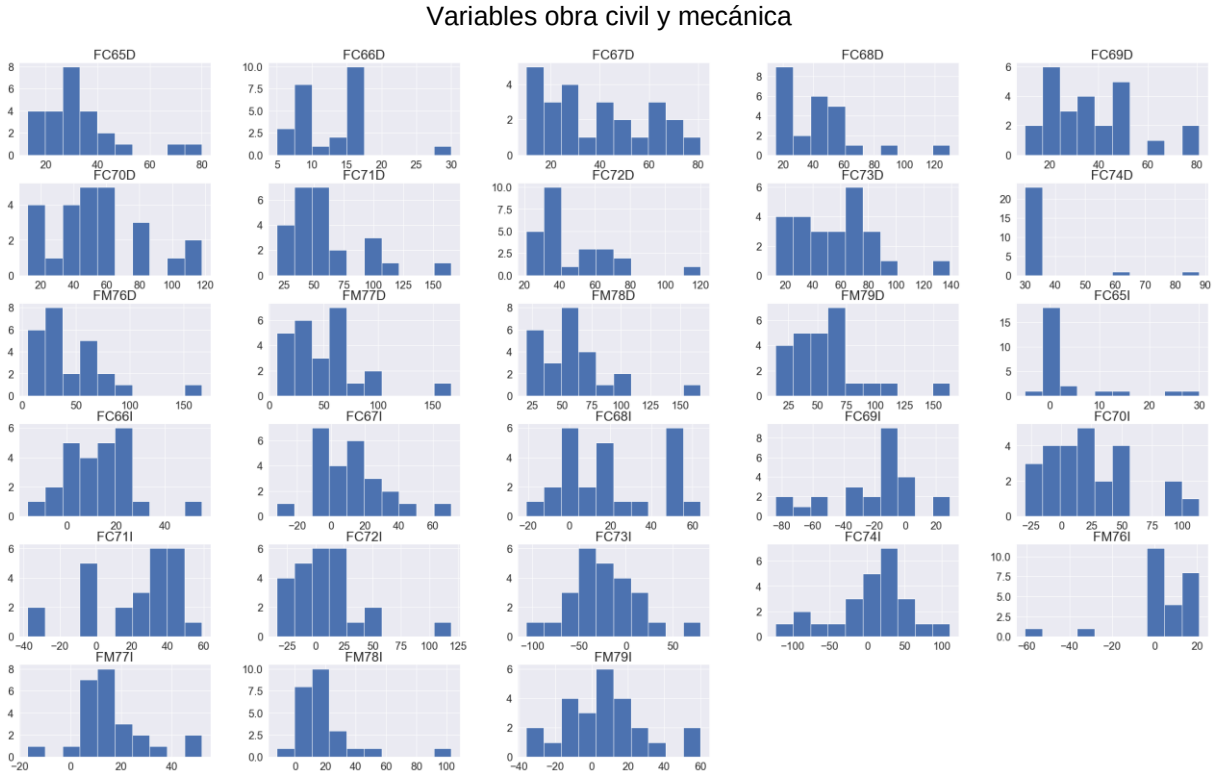
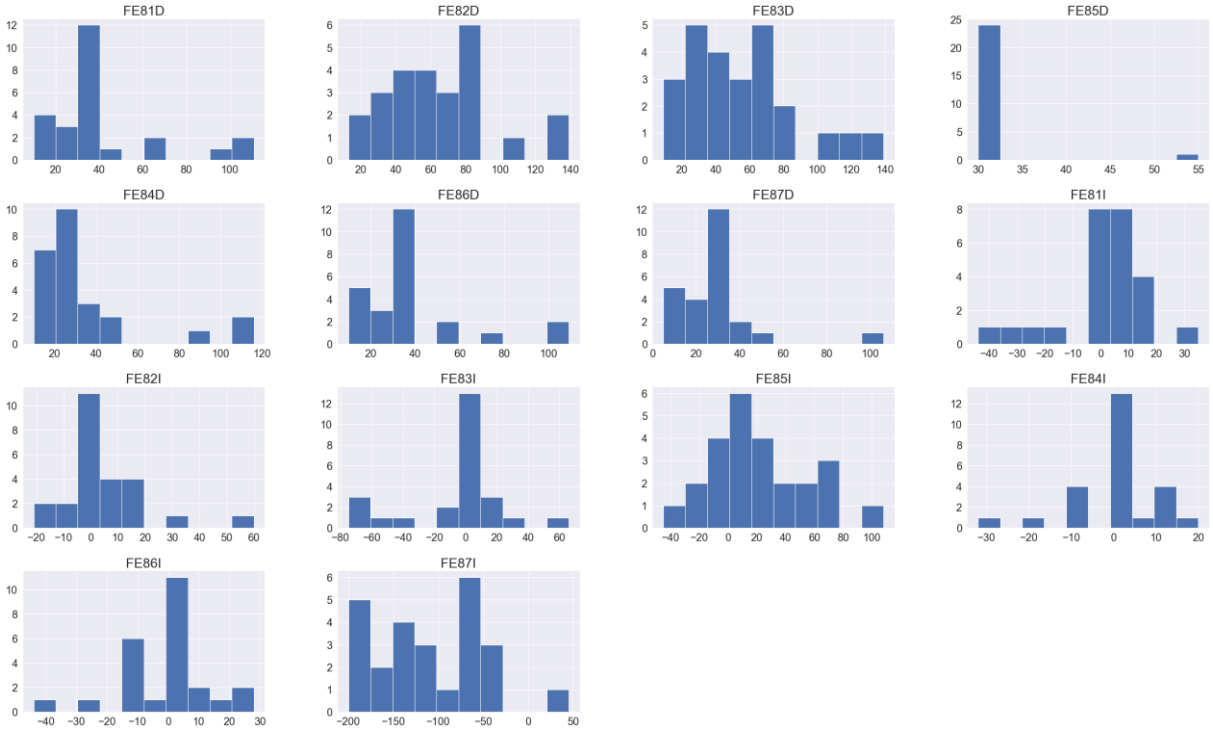


Figura 41. *Histograma de las variables de entrada.*

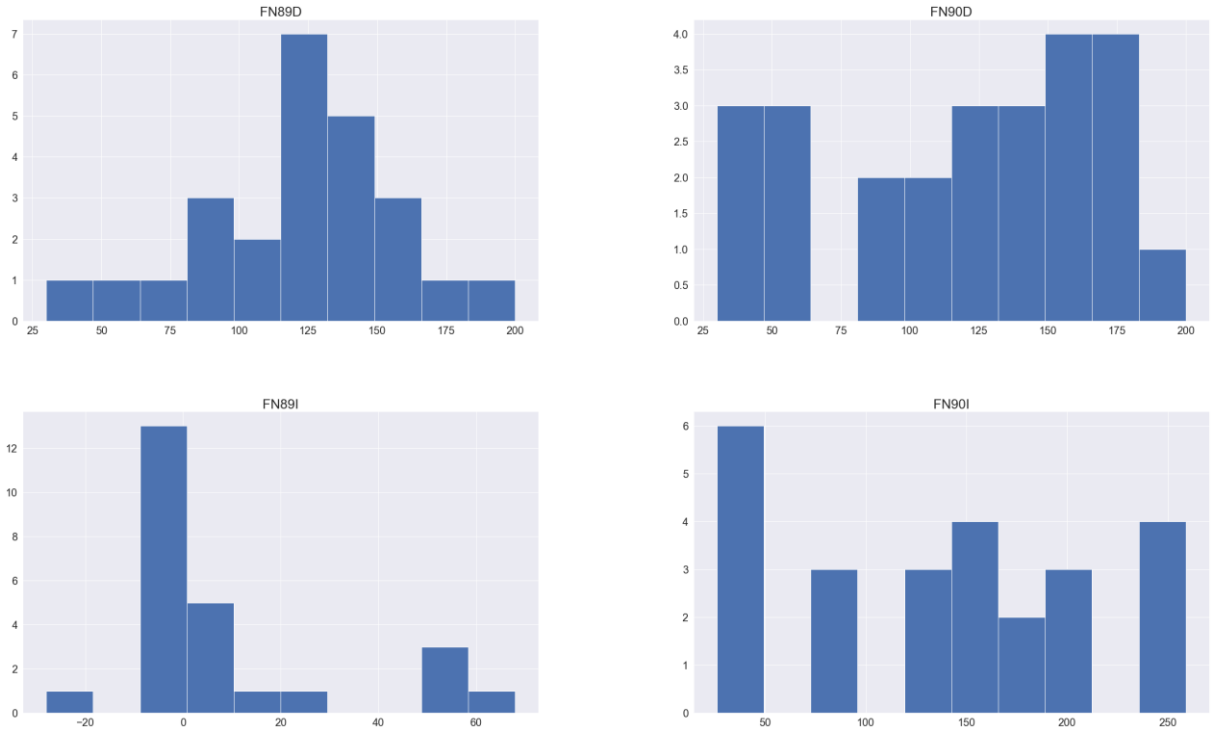
A continuación, se muestra en la figura 42 el histograma de las variables de salida, que ha sido utilizado para el análisis de datos descrito en el apartado 4.2.1.



Variables obra eléctrica



Variables línea de transmisión



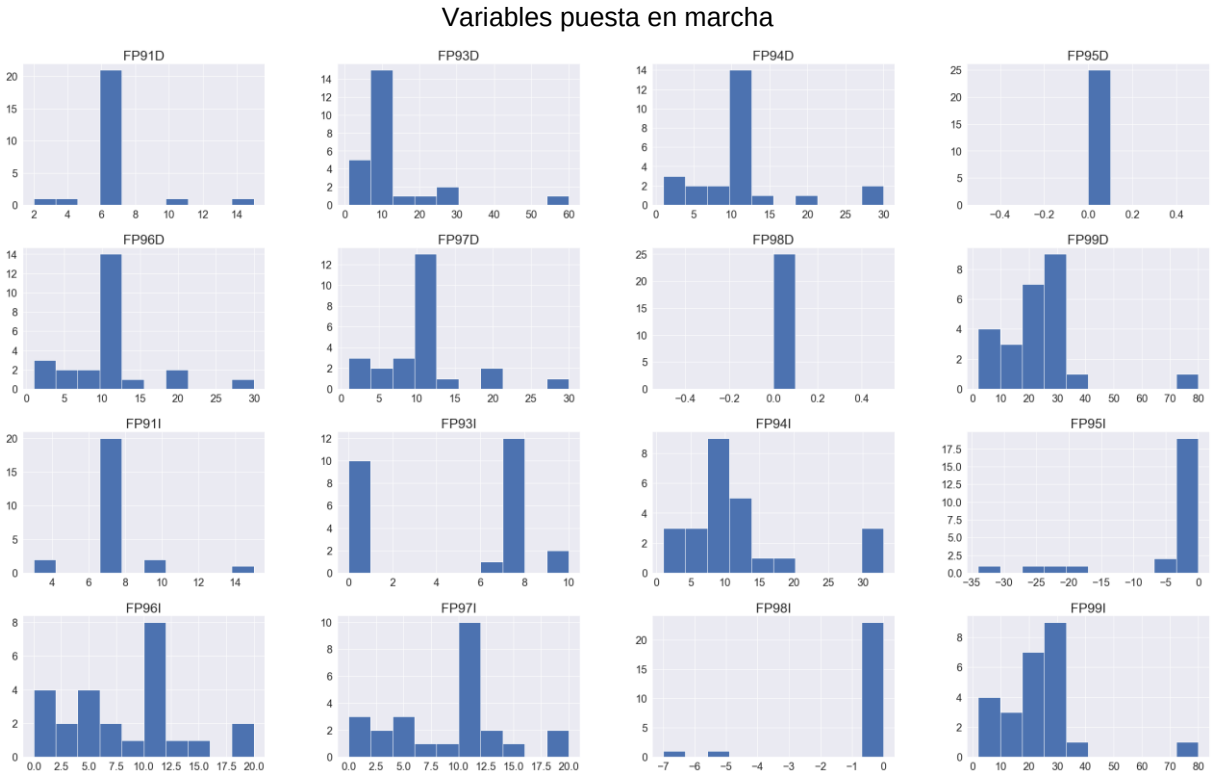


Figura 42. *Histograma de las variables de salida.*

Por ultimo, se observa que las variables que representan un número entero (como p.ej. 'Nº de pilares') están representadas con valores con decimales, en coma flotante, por lo que se redondean al número entero más próximo.

II.2. Depuración de valores anómalos

Este apartado desarrolla con amplitud la fase de depuración de valores anómalos. Los valores inicialmente detectados como tales figuran en la tabla 18.

Tal y como se mencionó en el capítulo 4, las alternativas para resolver estos problemas son:

- **Incorporar las variables de entrada** que puedan justificar los valores anómalos. Por ejemplo, la duración de la fase de 'Limpieza' (*Clear and grub*, FC65D), correspondiente a la limpieza y excavaciones previas del terreno, puede variar sensiblemente en función del desbroce que pueda ser necesario, aun para la misma superficie de parcela. En los datos de que se dispone, no existe ninguna variable de entrada que refleje el grado de desbroce necesario, por lo que el modelo no podrá incorporar estas variaciones, que resultarán en errores de predicción. Otro ejemplo similar se puede encontrar con variables que reflejen el método constructivo a emplear: el tender cable directamente enterrado en lugar de bajo tubo, condiciona que el inicio de las 'Zanjas'

no pueda realizarse hasta que se ejecute la 'Obra eléctrica BT y CC' y la 'Obra eléctrica MT'. Esto impacta significativamente en el intervalo entre fases FC72I.

- **Sustituir los valores anómalos por un valor extrapolado**. Tal y como se muestra en la tabla 18 , se emplean los métodos de *k*-vecinos e interpolación lineal -cuando el anterior método no es robusto-, para depurar los valores anómalos, en función de las variables de entrada más representativas, empleando un conocimiento básico del dominio.

Las limitaciones de este estudio a la hora de incorporar nuevas variables implican que la alternativa adoptada haya sido esta segunda.

Variables de entrada

Se observa que los posibles valores anómalos superiores corresponden a las plantas de mayor potencia, en las que tanto las variables de entrada (dimensiones de la planta) como las de salida (plazos de construcción) son, lógicamente, superiores. Una vez analizados los valores, no se considera que ninguno de ellos resulte anómalo.

De modo análogo, los posibles valores anómalos inferiores corresponden a las plantas de menor potencia, lo que se encuentra dentro de lo esperado, según el mismo razonamiento expuesto en el párrafo anterior.

Variables de salida

Se observan valores anómalos en las **duraciones de las fases** (Fx##D) que no pueden ser explicados por las variables de entrada del modelo, y que introducen ruido. Este análisis sirve para localizar incluso algún valor erróneo, como el de la planta con índice 3, en la que los trabajos de tipo civil y mecánico finalizan el 12/8/2019, mientras que la planta ya había sido puesta en servicio en fecha 2/8/2019.

Asimismo, se observan valores anómalos en los **intervalos entre inicio de fases** (Fx##I). Las dependencias de estas variables son complejas, pues dependen de la coordinación entre actividades. De modo general, no es factible realizar un análisis directo sobre si son explicables por las variables de entrada. Sí resulta posible observar el adelanto de alguna fase, sin encontrarse justificado por las variables de entrada, como en el caso del inicio de la fase de 'Cierre perimetral' (dado por FC69I) en la planta 8. Esta fase depende de otra fase anterior a la construcción y que no forma parte del modelo. Esta anomalía se puede normalizar, sin impactar al cronograma, posponiendo el inicio de dicha fase y adelantando en la misma medida la fase de 'Zanjas' (FC72I). Igualmente se pueden resolver los valores anómalos en los intervalos que definen el inicio de 'Movimiento de tierras' (dado por el valor entre fechas de inicio de su tarea previa FC66I) y 'Accesos y vías interiores' (FC67I), asignando un valor de aquellas plantas *k*-vecinas (*k* variable, en función de la similitud de los

ejemplos) de área más cercana. El resto de valores anómalos en FC69I se han resuelto con las plantas *k*-vecinas considerando la variable de entrada 'Perímetro'. Esto ha permitido, asimismo, mejorar la calidad del dataset ya que, indirectamente, ha corregido alguna anomalía entre las dependencias de fases (11 de los 25 cronogramas no respetaban las dependencias entre las fases de obra civil).

Por último, los valores anómalos entre FM76I y FE86I, ambos incluidos, se llevan a los límites inferior y superior dados por $\bar{x} \pm 1{,}5[Q_3(x) - Q_1(x)]$, comprobándose que los cronogramas no varían su duración de forma sensible ni se afecta a las dependencias entre fases.

Tabla 18. Valores de salida anómalos y su depuración.

Código fase	Descripción fase/fin intervalo	Planta	Valor inicial	Valor depurado	Método depuración	Variable referencia
FC65D	*Limpieza del terreno*	15	80	27	k-vecinos (k=5)	*Área*
FC68D	*Accesos y vías interiores*	16	90	22	k-vecinos (k=5)	*Carreteras*
FC69D	*Sistema de drenaje*	15	80	33	k-vecinos (k=5)	*Área*
FC70D	*Cierre perimetral*	11, 12, 22	118, 12, 107	71, 66, 61	Interpolación lineal	*Perímetro*
FC71D	*Pilares de estructuras fijas*	16	103	42	k-vecinos (k=5)	*Número de pilares*
		22	111	54	k-vecinos (k=5)	*Número de pilares*
FE83D	*Obra eléctrica (MT)*	4	109	67	k-vecinos (k=3)	*Línea MT*
FE84D	*Instalación CCTV*	4	116	30	k-vecinos (k=6)	*Área*
FE86D	*Instalación SCADA*	4, 15, 13, 17	109, 70, 12, 59	31, 28, 28, 31	k-vecinos (k=5, 4, 4, 5)	*Potencia total*
FP93D	*Puesta en marcha: obra civil y mecánica*	3	60	10	k-vecinos (k=11)	*Potencia total*
FP99D	*Aceptación provisional*	1	80	30	k-vecinos (k=2)	*Potencia total*
FC72I	*Zanjas*	8	119	21	k-vecinos (k=3)	*Zanjas, Número de pilares*
FC69I	*Cierre perimetral*	8	-84	14	Mantener duración cronograma	*FC72I*
FC65I	*Instalación de infraestructuras*	4, 11, 16,17, 20,23, 24	5, 2, 12, -5, 23, 14, 30	0	Moda	No aplica
FC66I	*Movimiento de tierras*	21, 24	55, -16	14, 15	k-vecinos (k=3)	*Área*
FC67I	*Accesos y vías interiores*	3, 6, 9	-32, 48, 71	11, 11, 11	k-vecinos (k=7, k=7, k=3)	*Área*
FC69I	*Cierre perimetral*	8, 10, 12, 16, 21	14, 29, -69, -78, -60	-16, -3, -20, -3, -11	k-vecinos (k=4, 4, 10, 4, 4)	*Perímetro*
FM77I	*Montaje de estructuras*	18	-17	11	k-vecinos (k=4)	*Paneles solares: nº total*
FM79I	*Instalación de inversores*	18	-3	34	Mantener duración cronograma	No aplica

Por último, se prescinde de las variables de salida relacionadas con la ‘Sala de medida’, tanto en obra civil como en obra eléctrica (fases 74 y 85), por los siguientes dos motivos:

1. Se trata de una instalación que no está presente en todas las plantas.
2. Su ejecución es, en gran medida, asíncrona con respecto al resto de las fases, por lo que su modelado presenta una gran variabilidad e introduce un gran ruido.

Este último hecho se ha observado, además, en el análisis de correlaciones.

La figura 43 muestra la dispersión del intervalo entre fechas definido a partir del inicio de la instalación de la sala de medida, frente a aquellas que se identifican como variables de mayor importancia y cómo, dentro de un mismo rango de entrada, la salida puede adoptar valores en extremos opuestos (hecho que, analizado en profundidad, tampoco se ve justificado por ninguna otra variable de entrada).

La eliminación de estas variables del modelo es simple: es suficiente añadir al intervalo de la fase anterior el valor del intervalo de esta fase.

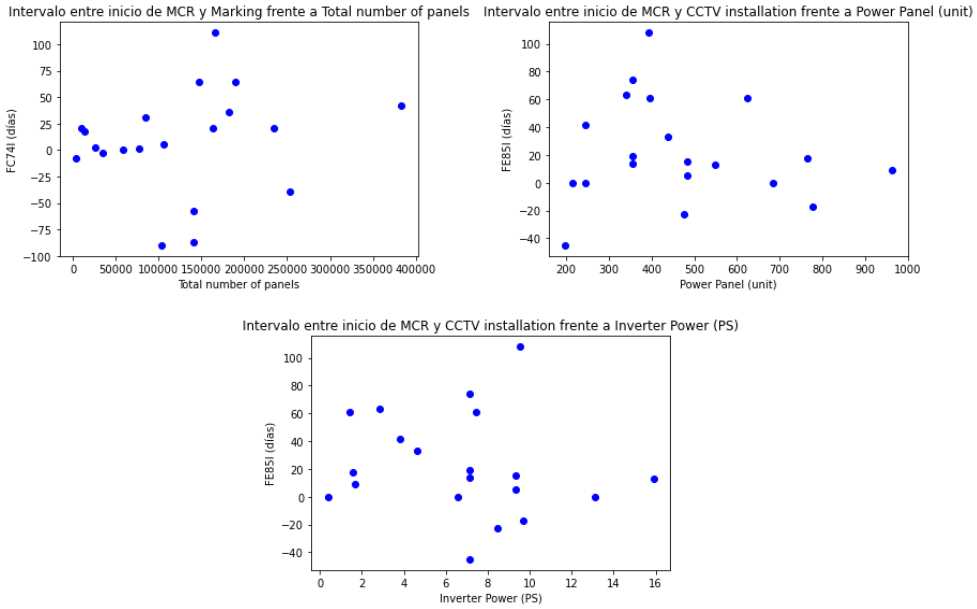


Figura 43. *Intervalo dado por la sala de medida y sus variables de mayor importancia.*

Asimismo, se resuelven inconsistencias en duraciones e intervalos que no se corresponden con las fechas definidas en los cronogramas, así como valores que son muy divergentes aun cuando las características de la planta son muy similares, o incluso idénticas.

Anexo III. Pruebas

En esta sección se tratan los resultados obtenidos por dos algoritmos cuyo rendimiento ha sido notablemente inferior al resto de los empleados en la comparativa.

III.I. Árbol de regresión multivariable

La primera aproximación al problema se ha realizado con un único árbol de regresión "multivariable", en el que cada hoja devuelve de forma conjunta un vector con todas las variables de salida. En este primer test se han empleado las variables de entrada y salida de obra civil, por ser las que mayor correlación muestran. El resultado se muestra en la tabla 19 y resulta muy inferior al dado por la línea base, siendo, por tanto, poco satisfactorio.

Tabla 19. Rendimiento del árbol de regresión multivariable.

		Línea base	Árbol multivariable
MAE entrenamiento	días	8,48	10,15
MSE entrenamiento	días^2	259,12	224,03
R^2 test	-	0,0000	0,3790
MAE test	días	9,17	54,67
MSE test	días^2	275,86	857,07
R^2 test	-	-0,4201	-2,4818
Tiempo	s	0,0000	0,50

III.2. Optimización local de enjambre de partículas

Tal y como se ha indicado en el apartado 5.2, esta técnica es una evolución de la optimización de enjambre de partículas "global", en la que se definen varios subenjambres a fin de evitar caer en un único mínimo local. Este algoritmo precisa definir, de forma adicional al algoritmo global, los siguientes hiperparámetros:

- número de partículas en cada subenjambre: se han realizado pruebas con un número de 2 hasta 200 partículas por subenjambre.
- Norma para evaluar la distancia entre partículas: se ha empleado la norma 1 y la norma 2 (euclídea).

Los resultados de este algoritmo mejoran a medida que el número de partículas de cada "subenjambre" tiende a uno, lo que pone de relieve el superior rendimiento de la estrategia de optimización global para este caso. Las **métricas** obtenidas en las pruebas con mejores

resultados figuran en la tabla 20. En ella puede observarse el muy bajo rendimiento obtenido con esta técnica, muy por debajo de la línea base.

Tabla 20. Rendimiento de optimización de enjambre de partículas local.

		Línea base	PSO local
MAE entrenamiento	días	8,48	74,63
MSE entrenamiento	días^2	259,12	8405,30
R^2 test	-	0,0000	-46,5417
MAE test	días	9,17	86,78
MSE test	días^2	275,86	13053,44
R^2 test	-	-0,4201	-18,3500
Tiempo	s	0,0000	571,7867

Anexo IV. Artículo de investigación

Técnicas de inteligencia artificial aplicadas a la planificación de construcción de plantas fotovoltaicas

Manuel Ángel López Ferreiro

Universidad Internacional de la Rioja, Logroño (España)

14 de julio de 2022

Resumen

La generación fotovoltaica es una de las tecnologías clave para alcanzar los retos de transición energética. Así, en los últimos años se ha observado un incremento en la ejecución de proyectos de grandes plantas fotovoltaicas. La principal dificultad para cumplir con el plazo de construcción es la elaboración de una planificación adecuada. Una herramienta que genere cronogramas de forma automática puede resultar de gran ayuda a la hora de establecer planificaciones iniciales de referencia. A tal fin, este trabajo compara cinco técnicas de inteligencia artificial, sobre un conjunto de datos formado por 25 planificaciones. La evaluación de los resultados obtenidos sobre datos de test demuestra que ANFIS es la técnica que obtiene mejor rendimiento en todas las métricas de error, si bien conlleva un alto coste computacional. El modelo así obtenido consigue generar un cronograma completo de construcción con un error de un 8% sobre la duración total.

Palabras Clave

ANFIS, Cronograma de Construcción, Inteligencia Artificial, Neuroevolución, Planificación de Proyectos.

I. Introducción

La **generación fotovoltaica** es una de tecnologías clave en el futuro de la producción de energía renovable. Supone en torno a un 25% de la potencia instalada en energías renovables a nivel mundial [1] y se trata de la tecnología que está experimentando un crecimiento más rápido en valor absoluto, con un incremento a nivel mundial de un 21% en el año 2020 [2].

Un factor clave para alcanzar estos objetivos en el plazo fijado es una adecuada **planificación de los proyectos de construcción** de estas plantas. Este estudio pretende sistematizar la elaboración de cronogramas de construcción de forma precisa, minimizar los sesgos y facilitar la toma de decisiones. Para ello, este trabajo compara cinco técnicas de inteligencia artificial, sobre un conjunto de datos formado por 25 planificaciones. El resultado esperado de dicha comparativa es poder determinar una técnica que permita obtener resultados satisfactorios a la hora de elaborar cronogramas de construcción de forma automática.

Para ello, se partirá de una revisión del estado del arte, con la que se determinarán los modelos que formarán parte de la comparativa. Estos modelos se elaborarán primeramente desde un punto de vista conceptual. Tras el análisis del conjunto de datos y su preparación, se implementarán los modelos en software, para su entrenamiento con un conjunto de datos diferente del que servirá para calcular las métricas de rendimiento. Los resultados obtenidos serán objeto de validación y análisis, para determinar el modelo óptimo para la generación automática de cronogramas.

II. Estado del Arte

Existen numerosas técnicas de inteligencia artificial que han sido aplicadas a la **gestión de proyectos.** Chua et. al [3] utilizaron una red neuronal para predecir el éxito de un proyecto a partir de ocho factores clave. Georgy et al. [4] emplearon un sistema neurodifuso para predecir el grado de desempeño en un proyecto de construcción. Este tipo de sistemas, como ANFIS, cuentan con muy diversas aplicaciones, como la estimación del coste de proyectos, la predicción de la resistencia de materiales de construcción, el estado de mantenimiento de canalizaciones o la predicción de consumo eléctrico [5], [6]. La introducción de algoritmos genéticos en estos sistemas ha permitido a Cheng et al. predecir el éxito de un proyecto [7], determinar el coste de un proyecto [8] o evaluar subcontratistas [9]. Los algoritmos bioinspirados han servido para resolver muy diversos problemas. Los basados en colonias de hormigas [10] han demostrado experimentalmente que son capaces de mejorar el valor actual neto de un proyecto y han resuelto problemas de enrutamiento en redes de telecomunicaciones, transporte y distribución [11]. Otros, como evolución diferencial y optimización de enjambre de partículas (PSO) han servido para resolver problemas de almacenaje de mercancías y enrutamiento de flotas de vehículos [12]. Técnicas como los mapas cognitivos borrosos, algoritmos bayesianos, máquinas de vector de soporte o *K-means* han sido también usados para la identificación de factores críticos para el éxito de un proyecto o para la predicción de su éxito [13].

Dentro del universo de problemas de planificación, la **elaboración automática de cronogramas de construcción** ha sido objeto de investigación desde la década de 1960. Una primera clasificación de las posibles soluciones es la siguiente:

1. **Modelos generales**, que contienen información sobre las características de los proyectos.
2. **Modelos basados en casos** o *case-base reasoning* (CBR). Consiste en la elaboración de nuevos cronogramas a partir de ejemplos históricos con características similares.

Faghihi et al. [14] realizan una revisión de las técnicas empleadas para la elaboración de cronogramas, distinguiendo entre los modelos generales y los basados en casos. Fazel et al. [15] presentan un conjunto exhaustivo de las técnicas empleadas durante los últimos 40 años, habiendo investigado más de 540 artículos. Es de reseñar asimismo el notable conjunto de

metaheurísticas que presentan Pellerin et al. [6] para resolver los RCPSP (*Resource-Constrained Project Scheduling Problem*) o problema de planificación de proyectos con restricción de recursos, un problema que, si bien no se encuentra basado en casos, ha sido objeto de revisión en este trabajo, dada la amplia investigación existente sobre él.

De modo general, las diferentes técnicas se subdividen en dos grupos: **técnicas básicas**, que contemplan un único algoritmo de inteligencia artificial, y **técnicas híbridas**, que combinan varias técnicas básicas. A continuación, se presentan algunos de sus ejemplos más relevantes.

A. Técnicas básicas

En este apartado se relacionan las técnicas básicas que han sido utilizadas en la elaboración de cronogramas, bien de forma aislada o en combinación con otras técnicas.

1. Redes neuronales artificiales

Las redes neuronales imitan el principio de funcionamiento de las neuronas del cerebro humano, a través de un conjunto de nodos, llamados neuronas, interconectados entre sí, y que intercambian información. Estas neuronas se combinan en una serie de capas, dando lugar a la red neuronal. Una red neuronal aprende a interpretar los conjuntos de datos con que se entrena y, con ello, predecir el comportamiento ante situaciones similares a las que ha sido entrenada. En 1997 fue cuando, por primera vez, se aplicó esta técnica a la elaboración de cronogramas de construcción [16]. Desde entonces, se han producido numerosos avances: el método CONSCOM, el generador de estructuras de desglose de trabajo (*Work Breakdown Structure* o WBS) [17] o *Single-machine scheduling* [18], que elabora un cronograma para cada máquina, o la predicción del tiempo necesario para rematar un proyecto, con modelos *long short-term memory* [19].

2. Lógica borrosa

La lógica borrosa nació como una herramienta para describir la incertidumbre y la inexactitud del conocimiento, imitando el modo en el que el cerebro humano toma decisiones frente a la incertidumbre o la vaguedad, permitiendo modelar sistemas complejos o difíciles de definir [20]. Los modelos de lógica borrosa se componen de un "fuzzificador", que convierte las entradas en variables borrosas, una base de reglas borrosas, un motor de inferencia a partir de dichas reglas y un "defuzzificador" que convierte las variables borrosas en salidas. La lógica borrosa se expresa a través de una serie de "funciones de pertenencia", que permiten representar la ambigüedad y, con posterioridad, establecer relaciones lingüísticas entre las variables. Dadas sus limitaciones, por su incapacidad para aprender de los datos y su dependencia del conocimiento experto y del contexto, su aplicación a la planificación de proyectos se realiza en conjunto con otras técnicas, colaborando a salvar las desventajas de estas.

3. Árboles de decisión

Los árboles de decisión permiten establecer una salida, dado un conjunto de atributos de entrada y un esquema en forma de árbol, en el que se va avanzando hacia la salida al tomar una decisión en cada nodo del árbol en función del valor de los citados atributos. Los árboles de decisión permiten elaborar modelos sencillos y explicables. El rendimiento de un árbol aislado puede ser mejorado a través de técnicas basadas en un conjunto de árboles o *ensemble*, como *boosting* y *bagging*. En el ámbito de la planificación de proyectos, han sido empleados para tratar con la incerteza de los problemas de planificación [21] así como en la obtención de reglas de prioridad en el RCPSP [22].

4. Algoritmos evolutivos

Los algoritmos evolutivos pretenden simular la evolución de las especies. Su principio de funcionamiento se basa en los tres siguientes pasos: en primer lugar, se parte de una población de individuos, generada de forma aleatoria. Cada uno de ellos representa una posible solución al problema. A continuación, se evalúa el desempeño de cada individuo como posible solución al problema. Por último, se genera una nueva población, por evolución de la preexistente. Este proceso se lleva a cabo hasta que alguno de los individuos cumple las condiciones fijadas para la resolución del problema [12]. Han sido ampliamente aplicados en el área de la elaboración de cronogramas de proyectos, con los mejores resultados en la resolución de problemas como RCPSP. Se describirán dos de los algoritmos utilizados para ello: algoritmos genéticos y evolución diferencial. Los primeros resultan más aptos para problemas de optimización discretos, mientras que los segundos son preferibles para optimización continua.

En los **algoritmos genéticos**, las soluciones se representan como cromosomas, que se evalúan y ordenan de mejor a peor según su valor conforme a una función de aptitud o *fitness*. Las nuevas soluciones se generan simulando la selección natural a través de la aplicación de forma repetida de tres operadores genéticos: selección, cruce y mutación. Los algoritmos genéticos permiten una implementación simple que, de forma directa, no funciona bien con problemas complejos, si bien sí han sido incorporados con éxito a modelos híbridos.

La **evolución diferencial** se basa en una población inicial de vectores generados aleatoriamente a partir de una distribución uniforme dentro del hiperespacio de posibles soluciones. La descendencia de un vector se calcula a través de modificaciones sobre este, obtenidas con la diferencia escalada de otros dos vectores seleccionados aleatoriamente. El vector resultante solamente reemplaza a su "padre" si su rendimiento es superior. Su marco teórico sencillo y facilidad de implementación, así como su fiabilidad y alto rendimiento, han hecho que esta técnica sea ampliamente empleada. Las principales desventajas son su tendencia al estancamiento en soluciones de baja calidad así como a sufrir de "maldición de la dimensión", además de la dificultad del ajuste de hiperparámetros en problemas complejos. Dentro del área de elaboración de cronogramas, este algoritmo ha sido empleado para la resolución del RCPSP [23], [24].

5. Sistemas emergentes bioinspirados

Los sistemas emergentes bioinspirados son aquellos que toman como modelo sistemas naturales que en conjunto manifiestan propiedades que no poseen cada una de las partes de dicho sistema. Entre ellos, cabe destacar PSO, colonia de hormigas, colonia de abejas, algoritmo de luciérnaga, *shuffle frog-leaping* y colonia de termitas ([5], [6], [25]). A continuación, se desarrollará el primero de ellos, por resultar uno de los más representativos en el estado del arte.

PSO (*particle swarm optimization*) se inspira en el movimiento de sistemas de organismos vivos, como bandadas de aves. En un enjambre de partículas, cada partícula representa una posible solución, y se mueve a una nueva posición sumando su velocidad. Inicialmente esta posición y velocidad son aleatorias. Una vez que las partículas alcanzan su nueva posición, se evalúa la función de *fitness* para cada una de ellas y se actualiza cuál ha sido el mejor *fitness* de la partícula y del enjambre. Con esta información, se actualiza la velocidad de cada partícula, como la suma ponderada de la velocidad anterior, la diferencia de la posición de la partícula con su mejor posición histórica -su experiencia individual- y la diferencia de la posición de la partícula con la mejor posición histórica de todas las partículas del enjambre -la experiencia grupal-. En estos dos últimos sumandos se introduce un factor aleatorio. Este proceso se repite hasta que se alcanza el criterio de parada. Gracias a su sencillez,

su rendimiento es bueno cuando trabaja con poblaciones grandes. Una desventaja es su tendencia a devolver soluciones subóptimas.

Este algoritmo ha sido empleado para la resolución con éxito del RCPSP [26], considerando cada partícula como un posible cronograma, con la capacidad de encontrar óptimos globales.

Además de las técnicas descritas, dentro del área de la elaboración automática de cronogramas han sido empleados otros algoritmos, como aprendizaje por refuerzo, recocido simulado o lista tabú [15].

B. Técnicas híbridas

La hibridación es el proceso por el que se combinan dos o más técnicas básicas, para integrar sus puntos fuertes y soslayar sus debilidades individuales. Debido a su extensión, se desarrollarán únicamente las principales líneas de estudio, que integran fundamentalmente lógica borrosa, redes neuronales y algoritmos evolutivos. Los sistemas borrosos dotan de explicabilidad a las soluciones, pero cuentan con la dificultad del ajuste de sus parámetros. Los algoritmos evolutivos permiten la optimización global de estos y el uso de redes neuronales permite optimizar la búsqueda local.

1. Neuroevolución

Estas técnicas consisten en la hibridación de redes neuronales con algoritmos genéticos, de modo que un algoritmo genético es el responsable de la evolución de la red neuronal. En las primeras implementaciones, el algoritmo genético buscaba los pesos más adecuados para la red. Con posterioridad, estas técnicas ampliaron su alcance a la determinación de una topología óptima de la red [27]. Algoritmos basados en redes *feed-forward* y *fully-connected* logran disminuir el tiempo de procesado para la búsqueda de una red óptima en torno a un 80% [28].

En el área de planificación, estas técnicas han sido empleadas para la resolución de RCPSP por Agarwal et al. [29] en un algoritmo en el que se aplican iteraciones en serie combinando algoritmos genéticos y redes neuronales.

2. Sistemas neurodifusos

Los sistemas neurodifusos (*neuro-fuzzy*) son modelos híbridos que combinan el poder de aprendizaje de las redes neuronales y la funcionalidad de la lógica borrosa. Las metodologías para desarrollar sistemas neurodifusos utilizan el algoritmo de aprendizaje para obtener las capas de *fuzzificación* e inferencia.

Entre estas técnicas, **ANFIS** (*adaptive neuro-fuzzy inference system*) [30] es la que mejores resultados ha conseguido y ha sido más ampliamente utilizada [5], [25]. ANFIS es un sistema borroso que ajusta los parámetros de las reglas con una red neuronal, averiguando, además, las funciones de pertenencia óptimas. Una ventaja de ANFIS es que no necesita la opinión de expertos para modelar y entrenar estos sistemas [31]. En la elaboración automática de cronogramas ha sido aplicado para agendar trabajos realizados por maquinaria [32]. La principal desventaja de ANFIS es su coste computacional, que puede generar modelos complejos para problemas relativamente sencillos y la necesidad de una topología y funciones de pertenencia para cada aplicación específica [25].

3. Sistemas neurodifusos con algoritmos evolutivos

El problema de determinar automáticamente la topología y las funciones de pertenencia ha sido resuelto combinando un algoritmo evolutivo con los algoritmos neurodifusos, dando lugar a **EFNIM** (*Evolutionary Fuzzy Neural Inference Model*) [33]. Una evolución de EFNIM es **EFHNN (***Evolutionary Diffuse Hybrid Neuronal Network*) [8] que, en lugar de redes neuronales tradicionales, emplea redes neuronales híbridas, con términos no lineales, que permiten capturar mejor las no linealidades y, con ello superar el desempeño de las redes neuronales tradicionales.

Dentro del ámbito de la elaboración automática de cronogramas, estos algoritmos han sido aplicados a la planificación de proyectos de plantas fotovoltaicas [34].

Además de estas técnicas híbridas, otras relevantes son los sistemas borrosos hibridados con algoritmos evolutivos o con sistemas emergentes bioinspirados [5], EFSIM (*Evolutionary Fuzzy Support Vector Machines Inference Model*), que sigue el esquema de EFNIM, si bien utiliza máquina de vector de soporte en lugar de la red neuronal [13], o técnicas que combinan aprendizaje por refuerzo adaptativo [15].

III. Objetivos y Metodología

El objetivo de este trabajo es realizar el análisis comparativo de cinco técnicas de inteligencia artificial basadas en aprendizaje automático para la elaboración de cronogramas de construcción de grandes plantas fotovoltaicas (> 1 MW). En dicho análisis se obtendrá un conjunto de métricas estándar (MSE, MAE, R^2), con las que se determinará cuál técnica consigue la mejor solución, así como si superan una determinada línea base.

Los objetivos específicos de este trabajo son los siguientes:

3. **Selección de las técnicas** de inteligencia artificial más adecuadas para la comparativa.
4. **Elaboración de los modelos** conceptuales, implementación en software y optimización.
5. **Realización de la comparativa** entre los modelos y determinación de un posible modelo óptimo.
6. **Desarrollo de una herramienta** que permita obtener un cronograma de construcción a partir de unos datos de entrada a través del modelo óptimo.

Para alcanzar los objetivos propuestos, se seguirán los siguientes **pasos,** a través de una metodología empírico-cuantitativa: 1) revisión del estado del arte y trabajos relacionados; 2) determinación de los modelos que formarán parte de la comparativa, en base al análisis del estado del arte; 3) elaboración de los distintos modelos y métricas desde un punto de vista conceptual; 4) preparación del conjunto de datos, formado por 25 cronogramas, y análisis exploratorio de estos; 5) modelado e implementación en software a través de Python; 6) entrenamiento de los modelos; 7) obtención de las métricas de los distintos modelos: MAE, MSE, R^2 y tiempo de ejecución; 8) validación de los resultados obtenidos; 9) optimización de los distintos modelos, en función del resultado de la validación; 10) determinación de un posible modelo óptimo; 11) desarrollo de una herramienta que permita obtener un cronograma de construcción a partir de unos datos de entrada a través del modelo óptimo; 12) elaboración de conclusiones. Los pasos entre el entrenamiento y la validación se repetirán de forma iterativa para optimizar los modelos, hasta alcanzar métricas satisfactorias. Asimismo, la optimización de los modelos puede conllevar la necesidad de nuevos análisis exploratorios de los datos.

IV. Contribución

A partir de la aplicación del estado del arte al problema objeto de estudio, y siguiendo criterios de optimalidad, explicabilidad, variedad y novedad, se han seleccionado cinco algoritmos para la comparativa de evaluación de rendimiento:

1. **Árboles de decisión**, por su explicabilidad.
2. **PSO**, por ser el segundo grupo con mejores resultados en la comparativa de Pellerin et al. ([6]). Además, es posible dotar a su implementación de explicabilidad.
3. **Evolución diferencial**, como representante de los algoritmos evolutivos, habiendo conseguido estos los mejores resultados en la comparativa de Pellerin et al.

Asimismo, puede implementarse de forma explicable.

4. **Neuroevolución**, por su novedad en este problema.
5. **ANFIS**, por su buen desempeño en gran variedad de problemas y su explicabilidad gracias a la capa borrosa.

El conjunto de datos de que se dispone para el **modelado** consta de 25 registros y 85 variables. De estas variables, se han seleccionado como entrada al modelo las 13 características más representativas de la planta fotovoltaica: variables generales (potencia), de obra civil (área, perímetro, pilares, carreteras, zanjas), obra mecánica (seguidores, paneles), obra eléctrica (línea de baja tensión y de media tensión, número de inversores, potencia del inversor) y línea de transmisión (distancia). Para la elaboración del cronograma de construcción, ha sido suficiente tomar 55 variables de salida, agrupadas en las siguientes fases: obra civil, obra mecánica, obra eléctrica, línea de transmisión y puesta en marcha. Dichas variables corresponden a duraciones de fases (según la notación empleada, FC65D a FP99D) e intervalos entre el inicio de dos fases consecutivas (FC65I a FP99I).

Se observa, además, que la ejecución de la línea de transmisión se encuentra condicionada por tareas predecesoras diferentes a las de la planta y de las que no se dispone de información. Por ello, es preciso separar el modelo en dos partes: la línea de transmisión y la planta propiamente dicha. El modelo de la línea de transmisión consta únicamente de dos variables de salida. Una regresión lineal a partir de dos variables de entrada (potencia y distancia) devuelve resultados satisfactorios. El problema a resolver se centra, pues, en el modelado del cronograma de la propia planta, con 12 variables de entrada y 53 variables de salida.

Para evaluar la bondad de los distintos modelos, se definirá una **línea base**, en la que la predicción vendrá dada por el valor medio de cada una de las variables de salida, y sobre la que se obtendrán las mencionadas métricas. Para la **validación de cada modelo**, se realizará una batería de ocho test. En cada test se emplea un conjunto de entrenamiento de 20 registros y un conjunto de test de 5 registros. Todos los registros existentes forman parte de alguno de los conjuntos de test. Las métricas finales que serán objeto de la comparativa se obtendrán como promedio de las métricas obtenidas en cada uno de estos test.

A continuación, se desarrollan las técnicas empleadas en la comparativa.

1. Árboles de decisión

Para esta técnica se empleará tanto un árbol aislado como dos algoritmos *ensemble*: *XGBoost* y *Random forest*. La implementación del algoritmo se realiza empleando un árbol de regresión para cada una de las variables de salida [35]. Cada árbol (submodelo) se alimenta con todas las variables de entrada, según se muestra en la figura 1.

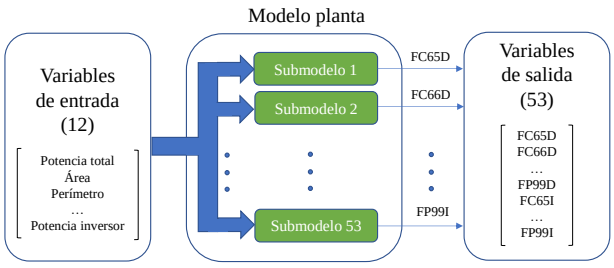


Fig. 1. Modelo de la planta, con un submodelo por cada variable de salida. Este modelo es válido para árboles, PSO y evolución diferencial.

Esta estrategia dota de suficiente flexibilidad e independencia a las variables de salida, lo cual es de gran importancia dado el reducido número de registros. Se busca la combinación óptima de hiperparámetros (profundidad máxima y número mínimo de elementos por hoja) por *k-fold cross validation*, con *k*=5. A fin de evitar sobreajuste, se limita la profundidad máxima de cada árbol a 6 niveles y un número de ejemplos por hoja de 2. Para tratar de mejorar su rendimiento, se realizan pruebas con un modelo basado en *boosting* -*XGBoost*- y otro basado en *bagging* -*Random Forest*-. Estos modelos perderán explicabilidad, al estar constituidos por un conjunto de árboles.

XGBoost

Esta técnica consiste en elaborar un modelo formado por sucesivos árboles de regresión, dando en cada una de las sucesivas iteraciones un mayor peso a aquellos ejemplos que hayan sido incorrectamente clasificados con anterioridad. El modelo resultante es la suma de todos los árboles calculados [36]. Esta estrategia permite resolver problemas detectados con los valores extremos de las variables de salida. Para poder evaluar la bondad de las predicciones durante el entrenamiento, y así realizar el reajuste, es necesario definir un conjunto de validación dentro del propio conjunto de entrenamiento. Se han tomado para ello 5 registros. Con esta única salvedad, el proceso de entrenamiento es análogo al descrito para árboles de decisión.

Random forest

Este algoritmo se basa en la generación un conjunto de árboles de decisión. Para cada uno de ellos se toma únicamente una parte de las variables de entrada (habitualmente $\sqrt{p}$, siendo p el número de variables de entrada) y una parte del conjunto de entrenamiento (habitualmente 2/3). El resultado de la predicción se toma como el promedio del resultado de cada uno de los árboles individuales. Con ello se pretende evitar el sobreajuste que suele existir al modelar con un único árbol, a costa de perder una explicabilidad sencilla, al encontrarse "diluida" entre todos los modelos. Tras diversas pruebas, el número óptimo de árboles se ha establecido en 500. Un mayor número de árboles conduce a mayor sobreajuste, sin mejorar las métricas de test. Un número menor de árboles conlleva problemas de estabilidad en las soluciones.

2. Optimización de enjambre de partículas (PSO)

El modelo empleado para PSO emplea un enjambre para cada una de las variables de salida (figura 1) [37]. Cada una de las partículas de un enjambre busca la posición óptima en el hiperespacio. Las dimensiones de la posición de una partícula representan los parámetros que relacionan a la correspondiente variable de salida con las variables de entrada. Se han supuesto dependencias lineales entre las variables de entrada y las variables de salida. Así, la estimación de cada variable de salida y_i puede representarse en función de las n variables de entrada x_j como

$$\hat{y}_i = \sum_{j=1}^{n}(w_j x_j) + b_i \qquad (1)$$

, siendo w_j, b_i los parámetros a calcular, que definen la posición de la partícula: un parámetro lineal w_j por cada una de las 12 variables de entrada más un término independiente b_i. La función de coste, a minimizar, es la suma de los cuadrados de los errores entre el valor calculado de la variable de salida $\hat{y}_i$ y su valor real y_i para las m muestras del conjunto de entrenamiento.

$$Coste\,(y_i, \hat{y}_i) = \frac{1}{m}\sum_{i=1}^{m}\|y_i - \hat{y}_i\|^2 \qquad (2)$$

Esta función de coste se evalúa para cada una de las partículas del enjambre. Con ella, es posible actualizar la posición de cada partícula, considerando que será atraída por su mejor posición histórica, con un peso c_1, la mejor posición histórica del enjambre, con un peso c_2, y una inercia frente a su movimiento anterior con peso w. Los mejores resultados se han obtenido con un total de 1000 partículas por enjambre. La estrategia de búsqueda de hiperparámetros en red es la que consigue mejores resultados, en torno a $c_1 = 0{,}8$, $c_2 = 0{,}3$ y $w = 0{,}8$. Esta búsqueda se lleva a cabo para cada una de las variables de salida. Se observa que el algoritmo converge de forma muy rápida.

3. Evolución diferencial

Este algoritmo emplea una población de vectores por cada salida (figura 1). Requiere definir los siguientes hiperparámetros básicos: tamaño de la población, grado de mutación y constante de recombinación. Este algoritmo puede plantearse a través de distintas estrategias. La que mejores resultados ha dado es la conocida como *'best1bin'* [38]. En ella, la diferencia entre los dos vectores seleccionados aleatoriamente se utiliza para mutar el mejor miembro de la población. Este vector mutado será el que transfiera algunos de sus parámetros al vector a evolucionar, en función de la constante de recombinación. La función de coste, a minimizar, es análoga a la descrita en PSO. Tras diversas pruebas, los mejores resultados se han obtenido con un parámetro de población de 25 vectores, un grado de mutación entre 0,5 y 1 y un grado de recombinación de 0,5. Un requerimiento de este algoritmo es acotar el espacio de búsqueda de los parámetros a través de un conjunto de límites, que se ha establecido en (-2,2).

4. Neuroevolución

La implementación realizada en esta técnica busca definir la topología de la red neuronal a través del algoritmo genético. Se parte de una capa de entrada con un número de neuronas igual al número total de variables de entrada (12) y una capa de salida con un número de neuronas igual al número total de variables de salida (53). Si bien las aproximaciones basadas en NEAT resultan interesantes desde un punto de vista teórico, su implementación práctica no resulta atractiva. Así, se ha optado por una implementación basada en Keras [28]. En ella, cada individuo de la población es una red neuronal alimentada hacia delante, completamente conectada y con varias capas ocultas. Cada individuo se encuentra definido por los siguientes parámetros o "genes", para los que los valores finales son:

1. *Número de neuronas por capa oculta*. 8 valores predefinidos en una lista de valores entre 120 y 370 neuronas.
2. *Número de capas ocultas*. 2, 3 ó 4 capas.
3. *Función de activación*: ReLU o eLU.
4. *Optimizador*: Adam, Adagrad, Adadelta, RMSprop.

Como hiperparámetros del algoritmo genético, se definen:

1. *número de generaciones*: 20.
2. *población de cada generación*: 20 redes por generación.
3. *ratio de la población que debe permanecer tras cada generación*: 0,4.
4. *selección aleatoria*: probabilidad de que una red no seleccionada por su fitness continúe en la población. Se emplea un valor de 0,1.
5. *grado de mutación*: representa la probabilidad de que una red sufra alguna mutación. Se emplea un valor de 0,2.

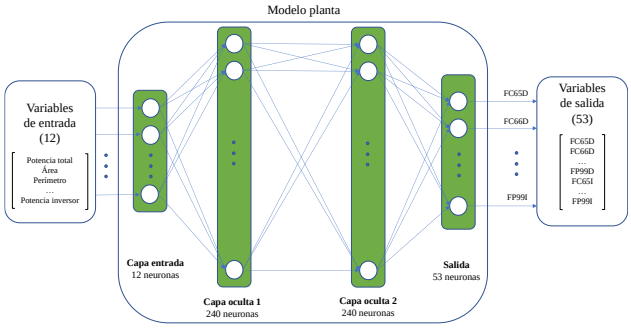


Fig. 2. Arquitectura de red neuronal obtenida por el algoritmo genético.

El entrenamiento de cada red se ejecuta con 100 *epoch* con condición de parada temprana en caso de que tras 10 iteraciones no se produzcan mejoras en el rendimiento. Una vez llevada a cabo la selección por parte del algoritmo genético, el modelo obtenido será el individuo (red) con mejor *fitness* de la última generación. La función de *fitness* debe ser maximizada, por lo que ha sido definida como el MSE cambiado de signo. Las mejores soluciones suelen encontrarse con 2 o 3 capas y en torno a 240 neuronas por capa. La figura 2 muestra un ejemplo de arquitectura. La función de activación en las capas ocultas que ha devuelto mejores resultados ha sido ReLU. Cada capa, excepto la de salida, lleva a continuación un *dropout* de 0,2, para regularizar el comportamiento de la red. Por último, la capa de salida también incorpora función de activación ReLU.

5. ANFIS

En esta implementación [39], los recursos de computación limitan en la práctica el número de variables de entrada a un máximo de cinco por modelo. Ello ha implicado la necesidad de reformular de cada variable de salida en función de un máximo de cinco variables de entrada, tal y como muestra la figura 3.

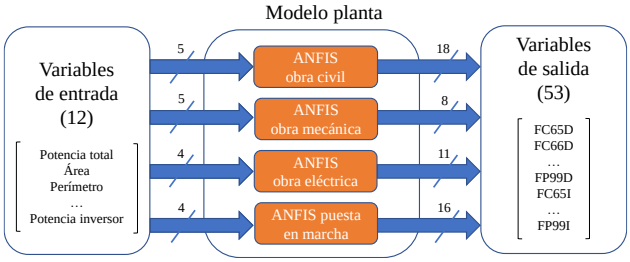


Fig. 3. Modelo global de la planta basado en ANFIS.

La figura 4 detalla cómo se realiza la descomposición de cada conjunto de modelos ANFIS, utilizando como ejemplo el conjunto ANFIS obra eléctrica. Se observa cómo para este caso existen 11 modelos ANFIS (uno por variable de salida), cada uno de ellos alimentado por las mismas 4 variables de entrada.

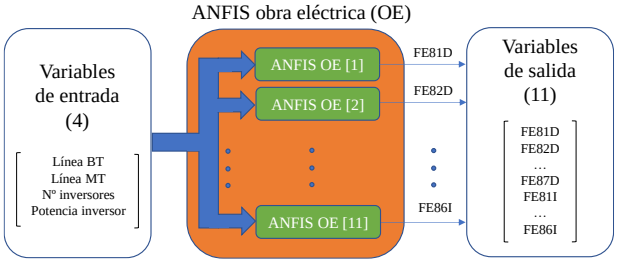


Fig. 4. Modelado ANFIS del grupo de variables de obra eléctrica.

A su vez, cada uno de estos modelos ANFIS implementa un sistema neurodifuso, ejemplificado en la figura 5 para la variable FE81D (duración de la fase de tendido de la línea de baja tensión).

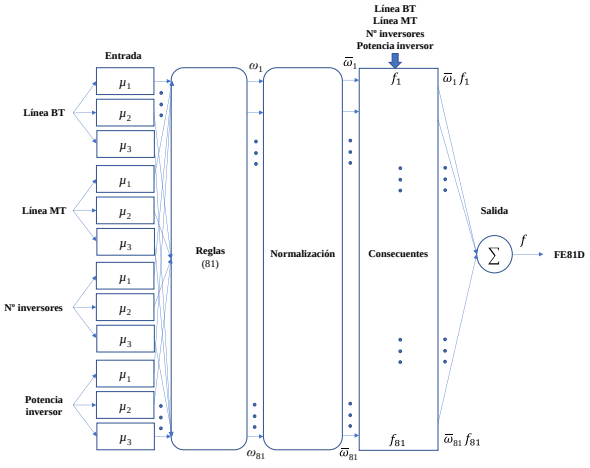


Fig. 5. Detalle del sistema neurodifuso de un modelo ANFIS.

ANFIS requiere para ello de la definición de un conjunto inicial de funciones de pertenencia borrosas $\mu(x)$ para cada una

de las variables de entrada x. Una definición que ha devuelto buenos resultados es con tres funciones de pertenencia gaussianas por variable, con σ=0,45 y μ=0, 0,5 y 1.

ANFIS realiza varios *epochs* en el entrenamiento. Un número a partir de 4 no conduce necesariamente a mejores resultados y en algún caso muestra inestabilidad.

V. Resultados

Las métricas principales obtenidas en promedio en los test pueden observarse en la tabla I. La figura 6 muestra estos resultados gráficamente. Se puede observar que las **técnicas que presentan un mejor rendimiento** son, por este orden, ANFIS, neuroevolución, árboles de decisión, PSO y evolución diferencial. ANFIS destaca por su buen rendimiento en todas las métricas, seguido por neuroevolución. Los árboles de decisión presentan un resultado de MAE cercano, si bien su rendimiento en MSE y R^2 es inferior. El siguiente modelo en cuanto a rendimiento es PSO. Aunque su MSE y R^2 mejoran la línea base, su rendimiento en MAE es peor, lo que significa que, en la práctica, no resulta de utilidad predictiva. Por último, los resultados de evolución diferencial son prácticamente análogos a los dados por la línea base, esto es, por la media.

La comparativa de las métricas de entrenamiento frente a las de test permiten evaluar posibles sobreajustes, que se observan en mayor grado en las técnicas de árboles, especialmente en *Random forest*, que en entrenamiento devuelve MAE próximo a 0 y R^2 cercano a 1. En el resto de técnicas no se observa un grado importante de sobreajuste, si bien los resultados de ANFIS en entrenamiento son claramente superiores a los de test.

Otra métrica relevante es el tiempo de ejecución. Las técnicas basadas en árboles son las más rápidas (entre 26 y 157 segundos). PSO y evolución diferencial presentan tiempos medios de ejecución de 572 y 213 segundos. Los algoritmos con un mayor tiempo de ejecución son neuroevolución (704 segundos) y, especialmente, ANFIS (2176 segundos). El análisis de los resultados para cada uno de los ocho conjuntos de test muestra consistencia en el orden de desempeño de los algoritmos, siendo ANFIS la técnica que ofrece el rendimiento más equilibrado en todos los test.

VI. Discusión

La técnica que ha obtenido mejor rendimiento en las métricas de error, de modo consistente y sin sobreajustes relevantes, ha sido **ANFIS**. Las funciones de pertenencia y las reglas borrosas consiguen valorar adecuadamente la importancia de cada una de las entradas y devolver una salida ajustada. Este buen rendimiento tiene su contrapartida en un alto coste computacional, con un promedio de 35 minutos, tres veces superior al de la siguiente técnica más costosa. Su explicabilidad no es inmediata, ya que cada salida se compone del resultado ponderado de 3^n reglas,

Tabla I

Comparativa entre las distintas técnicas

		Línea base	Árbol regresión	XGBoost	Random forest	PSO	Evolución diferencial	Neuro evolución	ANFIS
MAE entrenamiento	días	8,48	2,38	1,80	0,06	7,71	8,48	4,34	2,10
MSE entrenamiento	días^2	259,12	32,89	46,74	0,96	105,28	259,13	49,41	20,35
R^2 entrenamiento	-	0,0000	0,8748	0,8166	0,9963	0,6150	0,0000	0,8079	0,9242
MAE test	días	9,17	5,57	5,38	5,26	9,80	9,19	5,36	4,54
MSE test	días^2	275,86	129,29	117,45	137,65	193,34	276,16	75,41	67,19
R^2 test	-	-0,4201	0,3650	0,4354	0,3225	0,0383	-0,4216	0,5909	0,6545
Tiempo	s	0	26	157	47	572	213	704	2176

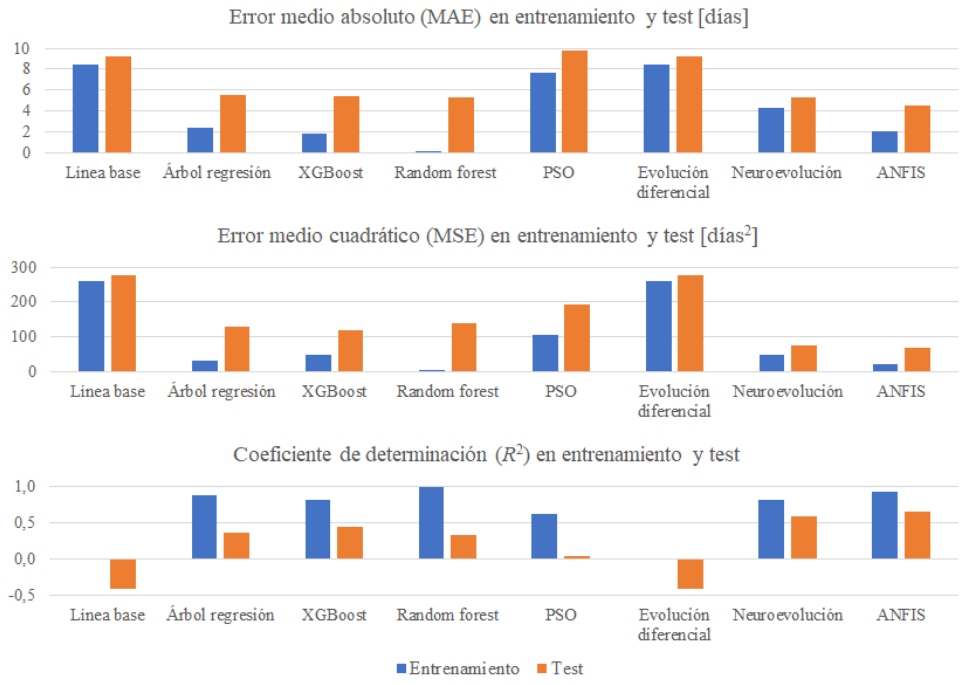


Fig. 6. Gráfico resumen de los resultados de la comparativa entre las distintas técnicas.

siendo *n* el número de variables de entrada. Una simplificación del conjunto de reglas puede paliar estas desventajas.

La siguiente técnica en rendimiento es **neuroevolución**. En relación a ANFIS, su MSE y R^2 son un 10% inferiores, y un 20% en el caso de MAE. Estos buenos resultados se repiten en todos los test y suponen una importante reducción en el coste computacional con respecto a ANFIS. Otra de las ventajas de este método es que su tendencia al sobreajuste es la menor. Su principal desventaja es su falta de explicabilidad, con típicamente más de 70.000 parámetros, sin significado físico. Aun con un coste computacional considerable (promedio de 12 minutos), sus soluciones son inferiores a las de ANFIS.

En consecuencia, se aprecia un **rendimiento superior de las técnicas híbridas** frente a las técnicas básicas.

Dentro de las técnicas básicas, aquellas que han ofrecido mejores resultados son las basadas en **árboles de decisión**. Su clasificación por orden de rendimiento en test es: *XGBoost*, *Random forest* y árbol de regresión aislado, sin diferencias muy significativas. Su MAE es muy similar a la de neuroevolución, si bien los resultados en R^2 y MSE son claramente inferiores, lo que significa que estas técnicas presentan un comportamiento peor en cuanto a errores grandes, una menor explicabilidad del modelo y la existencia de posibles sesgos. Pese a ello, el resultado es razonablemente bueno para obtener una primera aproximación, con tiempos de ejecución rápidos. Estas técnicas, sin embargo, adolecen de un importante sobreajuste. Por último, solamente la técnica basada en un árbol aislado ofrece buena explicabilidad.

Los resultados obtenidos con **PSO** han sido inferiores a los dados por árboles de decisión y pueden calificarse de insatisfactorios. MSE y R^2 mejoran a los dados por la línea base. Sin embargo, MAE, con un valor próximo a 10 días, ofrece un rendimiento inferior a la línea base. Esta técnica converge rápidamente a soluciones, en general, subóptimas. Su coste computacional ha sido relativamente elevado (en torno a 10 minutos). Una eventual ventaja, en caso de haber alcanzado resultados más satisfactorios, es su baja tendencia al sobreajuste. Si bien la hipótesis de linealidad no ha supuesto una solución adecuada a este problema, esta permite que el modelo sea explicable. Las soluciones de mayor calidad, como se ha observado, introducen mayor complejidad en su modelado.

La técnica de la comparativa que ha devuelto peores resultados ha sido **evolución diferencial**. El resultado de la predicción dada por este modelo es muy aproximadamente el valor medio de la variable en el conjunto de entrenamiento, por lo que las métricas de error son muy similares a las de la línea base. Un motivo de la mala calidad de las predicciones es la hipótesis de linealidad adoptada entre las variables de entrada y las de salida. Esta hipótesis impide explorar el conjunto de mejores soluciones, por lo que el algoritmo se estanca en regiones subóptimas. Su tiempo de ejecución es bajo (solamente superior al de árboles), lo que sería una eventual ventaja si las soluciones fuesen satisfactorias.

Una perspectiva adicional en el análisis de resultados es su aportación en términos absolutos, más allá de la comparativa, para su **aplicación práctica**. Para ello se emplea modelo que mejores resultados ha devuelto, ANFIS. Una primera medida de la bondad de los resultados alcanzados puede tomarse contrastando el valor de MAE en test (4,54 días) frente al valor absoluto medio de las variables de salida (29,17 días). Esto supone un error relativo promedio de un **16%** en el cálculo de las duraciones y los intervalos entre las distintas fases de construcción. Por otro lado, un R^2 de 0,6545 puede considerarse muy bueno ante un conjunto de datos muy pequeño y de gran variabilidad. Sirva como referencia que la línea base dada por la media de los datos de entrenamiento obtiene un R^2 de -0,4201.

La prueba final de la bondad del modelo consiste en **cotejar cronogramas reales de test frente a los cronogramas que elaboraría el modelo**. Para la máxima verosimilitud de la prueba, se ha utilizado el modelo ANFIS obtenido con aquel test -de entre los ocho realizados- donde las métricas de error han sido las más similares a las métricas promedio. Los resultados obtenidos con este modelo son notablemente buenos. Observando la **duración *D* total** de los cronogramas obtenidos (tabla II), su **error *e* relativo** es de un **8%**, con un máximo de un 10%, sin sesgos significativos al alza o a la baja. Puede concluirse que los errores existentes en cada una de las fases individuales (16%) no contienen sesgos relevantes, lo que permite su cancelación mutua parcial, para alcanzar el citado error en la duración total de un 8%.

TABLA II

RESULTADO DE LOS CRONOGRAMAS OBTENIDOS CON ANFIS

		Planta 1 96 MW	**Planta 21** 146 MW	**Planta 11** 65 MW	**Planta 3** 50 MW	**Planta 5** 37 MW
D real	días	253	340	220	226	162
D predicha	días	278	311	242	215	175
e total	días	25	-29	22	-11	13
e relativo	%	10%	9%	10%	5%	8%

VII. CONCLUSIONES

La necesidad de impulsar la construcción de infraestructuras de generación renovable sirve como motivación para tratar de resolver, a través de técnicas de inteligencia artificial, la principal dificultad para cumplir el plazo de construcción de estos proyectos: la elaboración de una planificación adecuada. Una herramienta capaz de generar cronogramas de forma automática, sin influencia de sesgos humanos, puede resultar de gran ayuda a la hora de establecer planificaciones iniciales de referencia. Este trabajo concreto ha profundizado en la generación de cronogramas de construcción de plantas de gran potencia basadas en tecnología fotovoltaica a partir de datos históricos.

Los resultados obtenidos muestran un rendimiento superior de las **técnicas híbridas**. Aquella que mejores resultados ha devuelto es **ANFIS**, con un error medio absoluto en torno a 4,5 días y un coeficiente de determinación R^2 de 0,65. El algoritmo de **neuroevolución** ha conseguido unos resultados satisfactorios, ligeramente inferiores a los de ANFIS, con un menor coste computacional. En cuanto a las **técnicas básicas**, solamente las técnicas basadas en **árboles de decisión** consiguen mejorar claramente la predicción dada por la media del conjunto de entrenamiento. Tanto **PSO** como **evolución diferencial** han sufrido de problemas de estancamiento en soluciones subóptimas, siendo esta última incapaz de superar a la línea base.

Por último, se ha desarrollado una **herramienta** capaz de establecer planificaciones de referencia en base al modelo y a partir de 14 variables de entrada -una de las cuales es la fecha de inicio de la construcción-. Los resultados obtenidos por esta herramienta, con un error relativo de un 8% en la duración total de la planificación, han superado las expectativas previas.

Algunas posibles **líneas de trabajo futuras** que han surgido como consecuencia de este estudio son:

1. **Testear nuevos modelos híbridos**, a la vista del superior resultado alcanzado por este tipo de modelos.
2. **Desarrollar modelos básicos** que permitan superar las limitaciones de los que han formado parte de la comparativa.
3. **Realizar un modelado paramétrico, con dependencias no lineales**, a partir de un conocimiento experto del dominio.
4. **Enriquecer el conjunto de entrada** con un mayor número de registros, mayor diversidad de los datos de entrada y con variables adicionales que permitan afinar el modelo.
5. **Ajustar duraciones de tareas en función de la época del año**. Factores como las horas de luz diarias o la climatología pueden afectar sensiblemente al avance de los trabajos.

REFERENCIAS

[1] IRENA, *Renewable capacity statistics 2020 International Renewable Energy Agency (IRENA), Abu Dhabi*. 2020.

[2] UNEF, «Annual report. 2021. Energía solar fotovoltaica. Nuestra Energía cambia el mundo», 2021.

[3] D. K. H. Chua, P. K. Loh, Y. C. Kog, y E. J. Jaselskis, «Neural networks for construction project success», *Expert Syst. Appl.*, vol. 13, n.º 4, pp. 317-328, 1997.

[4] M. E. Georgy, L. M. Chang, y L. Zhang, «Prediction of engineering performance: A neurofuzzy approach.», *J. Constr. Eng. Manag.*, vol. 131, n.º 5, pp. 548-557, 2005.

[5] N. G. Seresht, R. Lourenzutti, A. Salah, y A. R. Fayek, *Overview of Fuzzy Hybrid Techniques in Construction Engineering and Management*. 2018.

[6] R. Pellerin, N. Perrier, y F. Berthaut, *A survey of hybrid metaheuristics for the resource-constrained project scheduling problem*, vol. 280, n.º 2. 2020.

[7] C. H. Ko y M. Y. Cheng, «Dynamic Prediction of Project Success Using Artificial Intelligence», *J. Constr. Eng. Manag.*, vol. 133, n.º 4, pp. 316-324, 2007, doi: 10.1061/(asce)0733-9364(2007)133:4(316).

[8] M. Y. Cheng, H. C. Tsai, y E. Sudjono, «Evolutionary fuzzy hybrid neural network for conceptual cost estimates in construction projects», *2009 26th Int. Symp. Autom. Robot. Constr. ISARC 2009*, n.º 2005, pp. 512-519, 2009, doi: 10.22260/isarc2009/0040.

[9] M. Y. Cheng, H. C. Tsai, y E. Sudjono, «Evaluating subcontractor performance using evolutionary fuzzy hybrid neural network», *Int. J. Proj. Manag.*, vol. 29, n.º 3, pp. 349-356, 2011, doi: 10.1016/j.ijproman.2010.03.005.

[10] Y. Y. Shou, «An ant colony algorithm for improving the net present values of resource constrained projects», *J. Chinese Inst. Ind. Eng.*, vol. 23, n.º 6, pp. 478-483, 2006, doi: 10.1080/10170660609509344.

[11] M. Dorigo, M. Birattari, y T. Stützle, «Ant colony optimisation», *IEEE Comput. Intell. Mag.*, vol. 1, n.º 4, pp. 28-39, 2019, doi: 10.1007/978-3-319-93025-1_3.

[12] V. Kachitvichyanukul, «Comparison of Three Evolutionary Algorithms »:, *Ind. Eng. Manag. Syst.*, vol. 11, n.º 3, pp. 215-223, 2012.

[13] D. Magaña y J. C. Fernández, «Artificial Intelligence Applied to Project Success: A Literature Review», *Int. J. Interact. Multimed. Artif. Intell.*, vol. 3, n.º 5, p. 77, 2015, doi: 10.9781/IJIMAI.2015.3510.

[14] V. Faghihi, A. Nejat, K. F. Reinschmidt, y J. H. Kang, «Automation in construction scheduling: a review of the literature», *Int. J. Adv. Manuf. Technol.*, vol. 81, n.º 9-12, pp. 1845-1856, 2015, doi: 10.1007/s00170-015-7339-0.

[15] M. H. Fazel, A. A. Sadat, S. Sotudian, y O. Castillo, *A state of the art review of intelligent scheduling*, vol. 53, n.º 1. Springer Netherlands, 2020.

[16] H. Adeli y A. Karim, «Scheduling/Cost Optimization and Neural Dynamics Model for Construction», *J. Constr. Eng. Manag.*, vol. 123, n.º 4, pp. 450-458, dic. 1997, doi: 10.1061/(asce)0733-9364(1997)123:4(450).

[17] S. A. Hashemi Golpayegani y B. Emamizadeh, «Designing work breakdown structures using modular neural networks», *Decis. Support Syst.*, vol. 44, n.º 1, pp. 202-222, nov. 2007, doi: 10.1016/J.DSS.2007.03.013.

[18] R. Avila Rondon, A. Carvalho, y G. Hernandez, «Neural network modelling and simulation of the scheduling», *Eighth IFIP Int. Conf. Inf. Technol. Balanc. Autom. Syst. Porto, Port.*, pp. 231-238, Springer USA, 2008.

[19] M. Y. Cheng, Y. H. Chang, y D. Korir, «Novel Approach to Estimating Schedule to Completion in Construction Projects Using Sequence and Nonsequence Learning», *J. Constr. Eng. Manag.*, vol. 145, n.º 11, p. 04019072, 2019, doi: 10.1061/(asce)co.1943-7862.0001697.

[20] M. Y. Cheng, L. C. Lien, H. C. Tsai, y P. H. Chen, «Artificial Intelligence Approaches to Dynamic Project Success Assessment Taxonomic», *Life Sci. J. 2012*, vol. 9(4), pp. 49-56, 2012.

[21] T. Portoleau, C. Artigues, y R. Guillaume, *Robust Predictive-Reactive Scheduling : An Information-Based Decision Tree Model*, vol. 2. Springer International Publishing, 2020.

[22] W. Guo, M. Vanhoucke, J. Coelho, y J. Luo, «Automatic detection of the best performing priority rule for the resource-constrained project scheduling problem», *Expert Syst. Appl.*, vol. 167, n.º October, p. 114116, 2021, doi: 10.1016/j.eswa.2020.114116.

[23] W. Chen y X. Ni, «Chaotic differential evolution algorithm for resource constrained project scheduling problem», *Int. J. Comput. Sci. Math.*, vol. 5, n.º 1, pp. 81-93, 2014.

[24] N. Rahmani, V. Zeighami, y R. Akbari, «A study on the performance of differential search algorithm for single mode resource constrained project scheduling problem», *Decis. Sci. Lett.*, vol. 4, n.º 4, pp. 537-550, 2015.

[25] G. G. Tiruneh, A. R. Fayek, y V. Sumati, «Neuro-fuzzy systems in construction engineering and management research», *Autom. Constr.*, vol. 119, n.º July, p. 103348, 2020, doi: 10.1016/j.autcon.2020.103348.

[26] H. Zhang, H. Li, y C. M. Tam, «Particle swarm optimization for resource-constrained project scheduling», *Int. J. Proj. Manag.*, vol. 24, n.º 1, pp. 83-92, 2006, doi: 10.1016/j.ijproman.2005.06.006.

[27] K. O. Stanley y R. Miikkulainen, «Evolving Neural Networks through Augmenting Topologies», *Evol. Comput.*, vol. 10, n.º 2, pp. 99-127, 2002, doi: 10.1162/106365602320169811.

[28] M. Harvey, «Let's evolve a neural network with a genetic algorithm», 2017. https://blog.coast.ai/lets-evolve-a-neural-network-with-a-genetic-algorithm-code-included-8809bece164 (accedido jul. 01, 2022).

[29] A. Agarwal, S. Colak, y S. Erenguc, «A Neurogenetic approach for the resource-constrained project scheduling problem», *Comput. Oper. Res.*, vol. 38, n.º 1, pp. 44-50, 2011, doi: 10.1016/j.cor.2010.01.007.

[30] J. S. R. Jang, «ANFIS: Adaptive-Network-Based Fuzzy Inference System», *IEEE Trans. Syst. Man Cybern.*, vol. 23, n.º 3, pp. 665-685, 1993, doi: 10.1109/21.256541.

[31] P. Aengchuan y B. Phruksaphanrat, «Comparison of fuzzy inference system (FIS), FIS with artificial neural networks (FIS + ANN) and FIS with adaptive neuro-fuzzy inference system (FIS + ANFIS) for inventory control», *J. Intell. Manuf.*, vol. 29, n.º 4, pp. 905-923, 2018, doi: 10.1007/s10845-015-1146-1.

[32] A. Azadeh, N. Hosseini, S. Abdolhossein Zadeh, y F. Jalalvand, «A hybrid computer simulation-adaptive neuro-fuzzy inference system algorithm for optimization of dispatching rule selection in job shop scheduling problems under uncertainty», *Int. J. Adv. Manuf. Technol.*, vol. 79, jul. 2015, doi: 10.1007/s00170-015-6795-x.

[33] C. H. Ko y M. Y. Cheng, «Hybrid use of AI techniques in developing construction management tools», *Autom. Constr.*, vol. 12, n.º 3, pp. 271-281, 2003, doi: 10.1016/S0926-5805(02)00091-2.

[34] J. Gil, J. Martínez, y R. González, «Planificación Y Gestion De Proyectos De Plantas Fotovoltaicas Aplicando Inteligencia Artificial.», n.º August, pp. 1-15, 2021.

[35] Scikit-learn, «Decision tree regressor», *Scikit-learn 1.1.1 documentation*. https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html.

[36] XGBoost, «XGBoost Python package», *XGBoost*. https://xgboost.readthedocs.io/en/stable/python/index.html.

[37] L. J. V. Miranda, «PySwarms», *PySwarms 1.3.0 documentation*, 2017. https://pyswarms.readthedocs.io/en/latest/ (accedido jul. 01, 2022).

[38] SciPy, «Differential evolution», *SciPy*, 2020. https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.differential_evolution.html (accedido jul. 01, 2022).

[39] T. Meggs, «Python implementation of an Adaptive neuro fuzzy inference system», *GitHub*, 2020. https://github.com/twmeggs/anfis (accedido jul. 01, 2022).