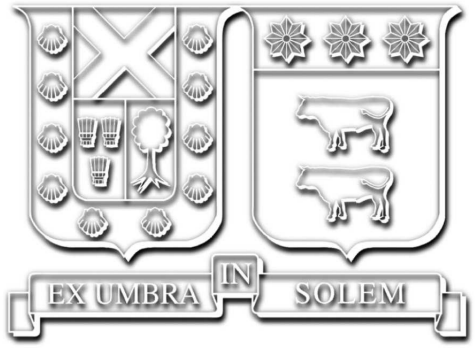
Universidad Técnica Federico Santa María
Departamento de Ingeniería Eléctrica
Valparaíso, Chile



Modelo multi-agente basado en aprendizaje reforzado profundo aplicado al problema de predespacho de unidades en sistemas hidrotérmicos

PHILIP HERNÁN GUERRA NÚÑEZ

2023

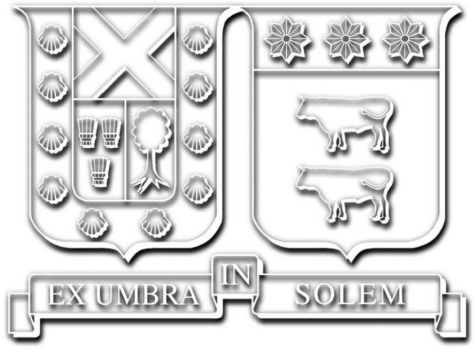
Requisito parcial para obtener el grado de:
Magíster en Ciencias de la Ingeniería Eléctrica

Profesor Guía:
Dr. Esteban Gil Sagás (UTFSM)

Comisión:
Dr. Víctor Hinojosa Mateus (UTFSM)
Dr. Carlos Valle Vidal (UPLA)

Valparaíso, Agosto 2023.

Universidad Técnica Federico Santa María
Departamento de Ingeniería Eléctrica
Valparaíso, Chile



Modelo multi-agente basado en aprendizaje reforzado profundo aplicado al problema de predespacho de unidades en sistemas hidrotérmicos

PHILIP HERNÁN GUERRA NÚÑEZ

2023

"Success is the sum of small efforts, repeated day in and day out."
— Robert Collier

Agradecimientos

Quiero expresar un profundo agradecimiento a mi familia, especialmente mi madre, por su amor incondicional y su incansable apoyo a lo largo de mi vida y durante mis estudios.

Quiero agradecer al Dr. Esteban Gil, mi guía durante el proceso de tesis, por su confianza en mí y por la asistencia proporcionada a lo largo de este período. Cada comentario y consejo que me brindó fueron fundamentales para el desarrollo de este trabajo. También quiero extender mi gratitud al Dr. Víctor Hinojosa, quien dedicó su tiempo a revisar mi trabajo y compartió conmigo valiosas enseñanzas a lo largo de mi trayectoria universitaria.

También quiero destacar a Pauline, a quien agradezco sinceramente por su comprensión y paciencia que me han acompañado a lo largo de este proceso. Sus palabras de aliento han constituido un pilar fundamental para llevar a cabo este trabajo con éxito.

Agradezco igualmente al Coordinador Eléctrico Nacional, por abrirme las puertas a desarrollar esta investigación junto a ellos. Durante todo el proceso, tuvieron una excelente disposición a responder a mis preguntas, proporcionarme datos relevantes y brindarme acceso a sus servidores para realizar las simulaciones necesarias. Especial agradecimiento para los Sres. Raúl Cárdernas, Gabriel Seguel y Felipe Cofré, miembros del departamento de Programación de la Operación, que con su conocimiento y apoyo enriquecieron mi trabajo y contribuyeron a su calidad.

Por último, doy las gracias al proyecto ANID-Basal FB0008 *Advanced Center for Electrical and Electronic Engineering* (AC3E) y al proyecto Fondecyt 1231892.

"Be the change you wish to see in the world"
— Mahatma Ghandi

Resumen

Este trabajo presenta una metodología para reducir el espacio de soluciones y acelerar los cálculos para el problema de predespacho de unidades (*Unit Commitment, UC*) en sistemas hidrotérmicos con un horizonte de tiempo de 168 horas basados en programación lineal entera mixta (Mixed-integer linear programming, MILP). La metodología *branch-and-bound* en problemas de UC basado en MILP enfrenta múltiples desafíos debido al aumento en el ciclaje de unidades a medida que los sistemas eléctricos reducen su huella de carbono. Con la metodología propuesta se mejora el rendimiento de los *solvers* aplicados al problema UC basado en MILP mediante el uso de cálculos *offline* y *online*.

El modelo *offline* entrena un modelo multi-agente basado en aprendizaje reforzado profundo (*Multi-agent deep reinforcement learning, MADRL*) utilizando datos históricos de operación del sistema eléctrico para predecir el estado de encendido/apagado de unidades térmicas seleccionadas. El modelo *online* utiliza las soluciones binarias obtenidas por el modelo *offline* para resolver un problema de UC con un espacio de soluciones reducido.

El enfoque multi-agente, basados en redes neuronales artificiales (*Artificial Neural Networks, ANN*) con una arquitectura de Red Convolucional Temporal (*Temporal Convolutional Network, TCN*), agrupa unidades que se encuentran en la misma región. Se utiliza una función de recompensa acumulativa compartida para ajustar simultáneamente los diferentes pesos de las ANNs durante la fase de aprendizaje.

La efectividad del método propuesto se demuestra utilizando datos reales de operación del sistema eléctrico chileno, logrando tiempos de cálculo significativamente más bajos y un error que se encuentra dentro del margen de integralidad del *solver*.

Abstract

This work presents a methodology to reduce the solution space to accelerate 168-hour-ahead Unit Commitment (UC) Mixed-Integer Linear Programming (MILP) computations in hydrothermal systems. The branch-and-bound methodology in MILP-based UC faces multiple challenges as power systems decarbonize due to the increased cycling of units. We improve solver performance in MILP-based UC by employing offline and online calculations.

The offline model trains a Multi-Agent Deep Reinforcement Learning (MADRL) model using historical power system operation data to predict specific units' on/off statuses. The online model uses the binary variable solutions obtained by the offline model to solve a UC problem with a reduced solution space.

The Multi-Agent approach allows each agent, based on Artificial Neural Networks (ANN) with a Temporal Convolutional Network (TCN) architecture, to group units that are located in the same region. A shared cumulative reward function is used to simultaneously adjust the different ANN's weights during the learning phase.

The effectiveness of our method is demonstrated using real operational data of the Chilean power system, achieving statistically significant lower computation times and a negligible error that is within the integrality gap of the solver.

Índice de Contenidos

Índice de Figuras

Índice de Tablas

Nomenclatura

Abreviaturas y siglas

ANN	:	*Artificial Neural Networks.*
CEN	:	Coordinador Eléctrico Nacional.
DL	:	*Deep Learning.*
ERV	:	*Energía renovable variable.*
MADRL	:	*Multi-agent deep reinforcement learning.*
ML	:	*Machine Learning.*
MILP	:	*Mixed-Integer Linear Programming.*
RL	:	*Reinforcement Learning.*
SEN	:	Sistema Eléctrico Nacional.
TCN	:	*Temporal Convolutional Network.*
UC	:	*Unit Commitment.*

Capítulo 1

Introducción

1.1 Contexto

Desde los últimos años ha habido un cambio significativo en los sistemas eléctricos con el crecimiento de la penetración de energía renovable variable (ERV) y la electrificación de la industria, lo que ha significado un gran desafío para sus operadores. Para el caso chileno, se prevee que las ERV seguirán aumentando rápidamente, principalmente la energía eólica y fotovoltaica, lo que genera un gran cambio en la matriz energética y un desafío importante para el operador del sistema, con tal de mantener una operación segura y confiable al menor costo posible. Este fenómeno se orienta a lo propuesto por el Coordinador Eléctrico Nacional (CEN), como ente técnico, independiente y autónomo responsable de la operación segura y económica del Sistema Eléctrico Nacional (SEN), para lograr una matriz eléctrica 100% renovable para el año 2050 [1]. Actualmente el CEN se encuentra en pleno proceso de descarbonización, retirando centrales a carbón con tal de que para el año 2025 se hayan retirado el 50% de estas [2]. Las ERV aumentan la variabilidad en el sistema, lo que genera un cambio importante con el que deben lidiar los operadores. Frente a esto, se necesita el desarrollo de nuevas herramientas para la planificación y operación del sistema con tal de obtener una modelación más precisa, granular y con tiempos de simulación más eficientes.

Dentro de los diversos problemas que se deben resolver en la planificación de la operación, el problema de predespacho de la generación (*Unit Commitment*, UC) es uno de los más relevantes. Este problema busca determinar la puesta en servicio o la salida de unidades generadoras para satisfacer la demanda dentro de un periodo determinado de tiempo, considerando las restricciones técnicas del sistema y buscando un costo de operación mínimo [3, 4]. El horizonte de análisis puede ir desde un día hasta una semana.

Este problema es formulado como un problema de optimización que contiene un gran número de cálculos. Dentro de los métodos para resolver este problema se encuentran los asociados a listas de prioridad [5], métodos de programación lineal y dinámica [6], métodos de relajación lagrangiana [7–9] y métodos de programación lineal entera mixta (*Mixed Integer Linear Programming*, MILP) [10], siendo estos últimos los que han mostrado mejores resultados en los últimos años, debido a que se basan en la aplicación del método de *Branch & Bound*, lo cual resulta eficiente si no hay muchas variables enteras [11].

A medida que van creciendo los sistemas de potencia, se complejiza la resolución del problema de UC, debido a la carga computacional que implica. A su vez, esto se complica aún más en sistemas con una fuerte componente hídrica, debido a que el horizonte de cálculo

aumenta por causa de la gestión de energía de los embalses, la incertidumbre hidrológica, las restricciones asociadas a las distintas series hidrológicas, y al modelamiento de los usos alternativos del agua. Esto ha hecho que se planteen distintas alternativas para su resolución, o bien, se plantee un modelo complementario que permita resolver el problema de UC. Algunas alternativas son los métodos de *Data Driven* [12–14] o que incorporen modelos de *Machine Learning* (ML) [15–18], los cuales se han vuelto populares en el último tiempo. Por otro lado, la incorporación de redes neuronales artificiales (*Artificial Neural Networks*, ANN) junto a aprendizaje reforzado (*Reinforcement Learning*, RL) [19–24] han mostrado un avance significativo para resolver problemas de optimización.

El CEN dentro de su política de operación [25] considera en su programa un predespacho de la generación el día previo con un horizonte de 24 horas (programación diaria), pero planteando en el problema matemático un horizonte de 168 horas debido a la gestión de los embalses. Para realizarlo se toma en cuenta la predicción de la demanda en el periodo de tiempo considerado, los pronósticos de la energía renovable (hidro, solar y eólica) y las restricciones operativas vigentes.

Este trabajo es parte de una iniciativa del CEN, como encargado de preservar la seguridad del sistema eléctrico chileno, de desarrollar un modelo en base a inteligencia artificial para asistir en la programación diaria de energía y servicios complementarios. En ese trabajo se propone utilizar datos históricos de entrada y salida de la herramienta `PLEXOS`, empleado actualmente para realizar la programación diaria, junto con data real de operación, con el fin de entrenar el modelo a desarrollar. El objetivo final es que el modelo propuesto sea una alternativa en el proceso de la programación diaria, con tal de disminuir los tiempos de computo y obtener resultados que se acerquen más a la realidad. Debido a esto, los modelos de resolución del problema de UC en el SEN, la data histórica y la capacidad de cálculo son provistos por el CEN.

1.2 Motivación

Con el fin de llegar a una participación de energías renovables del 100%, se necesitan desarrollar herramientas más avanzadas en la operación de los sistemas eléctricos, con tal de lograr una modelación más precisa y con tiempos de simulación más eficientes. El creciente aumento del tamaño del sistema junto a un horizonte de modelación semanal debido al uso de embalses, ha vuelto cada vez más complejo la resolución del problema de UC en el SEN.

Para el SEN, en la programación diaria se considera una formulación MILP con un horizonte de siete días por la gestión semanal de los embalses, donde los primeros tres días se considera una modelación más detallada, tomando en cuenta restricciones de servicios complementarios y bloques horarios. A partir del cuarto día se relajan el problema con tal de reducir la carga computacional. El creciente aumento de las ERV en la matriz energética ha hecho que las restricciones de flexibilidad operacional, como lo son las rampas, se vuelvan cada vez más importantes. En la actualidad el tiempo de resolución del problema es variable y depende de varios factores, pero en general el rango de tiempo varía de 20 a 90 minutos, donde en casos excepcionales el tiempo puede ser mayor. El CEN considera restricciones temporales que modelan las solicitudes de trabajo, como desconexión o limitación, tanto de centrales como líneas de transmisión. Existen por otro lado restricciones operativas de centrales, como variación máxima de carga horaria, y restricciones operativas de embalses, como volúmenes límite al final del horizonte de planificación. Para los

recursos de generación se consideran restricciones de gas, agua y ERV, tomando en cuenta la conectividad del sistema asociado a las series hidráulicas. Por último, se co-optimiza para la prestación de servicios complementarios en el sistema.

El desarrollo de un modelo complementario basado en redes neuronales artificiales para el problema de predespacho de unidades, puede disminuir significativamente los tiempos de resolución, fijando variables binarias para resolver un problema con un espacio de soluciones reducido con una formulación MILP, y obteniendo una solución dentro de los límites de integridad del *solver*.

1.3 Hipótesis

Es posible reducir significativamente los tiempos de resolución para el problema de *Unit Commitment* en sistemas hidrotérmicos, particularmente el chileno, sin alterar significativamente la calidad de la solución, fijando variables mediante un modelo multi-agente de RL basado en ANN y resolviendo el problema con un espacio de soluciones reducido mediante una formulación MILP.

1.4 Objetivos

1.4.1 Objetivo general

- Implementar un esquema *offline/online* basado en un modelo multi-agente de RL haciendo uso de ANN y la formulación MILP, con tal que permita reducir los tiempos de cálculo del problema de UC que se resuelve actualmente en el SEN sin alterar significativamente la calidad de la solución. Se busca que el modelo multi-agente obtenga de manera *offline* variables de decisión del problema de UC mediante ANN entrenadas en base a data histórica de operación, para después de manera *online* formular mediante MILP el problema con un espacio de soluciones reducido y obtener el costo de operación junto al despacho de las distintas unidades.

1.4.2 Objetivos específicos

- Identificar dentro de la data histórica de operación de la programación diaria del SEN parámetros que se relacionen con el comportamiento de un sistema hidrotérmico, y que puedan servir de entrada para una ANN que permita obtener variables de decisión del problema de UC.
- Diseñar e implementar un modelo multi-agente de RL basado en ANN usando una *policy* basada en gradiente, que tome como entrada parámetros característicos de un sistema hidrotérmico y obtenga como salida variables de decisión del problema de UC.
- Con los resultados del modelo *offline*, formular mediante MILP el problema de UC con un espacio de solución reducido, para luego determinar el costo de operación y los despachos de las distintas centrales.
- Probar y comparar el desempeño de la formulación MILP que se usa actualmente en el SEN frente al esquema *offline/online* propuesto, tomando en cuenta para esto el tiempo de cómputo y el costo total de operación.

1.5 Contribuciones al área

1.5.1 Contribución científica del trabajo

Si bien existen trabajos que incorporan técnicas de ML para resolver el problema de UC, o bien, plantear un modelo complementario, todos ellos tienen aplicabilidad en sistemas térmicos. Este trabajo, por un lado, propone introducir un modelo basado en ANN y RL que sirva como complemento al problema de UC, enfocado especialmente en sistemas hidrotérmicos, que permita fijar variables de decisión y resolver un problema con un espacio de solución reducido, con tal de disminuir significativamente los tiempos de cómputo y obtener una solución dentro de los límites de integralidad del *solver*. Por otro lado, se busca introducir una nueva metodología para la resolución del problema de predespacho de unidades, con tal de usar un modelo de ML entrenado en base a datos históricos de la programación diaria.

1.5.2 Contribución para el CEN

Este trabajo busca desarrollar un modelo basado en ANN y RL el cual logre resolver problemas que existen actualmente en el CEN. Los tiempos de resolución para la programación diaria han ido aumentando gradualmente a medida que ha ido creciendo el sistema y la variabilidad de este debido al gran aumento de la ERV en la matriz energética. Con la implementación de esta herramienta se logrará resultados más cercanos a la realidad junto a una disminución del tiempo de cómputo empleado.

El desarrollo de este modelo traerá mejoras a la resolución del problema de UC relacionado a la eficiencia computacional. Su correcta implementación podrá permitir a los operadores tener enormes beneficios económicos y mejorar la eficiencia del mercado, pudiendo considerar un modelo más preciso del sistema, además de poder tomar en cuenta restricciones que no se modelan actualmente y aumentar los horizontes de tiempo de estudio.

En la actualidad debido a la complejidad del problema y a la incertidumbre que añaden las ERV, existen diferencias respecto a lo programado con la operación real especialmente relacionado con los servicios complementarios. Este trabajo podría dar solución en parte a este problema, debido a que disminuirá la carga computacional que se tienen actualmente, lo que permitirá modelar de mejor manera restricciones de rampa y servicios complementarios.

1.6 Estructura del documento

El presente documento se organiza de la siguiente manera:

- Capítulo 2: Marco teórico y estado del arte.
  Se presenta una revisión del problema de UC en sistemas hidrotérmicos y explora la aplicación de ANN en este ámbito. Además, se expone el estado del arte junto a las principales contribuciones de este trabajo.
- Capítulo 3: Metodología.
  Se describe la metodología propuesta para el desarrollo de este trabajo de tesis y las consideraciones que se tuvieron en cuenta para entrenar las ANN.
- Capítulo 4: Caso de estudio.
  Se presenta el sistema eléctrico en el cual será probado el modelo propuesto, junto a los principales resultados.
- Capítulo 5: Conclusiones.

Capítulo 2

Marco teórico y estado del arte

2.1 Modelación matemática UC

El problema de UC puede variar dependiendo del detalle en la modelación del sistema y los requerimientos operacionales a los que esté sujeto, sin embargo, se puede considerar una formulación de tipo MILP de manera genérica:

$$\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{y}} \quad & \boldsymbol{c}^T\boldsymbol{x} + \boldsymbol{d}^T\boldsymbol{y} && \text{(2.1a)}\\
\text{subject to} \quad & A\boldsymbol{x} \leq \boldsymbol{b}, && \text{(2.1b)}\\
& H\boldsymbol{y} \leq \boldsymbol{h} && \text{(2.1c)}\\
& G\boldsymbol{x} + E\boldsymbol{y} \leq \boldsymbol{g} && \text{(2.1d)}\\
& \boldsymbol{x} \in \{0,1\}^{|\mathcal{G}|\times|\mathcal{T}|} && \text{(2.1e)}
\end{aligned}$$

El vector $\boldsymbol{x}$ representa la variables binarias relacionadas con los estados (encendido/apagado) de las centrales térmicas para el periodo de horizonte de planificación. El vector $\boldsymbol{y}$ representa las variables continuas relacionadas con la potencia despachada y reservas de cada unidad en el periodo de tiempo de análisis. Los conjuntos $\mathcal{G}$ y $\mathcal{T}$ contienen los índices que identifican cada generador y el periodo de tiempo, respectivamente. Cabe destacar que en esta formulación las funciones de costos y restricciones se han escrito de forma lineal.

La función objetivo busca minimizar los costos de producción de las unidades, considerando los costos de combustible y de transición (encendido y apagado), además de los costos de potencia no suministrada. El término $\boldsymbol{c}^T\boldsymbol{x}$ representa los costos de encendido y apagado de los generadores, además de los costos fijos de generación. El término $\boldsymbol{d}^T\boldsymbol{y}$ corresponde a los costos de despacho más los costos de potencia no suministrada.

La restricción (2.1b) incluye los tiempos mínimos de encendido y apagado. La restricción (2.1c) contiene los límites de rampa de los generadores y la restricción (2.1d) incluye los requerimientos de balance de potencia, reserva de las unidades y límites de generación de estas. Cabe destacar que en el problema también se pueden considerar restricciones asociadas a centrales renovables, asumiendo un nulo costo marginal, restricciones de transmisión usando una aproximación DC, o restricciones asociadas a balances energéticos [11].

Este problema presenta como resultado un conjunto factible no convexo debido a las restricciones que incluyen variables binarias. Se han investigado diversas técnicas para mejo-

rar el desempeño computacional del problema, donde destacan la búsqueda de restricciones redundantes [17, 26–28] y métodos de descomposición [8, 9, 29].

2.1.1 Consideraciones del UC en sistemas hidrotérmicos

Al considerar un sistema hidrotérmico se deben tener en cuenta una serie de restricciones propias de las unidades hidroeléctricas. Estas se subdividen dependiendo del tipo de central que se está modelando, tales como: centrales de tipo embalse, centrales de tipo serie o centrales de pasada.

En Chile, el programa de operación del CEN [25], considera la coordinación hidrotérmica, la cual consiste en la programación óptima de los recursos energéticos disponibles, minimizando los costos de operación, cumpliendo con las restricciones operacionales, e incluyendo el uso del recurso hidráulico en las cuencas y embalses del sistema. La coordinación hidrotérmica puede ser de largo plazo, considerando meses o varios años, o bien, de corto plazo, modelando un día o una semana. Para la modelación se deben considerar como datos básicos la demanda, unidades disponibles, afluentes de agua, niveles de combustible, estado final y convenios de riego.

En términos generales, la coordinación y el uso adecuado del agua (energía almacenada) se logra a través del encadenamiento jerarquico de modelos probabilísticos de largo, mediano y corto plazo. A medida que va disminuyendo el horizonte de tiempo se detallan de mejor manera algunas características importantes del problema. El modelo de corto plazo es el que determina la cantidad de agua que se usará de los embalses en cada etapa del horizonte de planificación, obteniendo los valores del agua y construyendo la política de operación del sistema para una semana. Esta incluye la energía a colocar por cada una de las centrales de embalse, y la manera en que estas se reparten los aumentos y disminuciones de consumo para la semana bajo estudio. A partir de la política de operación del modelo de corto plazo, se realiza el predespacho o programación diaria. El CEN considera un horizonte de tiempo para el problema de UC de una semana debido a la gestión de los embalses, incrementando de esta forma el número de variables binarias involucradas. De todas formas, solo los tres primeros días cuentan con una modelación detallada, relajando el problema los días siguientes para disminuir la carga computacional.

2.2 Redes neuronales artificiales

El desarrollo de la inteligencia artificial y técnicas de ML han tenido avances significativos dentro de los últimos años, no solo en el área de la energía, sino que en todo ámbito de la ciencia. Las ANN son una técnica dentro del área de ML, más especificamente dentro de lo que se conoce como *Deep Learning* (DL), que ha sido promisorio en modelar problemas con altos grados de no linealidad [30].

A grandes rasgos, una red neuronal es un grafo de computación que permite transformar datos de entrada en datos de salida, es decir, la red neuronal define un mapeo de $y = f(x; \theta)$ y aprende los valores de los parámetros θ que resulta en la mejor aproximación para la función, dado un conjunto de ejemplos de entrenamiento (x, y).

Las redes neuronales y el mundo del DL han demostrado que tienen potencial para asistir en los problemas de optimización [31–33]. A continuación se presentan tres tipos de redes neuronales y su potencial aplicación en optimización.

2.2.1 Feedforward neural networks

Una *feedforward neural network* (FNN) [34] es una red neuronal donde la conexión entre las distintas neuronas no forman un ciclo, la información se procesa en una sola dirección, por lo que no hay retroalimentación entre las distintas conexiones. Las FNN pueden descifrar complicadas correlaciones no lineales entre los distintos parámetros de la red. Este tipo de redes ha demostrado buenos resultados en problemas relacionados a UC [13, 15, 16] y también en problemas en otros ámbitos de la ingeniería como lo son los procesos químicos [35, 36].

2.2.2 Convolutional neural networks

Las *Convolutional Neural Networks* (CNN) [37] es una de las redes más populares dentro del mundo del DL. Sus principales aplicaciones radican en clasificación de imágenes y procesamiento de lenguaje, mostrando en estas tareas un excelente desempeño [38]. La arquitectura de una CNN típicamente consiste en capas convolucionales, capas no-lineales y capas de agrupamiento. Estas redes son diseñadas para desarrollar las ideas de conectividad local y pesos compartidos, con tal de extraer los mapeos de características a través de convoluciones y no sobrecargar la red con parámetros entrenables. Las capas de convolución son la esencia de una CNN. En estas capas, se aplican filtros a la entrada (generalmente una imagen) mediante operaciones de convolución. Un filtro es generalmente una matriz o un conjunto de parámetros que se utiliza para realizar la operación de convolución en la entrada. La convolución estándar entre una matriz de entrada X y un filtro K se define en la ecuación (2.2).

$$(X * K)_{i,j} = \sum_{m} \sum_{n} X_{i+m,j+n} \cdot K_{m.n}. \tag{2.2}$$

Donde i,j son las coordenadas de la posición en la que se está calculando el resultado, y m,n son los índices de la matriz del filtro.

Si bien los mejores resultados de este tipo de red se han presentado en el procesamiento de imágenes, también se han usado para aprender las características espaciales en problemas de relacionados a flujo vehicular [19] y para obtener pronósticos de demanda eléctrica [39].

2.2.3 Recurrent neural networks

Las *recurrent neural networks* (RNN) son redes especializadas en el procesamiento de secuencias, siendo capaces de tener en cuenta lo que han procesado y generar una respuesta en función la entrada en un determinado momento como de sus estados anteriores [40]. A diferencia de una FNN, una RNN es capaz de retener un estado que puede representar información desde una ventana de contexto de largo arbitrario. Sin embargo, las RNN tradicionales han mostrado dificultades para el entrenamiento por el problema del gradiente desvaneciente, el cual consiste en el no aprendizaje de la red neuronal cuando es muy profunda al aplicar métodos de aprendizaje basados en gradiente descendiente, debido a que el valor de gradiente cada vez es más cercano a cero, lo cual no permite la actualización de los pesos de la red en cada iteración. Para este problema se han propuesta nuevas arquitecturas para las RNN, como lo son las *long short-term memory* (LSTM) [41], que incorporan compuertas de entrada, salida y olvido que mejoran la capacidad para memorizar secuencias largas para data secuencial. En optimización secuencial bajo incertidumbre, se

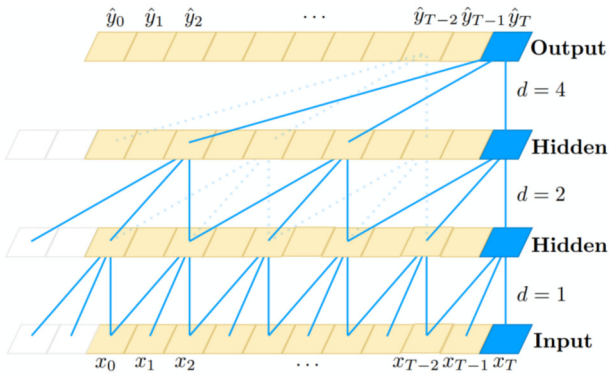


Figura 2.1. *Convoluciones causales dilatadas con factor de dilatación* $d = 1, 2, 4$ *y tamaño de filtro* 3. *[42]*

usa data histórica para modelar parámetros inciertos. Integrar DL y optimización multi-etapa bajo incertidumbre es un área de investigación prometedora [23].

2.2.4 Temporal convolutional neural network

Las *Temporal Convolutional Neural Networks* (TCN) [42] son una familia de arquitecturas dentro de las CNN que se destacan por poseer una capacidad mayor de memoria comparada a las clásicas redes recurrentes. Estas poseen varias características importantes. Por un lado, usan convoluciones causales, por lo que se previene la conexión de información del futuro al pasado, además de que el largo de la salida es igual al largo de la entrada. Las redes se caracterizan por ser capaces de capturar patrones largos debido al uso de convoluciones dilatadas [42], lo que permite tener un campo receptivo exponencialmente grande. Esto significa que una salida en una capa superior puede representar un rango más amplio de entradas. A diferencia de la convolución normal vista en (2.2), en estas se incluye un factor de dilatación d. Una convolución dilatada entre una matriz de entrada X y un filtro K se define en la ecuación (2.3).

$$(X *_d K)_{i,j} = \sum_m \sum_n X_{i+d\cdot m, j+d\cdot n} \cdot K_{m,n} \tag{2.3}$$

Donde d es el factor de dilatación, i,j son las coordenadas de la posición en la que se está calculando el resultado, y m,n son los índices de la matriz del filtro. Las convoluciones dilatadas permiten que los filtros tengan campos receptivos más grandes sin aumentar el número de parámetros en la red, lo que puede ser útil para capturar información contextual a diferentes escalas. En la Figura 2.1 se ilustra capas convolucionales que incluyen distintos factores de dilatación.

Por otro lado, para aumentar la profundidad de estas redes, lo que se hace generalmente es concatenar distintos bloques de redes TCN. Para esto se incluyen conexiones residuales [43] a la salida de cada bloque, donde se ha demostrado que ayuda en el procesos de aprendizaje cuando las redes se vuelven muy profundas. A medida que aumenta la profundidad de una red neuronal, puede ocurrir el problema de desvanecimiento del gradiente [44], donde agregar capas adicionales a la red lleva a un empeoramiento del rendimiento. Las conexiones residuales permiten que la información fluya directamente a través de las capas sin ninguna transformación, lo que ayuda a evitar este problema al facilitar la optimización y el aprendizaje de representaciones más profundas. En la Figura 2.2 se ilustra un ejemplo de bloque residual para una TCN.

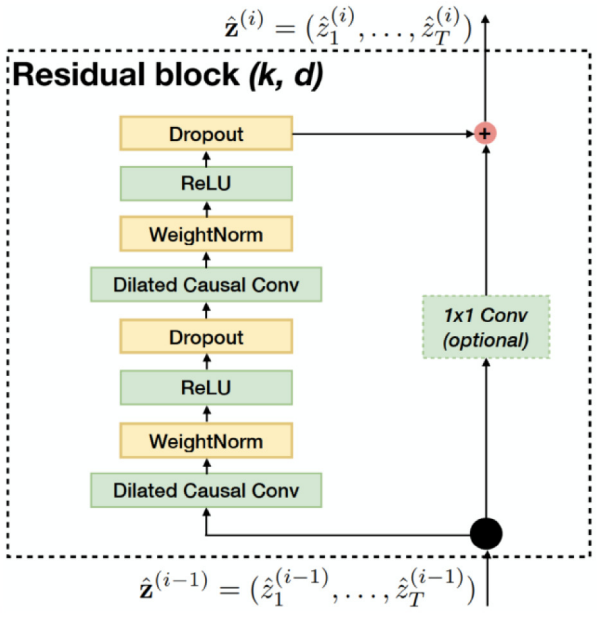


Figura 2.2. *Bloque residudal TCN [42]*

Una TCN presenta la ventaja respecto a las RNN, que a diferencia de estas últimas, no procesa la secuencia de entrada de manera secuencial, sino que permite procesar largas secuencias de manera paralela. También, las TCN son entrenadas con el algoritmo estándar de *backpropagation*, evitando los problemas de gradiente que se pueden generar con el algoritmo de *backpropagation-through-time* (BPTT) [45] que usan las RNN.

2.3 Reinforcement Learning

Reinforcement learning [46] es una técnica dentro del área del ML basadas en un sistema de prueba y error, donde un agente informático aprende a realizar cierta tarea dentro de un entorno dinámico con tal de maximizar una métrica de recompensa por la tarea hecha, sin intervención humana y sin estar programado explícitamente para completar la tarea. RL posee cuatro elementos esenciales:

- **Agente:** Es el programa que se entrena con el objetivo de efectuar una tarea.
- **Ambiente:** Es el entorno donde el agente realiza las labores.
- **Acción:** Son los movimientos efectuados por el agente, los cuales tienen un efecto en el ambiente.
- **Recompensa:** Es la evaluación de la acción realizada por el agente, la cual puede ser positiva o negativa.

En la Figura 2.3 se muestra un esquema clásico de RL para la interacción entre agente y ambiente.

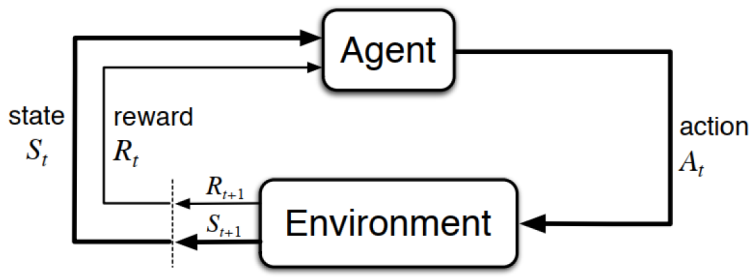


Figura 2.3. *Interacción agente-ambiente en RL [46]*

Las ANN entrenadas con RL han demostrado una gran capacidad para codificar comportamientos complejos. Dentro de las aplicaciones más notables en las que participa RL están: automatización de procesos robóticos, conducción automática de vehículos y videojuegos.

De manera general, RL busca que el agente aprenda una *policy* π que maximice las recompensas. Dentro de esto, existen dos principales estrategias para resolver problemas de RL: (i) métodos basados en *value functions* y métodos basados en *policy search*. De todas formas, también existen métodos *actor-critic* híbridos que hacen uso de ambos. A continuación se describe uno de los algoritmos con mejores resultados para entrenar ANN usando RL.

2.3.1 Deep Deterministic Policy Gradient

Deep deterministic policy gradient (DDPG) [47] es un algoritmo *off-policy actor-critic* que puede aprender *policies* en espacios continuos y de alta dimensionalidad. El método considera un espacio estándar de RL, con un agente interactuando con un ambiente E en tiempos discretos. En cada tiempo t el agente recibe una observación s_t, para tomar una acción a_t y recibir un escalar de recompensa r_t.

El comportamiento del agente es definido por una *policy* π, la cual mapea los estados s_t hacia una distribución de probabilidad sobre las acciones $\pi : \mathcal{S} \rightarrow \mathcal{P}(\mathcal{A})$. La conducta tomada por el agente se modela como un proceso de decisión Markov (*Markov Decision Process*, MDP) con un espacio de estados $\mathcal{S}$, un espacio de acciones $\mathcal{A} = \mathbb{R}^N$, un estado inicial de distribuciones $p(s_1)$, transiciones dinámicas $p(s_{t+1}|s_t, a_t)$, y una función de recompensa $r(s_t, a_t)$.

La respuesta desde un estado es definida como la suma de la recompensa futura descontada $R_t = \sum_{i=t}^{T} \gamma^{i-t} r(s_i, a_i)$ con un factor de descuento $\gamma \in [0, 1]$. El objetivo de RL es aprender una *policy* π que maximice la esperanza de la respuesta desde una distribución inicial $J = \mathbb{E}_{r_i, s_i \sim E, a_i \sim \pi}[R_1]$. La distribución para el estado de descuento de una *policy* π se denota como ρ^π.

La función *action-value* es usada en muchos algoritmos de RL. Esta describe la respuesta esperada luego de tomar una acción a_t en un estado s_t después de seguir una *policy* π:

$$Q^\pi(s_t, a_t) = \mathbb{E}_{r_{i \geq t}, s_{i>t} \sim E, a_{i>t} \sim \pi}[R_t | s_t, a_t] \tag{2.4}$$

Muchas aproximaciones en RL hacen uso de la relación recursiva conocida como la ecuación de Bellman:

$$Q^\pi(s_t, a_t) = \mathbb{E}_{r_t, s_{t+1} \sim E}\left[r(s_t, a_t) + \gamma \mathbb{E}_{a_{t+1} \sim \pi}[Q^\pi(s_{t+1}, a_{t+1})]\right] \tag{2.5}$$

Si la *policy* es determinística se puede describir una función $\mu : \mathcal{S} \rightarrow \mathcal{A}$ y simplificar la expresión:

$$Q^\mu(s_t, a_t) = \mathbb{E}_{r_t, s_{t+1} \sim E}\left[r(s_t, a_t) + \gamma Q^\mu(s_{t+1}, \mu(s_{t+1}))\right] \tag{2.6}$$

De esta forma, el valor esperado solo depende del ambiente E. Esto significa que es posible aprender una función Q^μ *off-policy*, usando transiciones generadas desde un comportamiento estocástico diferente dado por una *policy* β.

Dentro de los algoritmos *off-policy* usados en RL, uno de los más usados es Q-Learning [48], la cual hace uso de una *greedy policy* $\mu(s) = \arg\max_a Q(s,a)$. En DDPG se consideran aproximadores de función parametrizados por θ^Q, los cuales son optimizados minimizando la función de pérdida L:

$$L(\theta^Q) = \mathbb{E}_{s_t \sim \rho^\beta, a_t \sim \beta, r_t \sim E}\left[(Q(s_t, a_t|\theta^Q) - y_t)^2\right] \tag{2.7}$$

donde y_t se define como:

$$y_t = r(s_t, a_t) + \gamma Q(s_{t+1}, \mu(s_{t+1})|\theta^Q) \tag{2.8}$$

Si bien y_t también depende de θ^Q, por lo general esto es ignorado.

El uso de grandes aproximadores de funciones parametrizados para aprender la función *action-value* ha sido evitado en el pasado debido a que no es posible garantizar de manera teórica que el método obtendrá un buen rendimiento, además de que en el aprendizaje se tiene a inestabilizar. Sin embargo, estos problemas se han ido atenuando por dos cambios propuestos en [49] para escalar el método Q-Leaning, los cuales consisten en usar de un *replay buffer* y de una *target network* separada para calcular y_t.

El algoritmo DDPG mantiene una *actor function* parametrizada $\mu(s|\theta^\mu)$ que especifica una *policy* mapeando estados determinísticamente para una acción específica. $Q(s,a)$ es aprendida mediante la ecuación de Bellman como en Q-Learning. La función $\mu(s|\theta^\mu)$ es actualizada aplicando la regla de la cadena a la respuesta esperada desde la distribución J para los parámetros de la función:

$$\begin{aligned} \nabla_{\theta^\mu} J &\approx \mathbb{E}_{s_t \sim \rho^\beta}\left[\nabla_{\theta^\mu} Q(s, a|\theta^Q)|_{s=s_t, a=\mu(s_t|\theta^\mu)}\right] \\ &= \mathbb{E}_{s_t \sim \rho^\beta}\left[\nabla_a Q(s, a|\theta^Q)|_{s=s_t, a=\mu(s_t)} \nabla_{\theta_\mu} \mu(s|\theta^\mu)|_{s=s_t}\right] \end{aligned} \tag{2.9}$$

Uno de los inconvenientes al usar ANN para RL es que la mayoría de los algoritmos de optimización asumen que las muestras son independientes e idénticamente distribuidas. Como las muestras son generadas explorando secuencialmente en un ambiente, este supuesto no se sostiene.

Para lidiar con este inconveniente se usan los *replay buffer*. Estos son depósitos $\mathcal{R}$ de tamaño finito. Las transiciones se muestrean desde el ambiente de acuerdo a la *policy* de exploración, almacenando (s_t, a_t, r_t, s_{t+1}) en el *replay buffer*. Cuando el *buffer* se llena, las muestras anteriores se descartan. En cada paso $\mu(s, \theta^\mu)$ y $Q(s,a)$ se van actualizando muestreando un *minibatch* uniformemente desde el *buffer*.

Al implementar de manera directa Q-Learning en la ecuación (2.7) con ANN, se obtiene inestabilidad en muchos ambientes. Al actualizar $Q(s, a|\theta^Q)$ la red también lo usó para actualizar el *target value* de la ecuación (2.8), por lo que Q es propensa a caer en divergencia. DDPG usa la idea propuesta en [49], al crear copias para Q y μ, definidas como

$Q'(s, a|\theta^{Q'})$ y $\mu'(s, a|\theta^{\mu'})$, respectivamente, los cuales se usan para calcular los *target values*. Los pesos de estas redes son actualizados haciendo que sigan de manera lenta las redes aprendidas: $\theta' = \tau\theta + (1 - \tau)\theta'$ donde $\tau \ll 1$. De esta forma, los *target values* están restringidos a cambiar de manera lenta, mejorando la estabilidad en el entrenamiento. El método DDPG propuesto en [47] se describe en el algoritmo 2.1.

Algorithm 2.1 DDPG algorithm [47]

Randomly initialize critic network $Q(s, a|\theta^Q)$ and actor $\mu(s|\theta^\mu)$ with weights θ^Q and θ^μ.
Initialize target network Q' and μ' with weights $\theta^{Q'} \leftarrow \theta^Q, \theta^{\mu'} \leftarrow \theta^\mu$
Initialize replay buffer R
for episode = 1, M **do**
Initialize a random process $\mathcal{N}$ for action exploration
Receive initial observation state s_1
for $t = 1$, T **do**
Select action $a_t = \mu(s_t|\theta^\mu) + \mathcal{N}_t$ according to the current policy and exploration noise
Execute action a_t and observe reward r_t and observe new state s_{t+1}
Store transition (s_t, a_t, r_t, s_{t+1}) in R
Sample a random minibatch of N transitions (s_i, a_i, r_i, s_{i+1}) from R
Set $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$
Update critic by minimizing the loss: $L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$
Update the actor policy using the sampled policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum_i \nabla_a Q(s, a|\theta^Q)|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)|_{s_i}$$

Update the target networks:

$$\theta^{Q'} \leftarrow \tau\theta^Q + (1 - \tau)\theta^{Q'}$$

$$\theta^{\mu'} \leftarrow \tau\theta^\mu + (1 - \tau)\theta^{\mu'}$$

end for
end for

2.4 Estado del arte

Ha habido numerosas propuestas para mejorar el rendimiento computacional en el problema de UC centradas en el uso de datos históricos de operación de sistemas eléctricos. Una revisión de las tendencias actuales de ML aplicadas al problema de UC se muestra en [50]. Los primeros trabajos que incorporaron el uso de ANN para resolver el problema de UC datan de la década de 1990. En [51] y [52], se utilizaron ANN para resolver el problema de UC de manera simplificada. Específicamente, [51] utilizó una red neuronal con perfiles de carga en las barras como entrada y el estado de 26 unidades como salida, mientras que [52] empleó redes de Hopfield para un sistema con 17 unidades térmicas. En [15], se presentó un modelo basado en una FNN de tres capas para estimar el estado de las unidades utilizando perfiles de carga como entrada. Posteriormente, el problema de UC se resolvió utilizando un método de *simulated annealing*, un algoritmo heurístico de optimización que busca soluciones cercanas a óptimas en problemas de optimización combinatorial. Estos trabajos demostraron resultados prometedores en términos de mejorar los tiempos de resolución, pero solo se probaron en sistemas de prueba a pequeña escala con pocas unidades térmicas.

Trabajos más recientes han intentado utilizar enfoques basados en datos para encontrar patrones en el problema de optimización y tratarlo como un problema de clasificación. En [14], se propone una reducción de variables a la formulación clásica MILP para el problema de *Security Constrained Unit Commitment* (SCUC). Con un esquema de cálculo *offline/online*, el modelo *offline* obtiene las soluciones de las variables binarias, mientras que el modelo *online* resuelve el problema SCUC reducido con la formulación MILP. En este trabajo, se utiliza el método *K-means* para clasificar la demanda neta de carga en las barras y encontrar soluciones factibles para el estado de las unidades. En la Figura 2.4 se muestra el *flowchart* del esquema propuesto. Otra tendencia que se ha observado en los últimos años es la aplicación de ML para mejorar el rendimiento de los MILP *solvers* basados en la técnica de *Branch & Bound*. En [53], se utiliza ML para mejorar la estrategia de ramificación. El enfoque sugerido consta de dos fases principales. En primer lugar, se extraen características para representar la ramificación de una variable dentro de un nodo del árbol. Luego, se obtienen las decisiones de ramificación resolviendo un conjunto de instancias de entranemiento y, posteriormente, se entrena un regresor para predecir el puntaje estimado de ramificación en los nodos.

En [17], se proponen tres estrategias basadas en ML para extraer características del problema SCUC y resolver el problema reducido utilizando la formulación MILP. En su enfoque, utilizan un algoritmo *K-Nearest Neighbors* (kNN) para determinar qué restricciones deben incluirse inicialmente en la primera iteración de la resolución de UC y cuáles deben excluirse. Por otro lado, otro desafío en la resolución de problemas SCUC es obtener soluciones iniciales factibles de alta calidad. Para abordar esto, los autores proponen un predictor de soluciones utilizando el método kNN como *warm start* para el MILP *solver*. Finalmente, introducen un predictor de sub-espacio afín basado en SVM para reducir el área de búsqueda. Los resultados predichos mejoraron significativamente el tiempo de ejecución y la calidad de la solución del MILP *solver*. Los tiempos de solución se redujeron 4.3 veces en sistemas de 1888 y 6515 barras.

Algunos estudios han examinado diferentes algoritmos de aprendizaje ML para predecir el estado encendido/apagado de las unidades, incluyendo técnicas de factibilidad para evaluar la viabilidad de las soluciones. Estos estudios también han utilizado perfiles de demanda

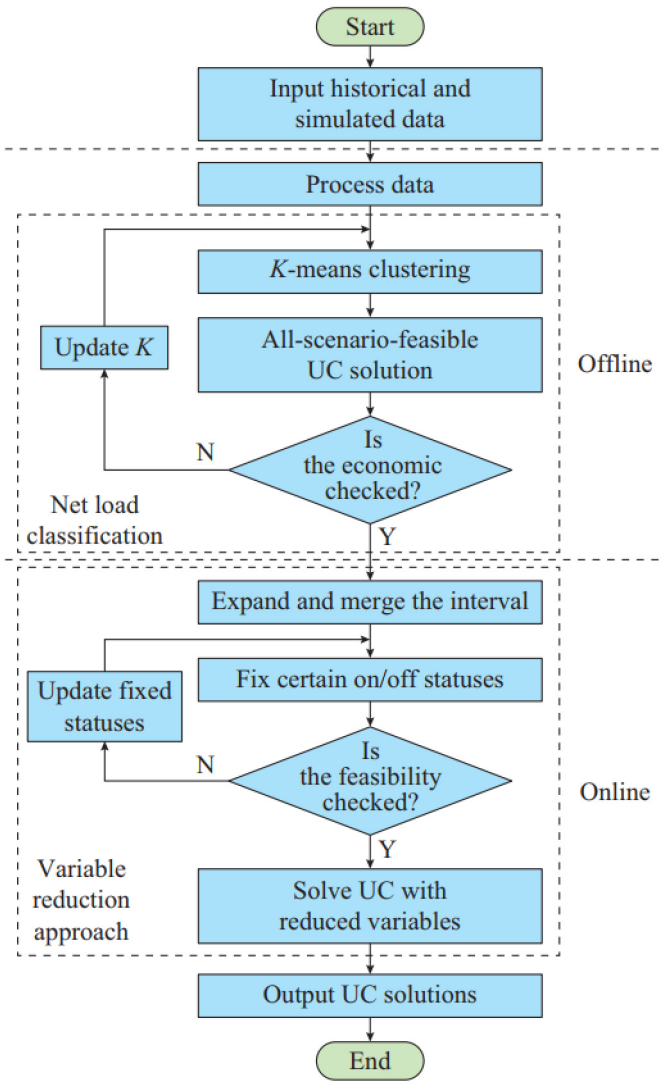


Figura 2.4. *Flowchart propuesto en [14]*

como entradas para entrenar los algoritmos de ML. Por ejemplo, en [54], se utilizan regresión logística (*Logistic Regression, LR*), ANN, bosques aleatorios (*Random Forest, RF*) y kNN para obtener un problema reducido de SCUC. La precisión de los resultados se valida utilizando diversos casos de prueba de sistemas de potencia, como el sistema IEEE 24-bus, IEEE 73-bus, IEEE 118-bus, el sistema South Carolina Synthetic Grid 500-bus y el sistema Polish 2383-bus. Los hallazgos demuestran una alta precisión de entrenamiento y la incorporación de una capa de factibilidad para abordar soluciones inviables. Los resultados muestran una significativa aceleración para el problema de UC en todos los sistemas probados, con *speed-up* que van de 3.4 a 3.6 en promedio. Incluso en sistemas más grandes como el sistema Polish 2383-bus se obtienen tasas de 6.9 y 8.5. De manera similar, en [18], se realizaron experimentos en el sistema IEEE 24-bus y un sistema práctico de 5655 barras utilizando algoritmos kNN, ANN, RF y árboles de decisión (*Decision Trees,* DT). Ciertos algoritmos superaron a otros en precisión de predicción, observándose al menos un 40% de mejora computacional para cada algoritmo mientras se mantenía la variación de costos dentro del 1%. Los autores concluyeron, basándose en los resultados de diversos modelos, que aprender el estado encendido/apagado de las unidades podría no requerir modelos tan sofisticados. En [55], los autores se enfocaron en analizar patrones de datos históricos de sistemas eléctricos, incorporando un modelo basado en *Graph Neural Networks* (GNN) y LSTM. Validaron sus hallazgos en diversos casos de prueba, como el sistema IEEE 24-bus, IEEE 73-bus, IEEE 118-bus y el sistema South Carolina Synthetic Grid 500-bus. Una de las contribuciones significativas de su investigación es la aplicación de la capa basada en GNN, aprovechando las ventajas de las estructuras basadas en grafos. El enfoque propuesto mostró ahorros significativos de tiempo, con reducciones que van del 20% al 50%,

dependiendo del caso de estudio específico. Estos trabajos obtuvieron buenos resultados en diversos modelos, pero no estudian las consideraciones a tener en cuenta cuando el sistema eléctrico tiene una fuerte componente hidroeléctrica.

Los trabajos mencionados muestran un enfoque común en el entrenamiento de modelos ML utilizando perfiles de carga o demanda neta como entradas. Estas entradas consideran los patrones de consumo de electricidad y tienen en cuenta la contribución de fuentes de energía renovable. Sin embargo, estos trabajos se han centrado en sistemas térmicos sin considerar sistemas con una fuerte componente hidroeléctrica, como el sistema eléctrico chileno. En estos sistemas, la programación de la operación está estrechamente vinculada a la gestión de los embalses de agua, lo que introduce desafíos únicos en el problema de UC, como horizontes más largos y el consiguiente aumento en el número de variables de decisión. La operación de las centrales térmicas en estos sistemas depende de cómo se gestionan los embalses de agua. Esto está estrechamente relacionado con el hecho de que la mayoría de los trabajos consideran modelos de ML, como kNN, RF o SVM [16,17,54,55], para tratar el problema como uno de clasificación. Sin embargo, no se ha explorado el uso de RL con un modelo multiagente para obtener variables de decisión relevantes para el problema de optimización, lo cual sería prometedor para reducir los tiempos de solución. Los avances recientes en técnicas de ML han demostrado el potencial de RL combinado con ANN para la toma de decisiones secuenciales. Esta integración resuelve eficazmente problemas complejos del mundo real al facilitar interacciones con el entorno y aprender de las señales de recompensa. Por ejemplo, en [24], se propone un esquema *offline/online* para obtener niveles de potencia despachados junto con niveles de voltaje en un problema de Flujo de Potencia Óptimo (*Optimal Power Flow,* OPF). Se utiliza RL para proponer una función de costo que incluya restricciones operativas. El *flowchart* propuesto se muestra en la Figura 2.5. En [56], se propone un esquema MADRL, basado en ANN y RL, para operar una planta fotovoltaica híbrida con almacenamiento de energía que participa en mercados de electricidad y servicios auxiliares. Se proponen dos agentes basados en ANN para los mercados al día siguiente y en tiempo real. En la Figura 2.6 se muestra el esquema propuesto. Ambas redes se entrenan bajo la misma función de recompensa, por lo que los pesos de ambas redes se ajustan simultáneamente. Un enfoque similar se adopta en este estudio, ya que cada agente puede especializarse en resolver un problema específico dentro del contexto de la tarea general, y los agentes pueden colaborar entre sí durante el proceso de entrenamiento. Por lo tanto, este enfoque mejora el aprendizaje global del sistema y puede llevar a soluciones más robustas.

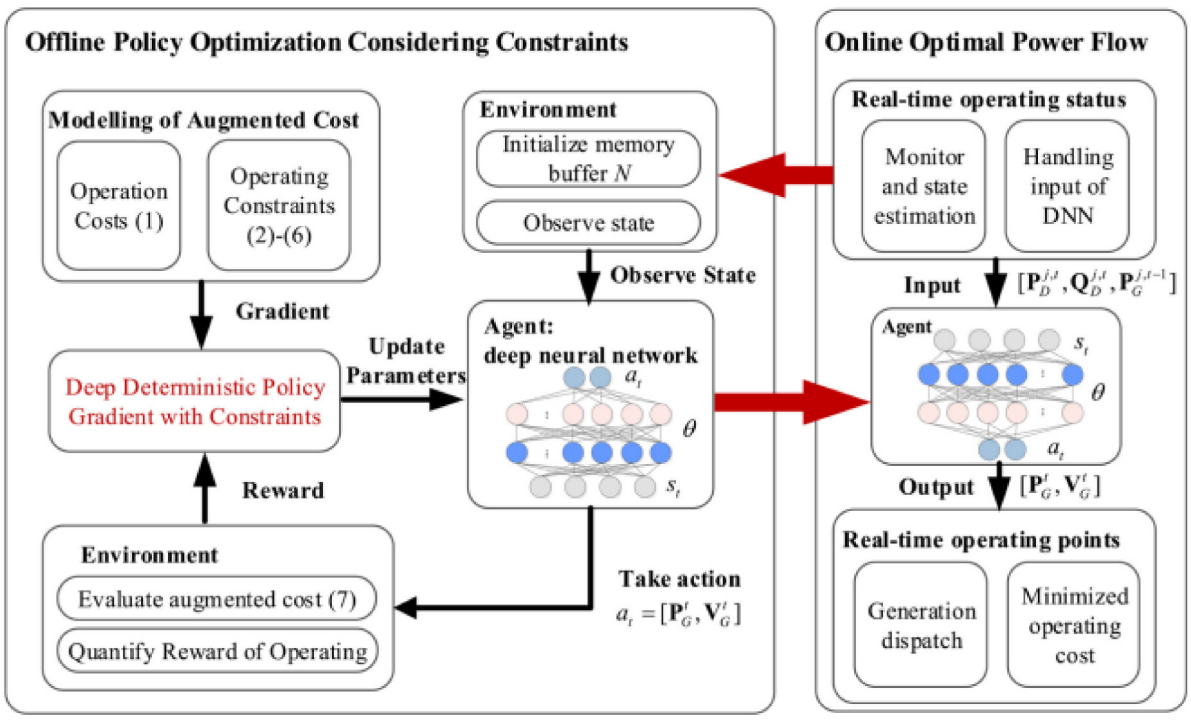


Figura 2.5. *Flowchart esquema propuesto en [24]*

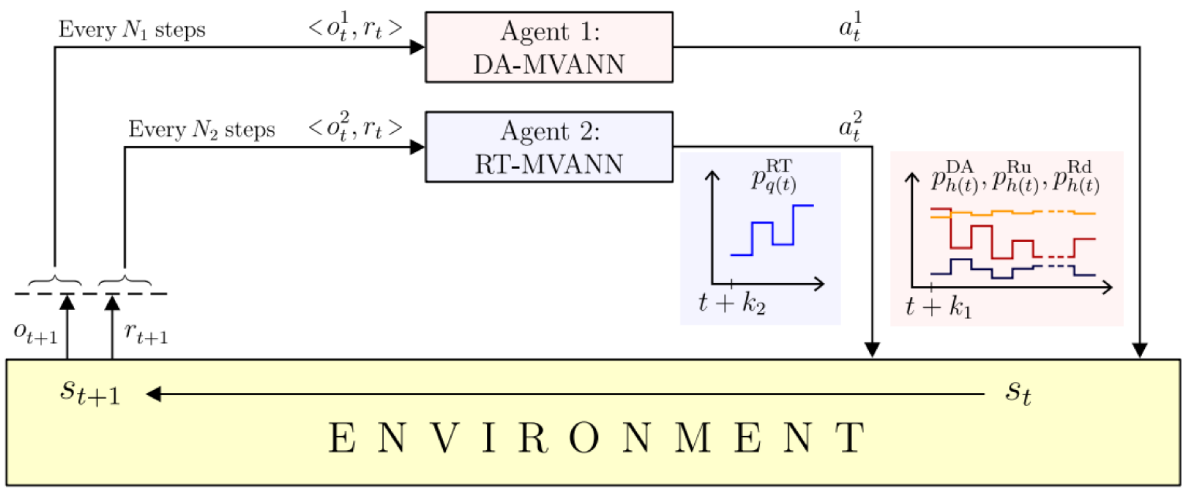


Figura 2.6. *Esquema multi-agente propuesto en [56]*

En este trabajo, se propone el uso de un modelo multi agente basado en aprendizaje reforzado profundo (*Multi-agent Deep Reinforcement Learning*, MADRL) [57–59] para reducir el espacio de soluciones en problemas de UC, prediciendo el estado de encendido/apagado de ciertas unidades, lo que acelera el rendimiento computacional de los MILP *solvers*. El modelo propuesto incluye cálculos tanto *offline* como *online*. En la parte *offline*, se obtienen variables binarias para un problema de UC utilizando datos históricos de operación del sistema eléctrico con los cuales se entrena un modelo MADRL. Una parte de la base de datos contiene la demanda en las barras y la generación de energía renovable. Para el sistema hidrotérmico, se utiliza la programación de generación hidroeléctrica de los embalses como entrada, debido a que la programación de las unidades térmicas en sistemas hidrotérmicos depende del agua utilizada durante la etapa actual. Finalmente, se incluye una representación temporal de la hora del día y la serie de tiempo del estado de encendido/apagado de las unidades a predecir. Esto permite analizar los patrones y tendencias temporales en la operación de las unidades, proporcionando información valiosa sobre su comportamiento a lo largo del tiempo. Varios agentes se utilizan para predecir el estado de encendido/apagado de las diferentes unidades. El número de agentes dependerá del

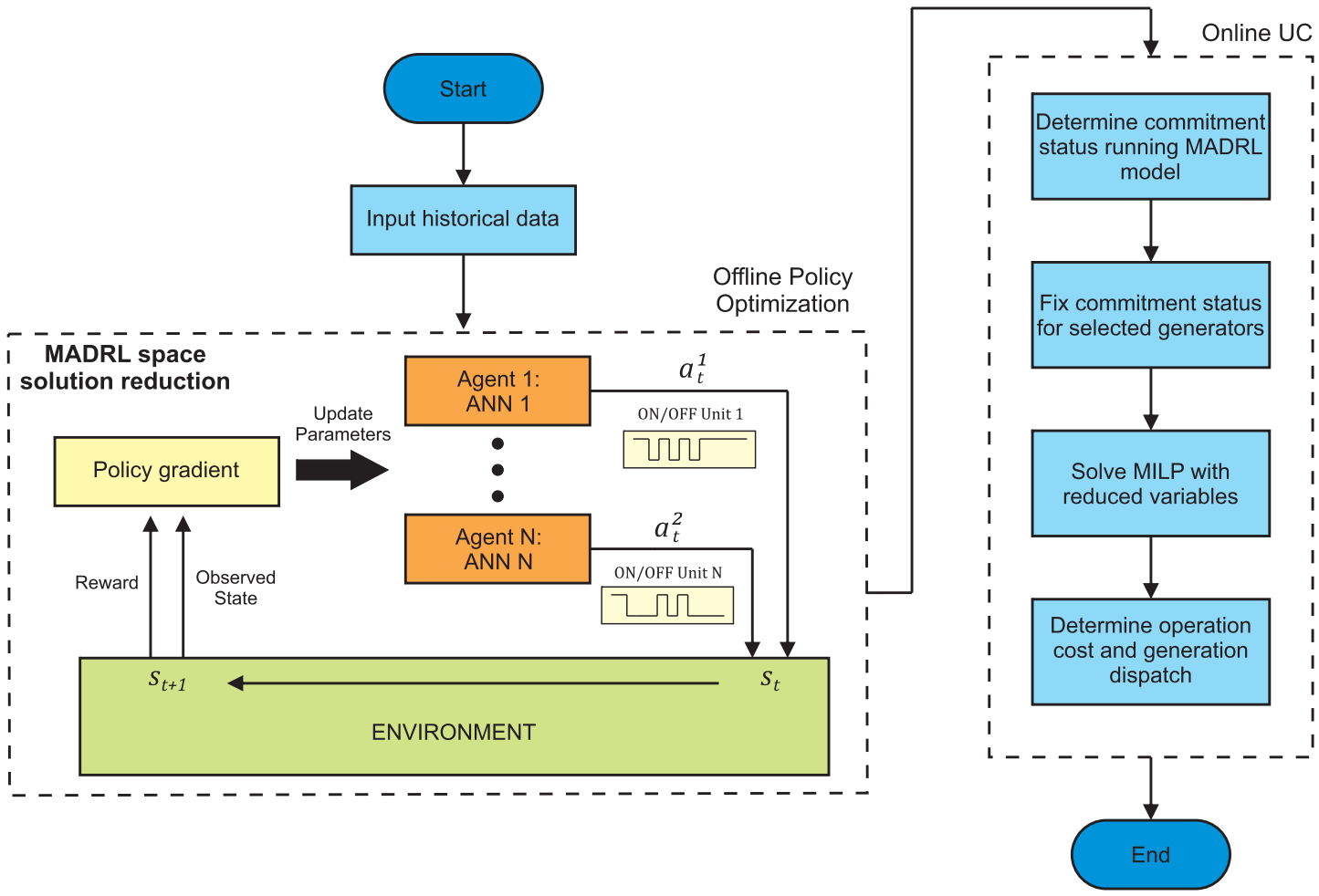


Figura 2.7. *Flowchart del esquema propuesto*

tamaño del sistema y su topología. Los agentes toman decisiones por hora basándose en las observaciones del estado del entorno (s). La fase de entrenamiento utiliza una función de recompensa compartida para fomentar la colaboración entre ellos en el aprendizaje. En el cálculo *online*, se resuelve el UC utilizando un *solver* estándar pero fijando las variables obtenidas del modelo MADRL. La Fig. 2.7 muestra un *flowchart* del esquema propuesto.

La principal contribución de este trabajo es proponer un esquema *offline/online* para reducir el espacio de soluciones de problemas de UC formulados mediante MILP en sistemas hidrotérmicos. Para implementar el componente *offline*, se introduce un algoritmo MADRL para aprovechar sus capacidades para la toma de decisiones secuenciales en entornos complejos. El algoritmo se entrena *offline* utilizando resultados históricos de programación de generación a corto plazo reales realizados por el CEN. Después de configurar los hiperparámetros con un conjunto de validación, se contrastan los resultados para un conjunto de pruebas *out of sample* con los resultados reales de UC obtenidos por el CEN.

Capítulo 3

Metodología

3.1 Modelo MADRL para la reducción del espacio de soluciones en problemas de UC

Se introduce un modelo MADRL que explora la relación entre los datos históricos y las soluciones de UC con el propósito de reducir significativamente la carga computacional del problema de UC usando una formulación MILP. Como se muestra en la Figura 2.7, el entrenamiento se realiza *offline* para predecir el estado encendido/apagado de varias unidades térmicas. La segunda etapa es un procedimiento *online*, que consiste en resolver el problema de UC con el estado de encendido/apagado fijado desde un comienzo en ciertas unidades. Este procedimiento reduce la cantidad de variables binarias en el problema MILP, disminuyendo así la carga computacional.

Una alternativa para el uso de múltiples agentes sería una única red neuronal bajo un esquema de aprendizaje multi-tarea [60], lo que permitiría ahorrar cómputo en el momento de la inferencia, ya que solo se evaluaría una única red. Lamentablemente, esto a menudo resulta en un rendimiento general inferior, ya que los objetivos de las tareas pueden competir [56]. Además, [61] demuestra empíricamente que la pérdida o ganancia de rendimiento depende de la relación entre las tareas entrenadas conjuntamente. De manera intuitiva, las características y el comportamiento de los diferentes generadores son lo suficientemente diferentes como para potencialmente comprometer el rendimiento de generalización. Por lo tanto, se utilizan agentes separados para predecir el estado de encendido/apagado de diferentes unidades o grupos de unidades con características similares.

3.2 Selección de datos de entrada

Una de las principales ventajas de las ANN es su capacidad para aprender relaciones no lineales y complejas. De esta forma, para el modelo propuesto se utilizan para mapear la *policy* de predicción de los agentes, usando como entrada datos históricos de operación. Como se ilustra en la Figura 3.1, cada agente basado en ANN recibe datos de series temporales que representan las variables de operación relevantes del sistema, las cuales reflejan el estado del sistema (s). La literatura recomienda que las mejores características que representan el estado del sistema son aquellas que cambian en el tiempo, ya que pueden capturar la dinámica de las operaciones del sistema de energía [14, 18, 51, 54, 55]. Además, se consideró la experiencia del CEN para seleccionar las variables de entrada

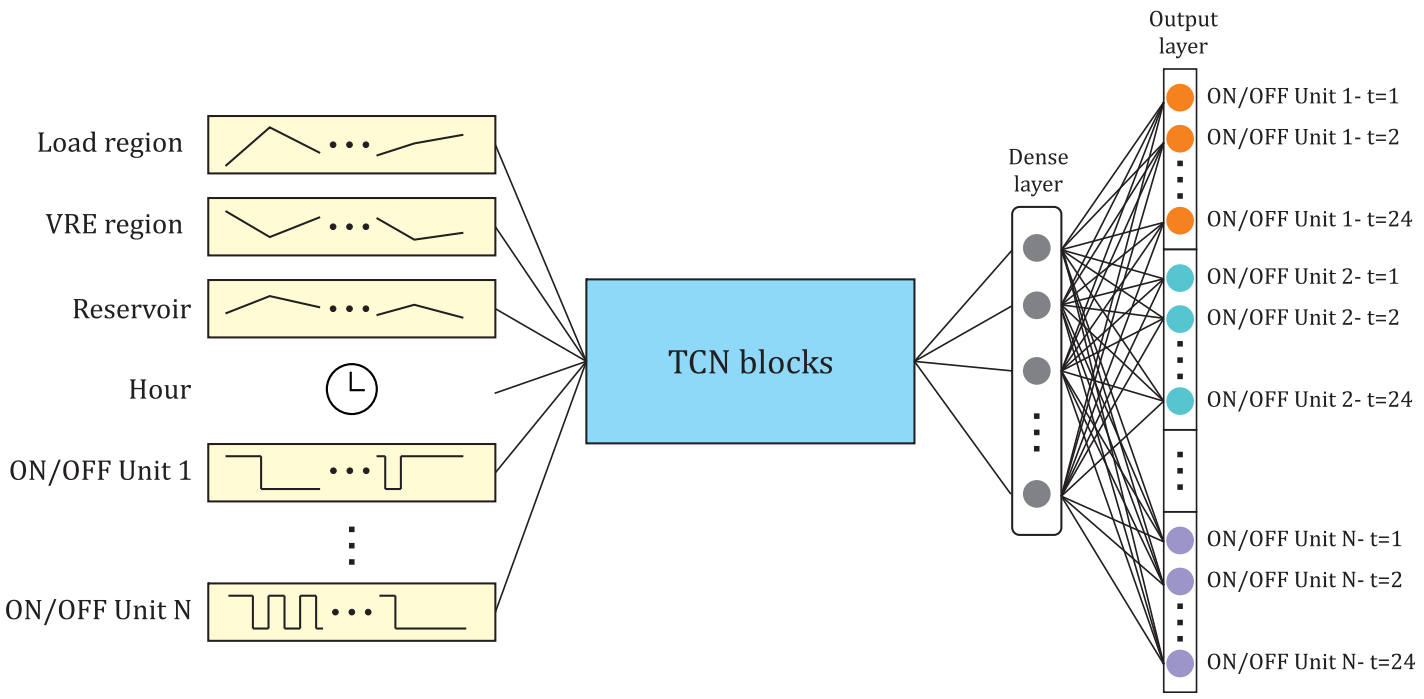


Figura 3.1. *Agente para el modelo MADRL basado en ANN*

relevantes. Para la implementación, se utilizaron perfiles de demanda en las barras, la programación de generación de energía renovable variable y el registro del estado de encendido/apagado de las diferentes unidades como entradas de cada ANN. Además, dado que el problema de UC cambia en sistemas hidrotérmicos, y considerando la experiencia del CEN, también se incluyeron como variables de entrada los programas de generación de unidades hidroeléctricas con embalses de agua significativos. Por último, se agregó una representación del tiempo en horas del día.

3.3 Arquitectura ANN

Un agente ANN se ilustra en la Figura 3.1. Se utiliza una secuencia de múltiples características para mapear el estado de encendido/apagado. En la predicción de series de tiempo en sistemas de energía, es común utilizar redes LSTM [41], pero en este trabajo, para los agentes basados en ANN se emplea una arquitectura TCN [42]. Como se explicó en la sección 2.2, esta novedosa familia de arquitecturas ha demostrado un rendimiento superior en comparación con redes LSTM en una amplia variedad de tareas, principalmente debido a su capacidad para mantener una memoria efectiva más prolongada. Además, a diferencia de las redes recurrentes como LSTM, TCN permite procesar secuencias de entrada de forma paralela, lo cual es más rápido que el procesamiento secuencial de las redes recurrentes.

La capa de salida consiste en las decisiones de encendido/apagado. La longitud de la salida dependerá de cuántas unidades se intenten predecir. La predicción del horizonte es de 24 horas, lo que significa 24 neuronas para una unidad. Si el agente toma decisiones para tres unidades, la capa de salida consistirá en 72 unidades. La función de activación para la capa de salida será $sigmoid()$, ya que el dominio de esta función es $[0, 1]$, lo que representa el estado de las unidades. 0 y 1 representan que la unidad está apagada o encendida, respectivamente.

Cabe destacar que aumentar significativamente el número de neuronas de la capa de

salida al tratar de predecir el estado de más unidades puede llevar a un mayor riesgo de *overfitting*. Esto se refiere a que el modelo se puede ajustar excesivamente a los datos de entrenamiento, perdiendo su capacidad de generalización [62]. Es fundamental encontrar un equilibrio entre la capacidad del modelo y la capacidad de generalización. Para esto se debe experimentar con diferentes combinaciones y evaluar el rendimiento del modelo con el conjunto de validación.

3.4 Fase de aprendizaje

Como es habitual en la manipulación de series de tiempo en ML, se divide el conjunto de datos en tres grupos consecutivos: entrenamiento, validación y prueba. Se ajustan los pesos de los agentes basados en ANN utilizando el conjunto de entrenamiento después de cada iteración. Se utiliza el conjunto de validación para seleccionar el mejor modelo teniendo en cuenta los diferentes hiper-parámetros de las ANN. Por último, el conjunto de prueba se utiliza para evaluar el modelo ajustado con datos *out of sample*.

Una de las claves del modelo multi-agente es lograr una fase de aprendizaje colaborativo entre los agentes basados en ANN utilizando una función de recompensa acumulativa compartida. En este caso, la salida de las diferentes neuronas se redondea a 0 o 1 debido al estado de encendido/apagado de las unidades. Esto es similar a un problema de clasificación multi-etiqueta (*Multi-label classification problem, MLCP*) [63], que consiste en predecir múltiples clases mutuamente no excluyentes. En el problema a resolver, se decidió que cada agente tendría como función de recompensa una función *binary cross entropy*. Esta función se utiliza comúnmente para comparar las probabilidades predichas con la salida de clase real en MLCP. Esta función es cero cuando el valor predicho y el valor real son iguales, y tiene un valor que depende de qué tan cerca o qué tan lejos está el valor predicho del valor real. La función de recompensa para un agente en particular se obtiene en la ecuación (3.1).

$$r_{BCE} = -\frac{1}{n}\sum_{i=1}^{n}\left[y_i \cdot \log a_i + (1-y_i)\cdot \log(1-a_i)\right]. \tag{3.1}$$

Donde y_i es el valor real, a_i es la decisión tomada por el agente y n el número de casos. La función de recompensa compartida acumulada promedio se obtiene en la ecuación (3.2).

$$R = \frac{1}{N}\sum_{j=1}^{N} r_{BCE,j}. \tag{3.2}$$

Esta es la media de las funciones de recompensa de los diferentes agentes en el modelo. Esta función dependerá de los pesos de cada agente.

El gradiente se calcula sobre las decisiones de UC tomadas por cada agente. Es importante mencionar que los agentes no comparten pesos entre sí, pero se ven influenciados por los demás debido a la función de recompensa compartida utilizada en la fase de aprendizaje. También se utilizan *mini-batches* para la fase de entrenamiento, lo que permite un proceso más eficiente en términos de computación. Se actualizan los pesos de cada ANN utilizando *back-propagation* en cada iteración de entrenamiento. El gradiente de cada ANN se puede descomponer de la siguiente manera:

$$\nabla_\theta R = \nabla_{\boldsymbol{a}} R \cdot \nabla_\theta \boldsymbol{a}, \tag{3.3}$$

donde $\nabla_{\boldsymbol{a}} R$ es el gradiente de la función de recompensa acumulada con respecto a las acciones $\boldsymbol{a}$, es decir, el estado de las unidades en el horizonte $\boldsymbol{a} = [\boldsymbol{\mu}_{n=1,\boldsymbol{t}}, \ldots, \boldsymbol{\mu}_{n=N,\boldsymbol{t}},]$ y $\nabla_\theta \boldsymbol{a}$ es el gradiente de $\boldsymbol{a}$ con respecto al parámetro de red neuronal θ. El algoritmo 3.1 muestra la fase de aprendizaje.

Algorithm 3.1 Learning phase

```
Randomly initialize critic ANN's weights
Initialize replay buffer R
for each training iteration do
    Randomly sample a mini-batch of replay buffer R
    Receive initial observation state s_0
    for t = 1 to T do
        Determine action a_t
        Collect UC solutions from ANNs
    end for
    Observe state and loss function
    Calculate the gradient ∇_θ R
    Update ANN's weights using policy gradient
    if stop criterion is met then
        break
    end if
end for
```

Capítulo 4

Caso de estudio

El modelo propuesto se prueba en el SEN. Hasta diciembre de 2022, el sistema cuenta con una capacidad instalada de 33.218 MW. Las centrales renovables representan el 62.0% de la capacidad instalada (22,3% hidroeléctrica, 24,1% solar, 13,0% eólica, 2,3% biomasa y 0,3% geotérmica), mientras que las centrales térmicas representan el 38.0% (13,0% carbón, 15,1% gas natural y 9,8% petróleo) [64]. Los experimentos de entrenamiento para las ANN se realizaron en un equipo con un procesador Intel(R) Xeon(R) @ 2.20GHz, una GPU NVIDIA A100-SXM4-40GB y 85 GB de RAM disponible. Por otro lado, las simulaciones MILP se llevaron a cabo en un equipo con un procesador Intel(R) Xeon(R) CPU E5-2650 y 128 GB de RAM. Las simulaciones de UC fueron simuladas usando PLEXOS 9.100 [65], un software de simulación de mercado eléctrico basado en formulación MILP que el CEN emplea para obtener la programación diaria con un horizonte de 168 horas. El problema matemático formulado por PLEXOS se resolvió mediante Gurobi versión 9.5.2 [66].

4.1 Datos

La demanda en las barras, la programación de energías renovables variables, la programación de generación para los embalses y los datos históricos de operación de las unidades térmicas son obtenidos desde el sitio web del CEN [67,68]. Estos conjuntos de datos abarcan las operaciones reales y planificadas del sistema desde el 1 de enero de 2019 hasta el 31 de diciembre de 2022, con una resolución horaria. El conjunto de datos completo se dividió en conjuntos de entrenamiento, validación y prueba. Los datos de 2019 a 2021 se utilizaron para el entrenamiento, los primeros dos meses de 2022 se utilizaron para la validación y el resto de los datos de 2022 se utilizaron con fines de prueba.

Se seleccionaron nueve unidades térmicas para predecir y fijar su estado al resolver el problema de UC con espacio de soluciones reducido. La selección de las unidades se basa en dos aspectos: su importancia relativa para el costo total del problema de UC y el número de variables binarias que cada unidad representa. Diferentes subconjuntos de unidades se probaron utilizando el conjunto de validación, donde se seleccionó el subconjunto con mejores resultados para ser probado en el conjunto de prueba. El subconjunto de nueve unidades térmicas seleccionadas para fijar en el problema de UC con un espacio de solución reducido representan 1,8 GW de potencia y 129 variables binarias.

Agente	Número de unidades	Potencia (MW)
1	2	57
2	2	762
3	2	120
4	1	255
5	1	370
6	1	248

Tabla 4.1

Número de unidades y potencia máxima de estas para cada agente

Hiper-parámetro	Espacio de búsqueda	Valor final
Largo entrada	72, 96, 120, 144, 168	168
Neuronas capa densa	10 - 64	48
Tamaño *kernel*	3 - 4	3
Filtros	16 - 64	16
Bloques	1 - 2	1
Factores de dilatación	[1, 2, 4, 8, 16], [1, 3, 9, 27, 81], [1, 3, 6, 12, 24], [1, 5, 7, 14, 28]	[1, 2, 4, 8, 16]

Tabla 4.2

Valores de los hiper-parámetros para cada ANN

4.2 Ajustes ANNs

Seis agentes basados en ANN se usaron para representar los estados de encendido/apagado de las nueve centrales térmicas. Se agruparon algunas unidades debido a que se encuentran en áreas cercanas y tienden a comportarse de manera similar. La Tabla 4.1 muestra el número de unidades asociadas a cada agente y su capacidad.

Para las iteraciones de entrenamiento se implementó un criterio *early stopping* con una paciencia de 50 iteraciones. Se utilizaron activaciones *ReLU* para las neuronas y el optimizador Adam para el entrenamiento. Este último cuenta con una tasa de aprendizaje adaptativa para mejorar la convergencia de las redes [69].

Se llevó a cabo un estudio experimental exhaustivo utilizando el conjunto de datos de validación para seleccionar los hiper-parámetros de las ANN, involucrando combinaciones de diferentes valores. Los hiper-parámetros usados fueron el número de filtros y bloques residuales apilados, el tamaños del *kernel* y los factores de dilatación. La longitud de la entrada también es un hiper-parámetro ajustable. La Tabla 4.2 muestra el espacio de búsqueda para cada parámetro y sus valores finales.

La Figura 4.1 muestra la evolución de la recompensa acumulativa para las ANN seleccionadas.

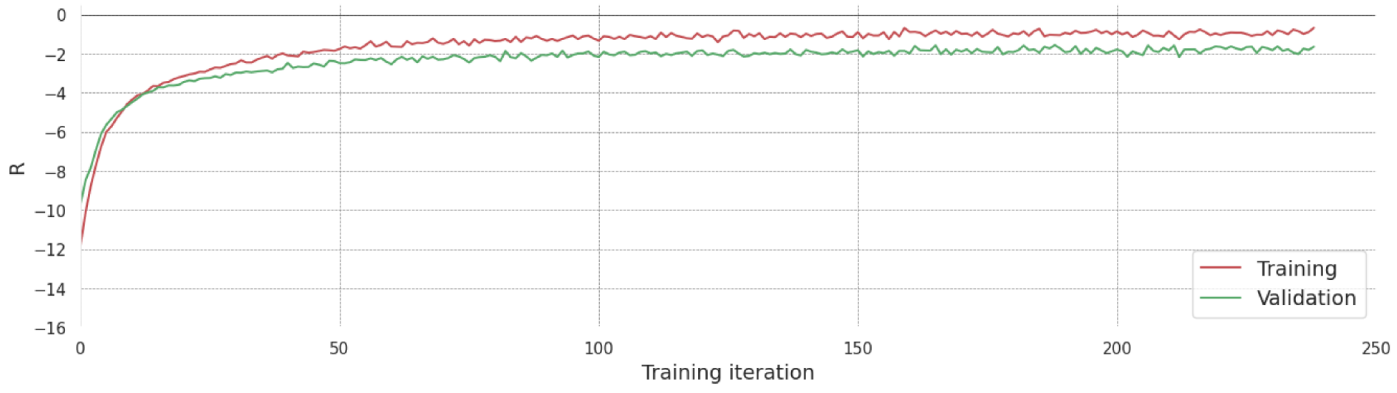


Figura 4.1. *Evolución para la función de recompensa en los conjuntos de entrenamiento y validación para las ANN seleccionados*

4.3 Resultados

4.3.1 Rendimiento ANNs

Dado que los casos de UC en PLEXOS y sus resultados están disponibles públicamente en el sitio web del CEN [68], se puede comparar los estados de encendido/apagado de las unidades seleccionadas predichos por el modelo MADRL con los resultados reales de PLEXOS. Los resultados se muestran en la Tabla 4.3. El rendimiento de las predicciones se evaluó utilizando métricas como exactitud, precisión, *recall* y especificidad [70]. Estas métricas brindan perspectivas sobre la eficacia del modelo propuesto para predecir correctamente el estado de las unidades térmicas. Los resultados obtenidos destacan la capacidad del modelo para predecir con precisión períodos en los que las unidades térmicas están apagadas, como lo demuestran los altos valores de exactitud y especificidad. Sin embargo, es crucial señalar que tanto el *recall* como la precisión presentan valores cercanos a 0.76. Esto se debe a un desequilibrio observado entre los estados de encendido y apagado para ciertas centrales, lo que plantea un desafío para que la ANN capture efectivamente la clase minoritaria. Sin embargo, si se adopta una estrategia de fijar ceros en el problema de UC, se mitiga el impacto de este problema y permite que la red destaque en su rendimiento.

Es importante observar que los resultados en la Tabla 4.3 solo comparan qué tan similares son los estados de encendido/apagado de la solución original de PLEXOS y las predicciones realizadas por el modelo MADRL. La calidad de las soluciones en términos del valor de la función objetivo y el rendimiento computacional se discuten a continuación.

Recall	**Especificidad**	**Precisión**	**Exactitud**
0,7556	0,9624	0,7614	0,9704

Tabla 4.3

Resultados de predicción para los estados encendido/apagado de las unidades

4.3.2 Rendimiento computacional del modelo propuesto

Esta sección compara la resolución normal del problema de UC con el modelo propuesto de reducción del espacio de soluciones basado en MADRL. Para que la comparación de rendimiento computacional sea justa, se ejecutó nuevamente las simulaciones originales de PLEXOS (es decir, sin reducir el espacio de soluciones) en el mismo equipo donde se ejecutaron las optimizaciones de UC con un espacio de soluciones reducido. Dado que los tiempos de solución pueden llevar varias horas, solo se realizaron pruebas para un

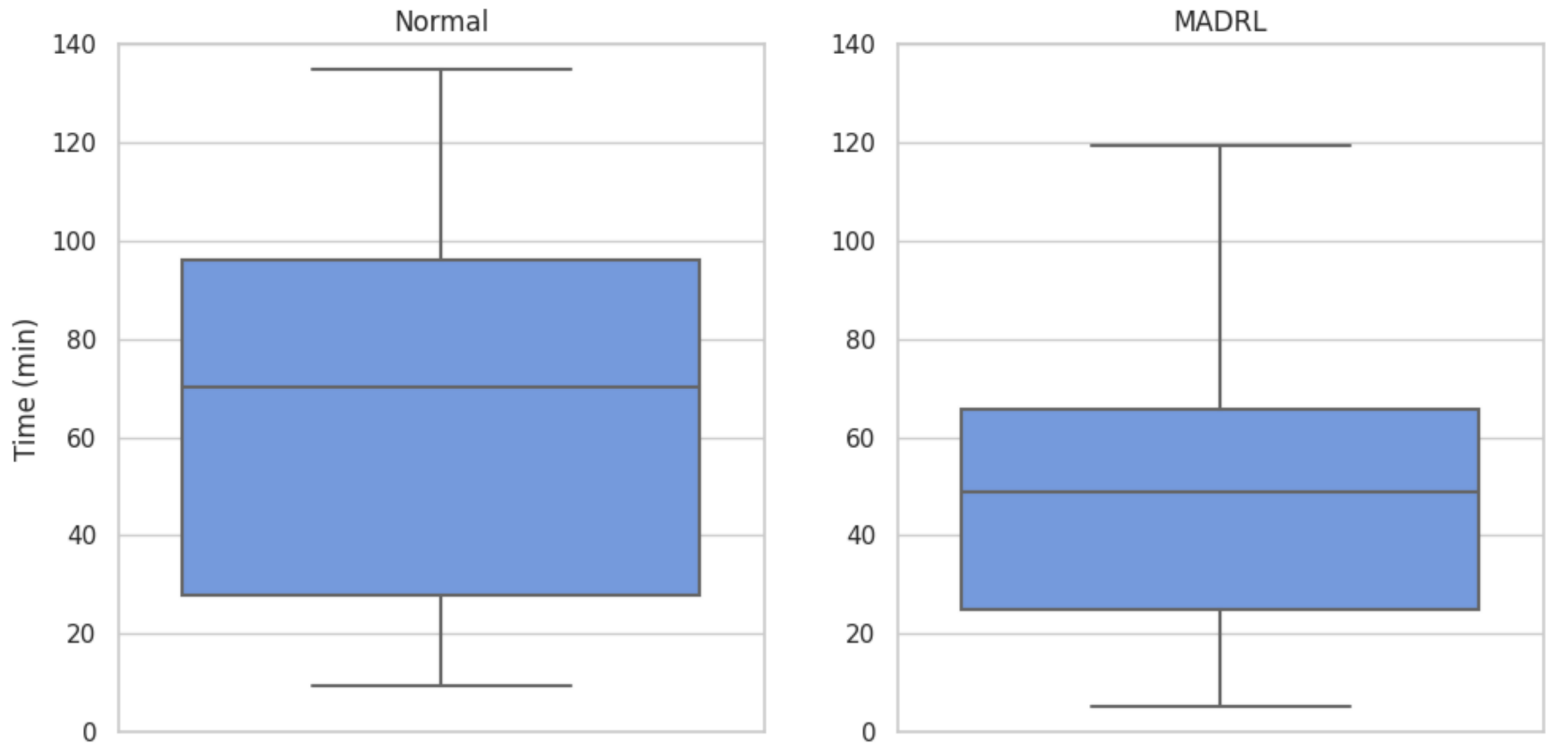


Figura 4.2. *Comparación tiempos de resolución para los casos normales y usando el modelo MADRL*

número limitado de días. Además, para representar de manera integral diversas condiciones hidrológicas dentro de un año único, se realizaron pruebas para 60 días seleccionados al azar entre marzo y diciembre de 2022. Este enfoque tiene como objetivo capturar una amplia gama de escenarios y considerar los diversos niveles de producción hidroeléctrica. Cabe destacar que los meses de octubre, noviembre y diciembre se caracterizan por un aumento en la producción hidroeléctrica debido al deshielo. Notablemente, el método propuesto mostró un rendimiento consistente tanto en meses lluviosos como en períodos de deshielo, sin observarse diferencias significativas. Estos resultados sugieren que el modelo es efectivo y capaz de proporcionar soluciones confiables independientemente de las condiciones hidrológicas predominantes.

En el caso sin espacio de soluciones reducido, el tiempo promedio de solución es de 65,9 minutos. Mientras tanto, el tiempo promedio de solución para el modelo propuesto es de solo 48,7 minutos. Esto representa una reducción promedio de tiempo del 26,1%. Por supuesto, el rendimiento puede variar según la caso de UC que se esté resolviendo. La Figura 4.2 muestra *boxplots* para el tiempo de resolución del UC. El rango intercuartil para el caso normal se sitúa entre 30 y 100 minutos, mientras que para el modelo propuesto, oscila entre 25 y 65 minutos, lo que demuestra la eficacia de método en términos de tiempo de resolución. Además, se obtuvo un *speed-up* promedio de 1.6, lo que confirma aún más la eficiencia computacional del enfoque propuesto.

La Figura 4.3 muestra la distribución del *speed-up* para el modelo propuesto. Se puede observar que la mayoría de los casos de prueba tienen mejoras de velocidad que oscilan entre 1 y 2. Incluso en ciertas instancias, las mejoras de velocidad pueden alcanzar valores entre 6 y 8. Se realizó un *t-test* para evaluar la reducción de tiempo lograda por el método. Un *p-value* de 6.46×10^{-3} indica una reducción de tiempo estadísticamente significativa. También un *F-test* reveló que hay una reducción estadísticamente significativa en la varianza del tiempo de solución (valor p de 4.92×10^{-2}).

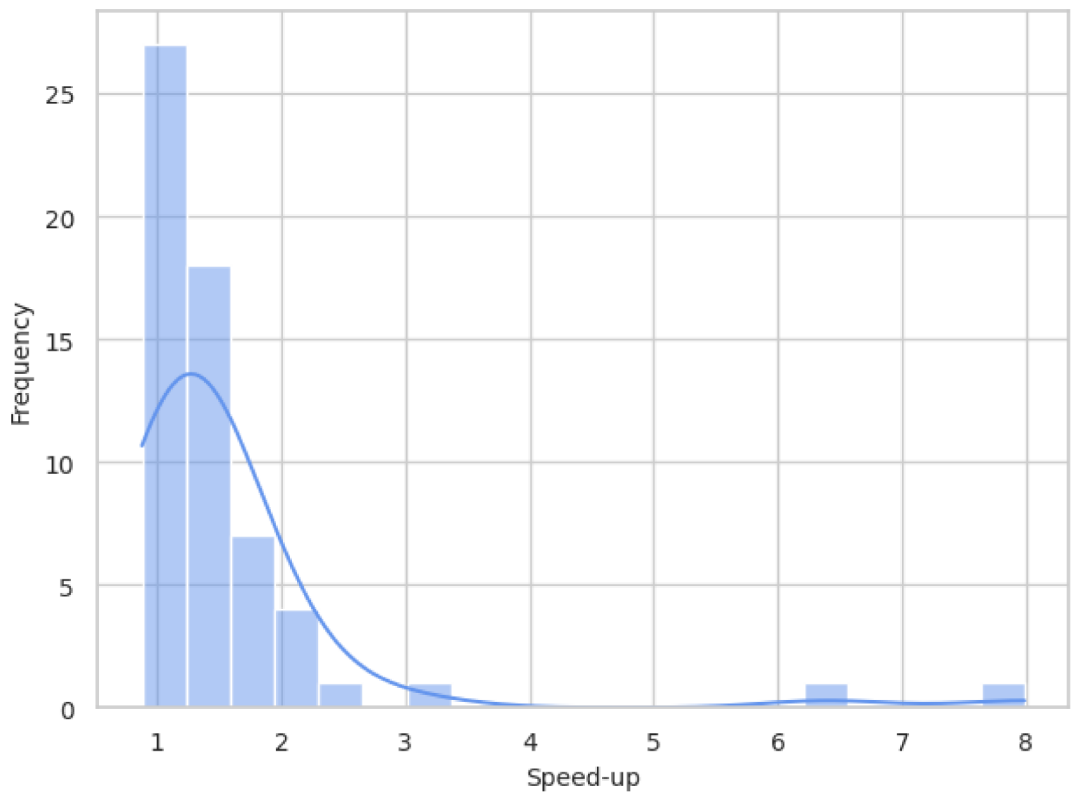


Figura 4.3. *Distribución del speed-up*

Otro aspecto importante es el costo total. El objetivo principal era lograr un método que acelere los tiempos de resolución sin comprometer la calidad de la solución en términos de costo total. Esto significa que la solución proporcionada por el método propuesto debe ubicarse entre los límites inferiores y superiores establecidos por el *gap* de la solución. La Figura 4.5 muestra el costo normalizado para los casos de prueba. Se incluyen límites superiores e inferiores, calculados como un promedio entre los diferentes límites. En la mayoría de los casos, la solución se encuentra dentro de los márgenes establecidos y, en algunos casos, se lograron soluciones más eficientes en términos de costo.

La Figura 4.4 compara la mejora de velocidad, el ahorro de tiempo y el costo operacional. Se observan ahorros significativos de tiempo al utilizar el modelo propuesto, logrando en algunos casos más de 40 minutos de reducción en el tiempo de ejecución. En las pocas instancias donde la mejora de velocidad es menor que uno, la diferencia de tiempo no supera los tres minutos. Además, existe un costo operativo más bajo para estos casos en comparación con el caso normal. La combinación de ahorro de tiempo y variación de costo aceptable demuestra la efectividad y practicidad del método propuesto para resolver el problema de UC.

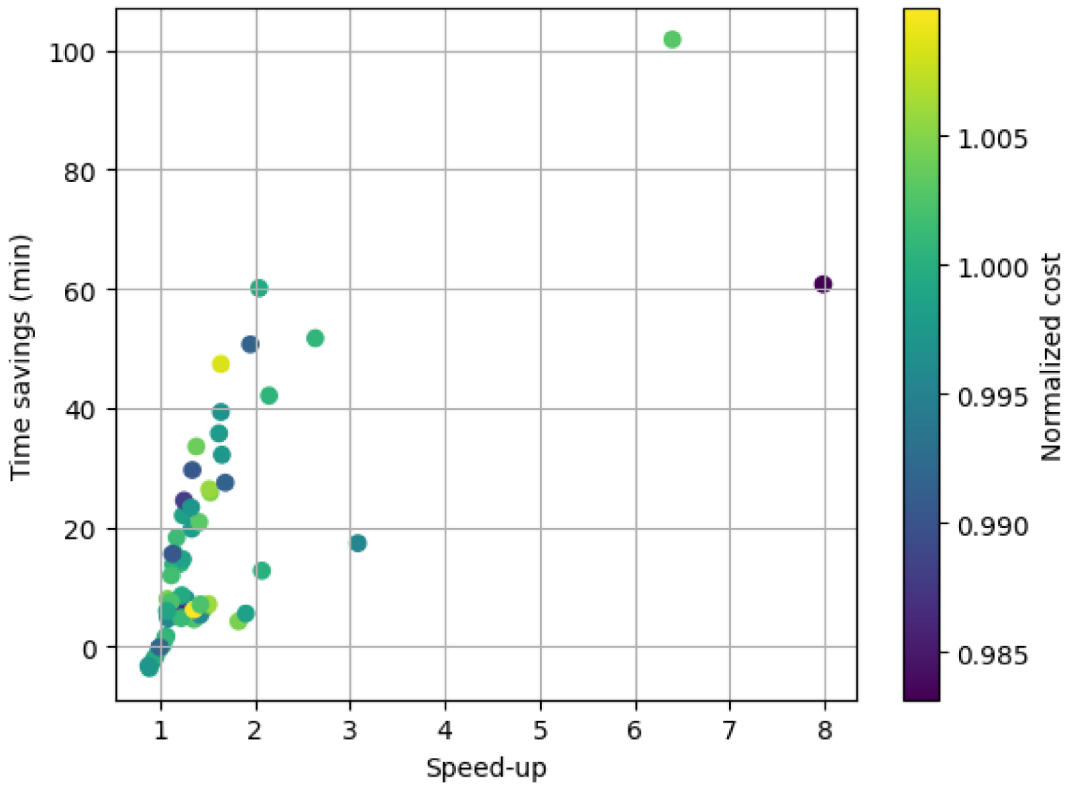


Figura 4.4. *Comparación entre el tiempo ahorrado, speed-up y costo operacional*

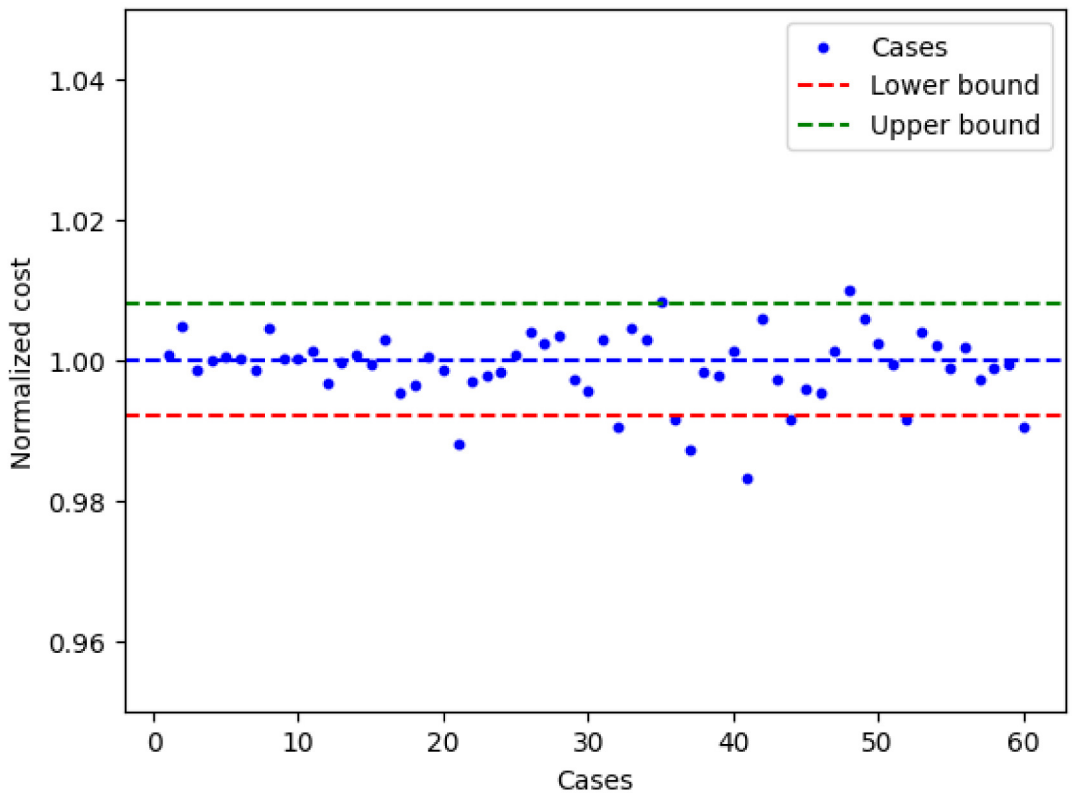


Figura 4.5. *Costo normalizado para las casos probados*

Capítulo 5

Conclusiones

Este trabajo presenta un enfoque innovador que utiliza ANN para mejorar el proceso de resolución del problema de UC en sistemas hidrotérmicos. Utilizando ANN con una arquitectura TCN se predice con precisión los estados de encendido/apagado de las unidades térmicas, demostrando así ser una estrategia efectiva para reducir el espacio de solución del problema de UC.

Todas las pruebas se realizaron en casos reales de procesos de programación de la operación del SEN, con un horizonte de tiempo de 168 horas. Los resultados demuestran que el método propuesto acelera significativamente los tiempos de resolución del problema de UC. Además, la mayoría de los casos evaluados indican que el costo asociado con el enfoque propuesto se mantiene dentro de los márgenes de error establecidos. Se resalta de igual manera que en algunos de los casos probados se obtuvieron soluciones incluso más eficientes desde el punto de vista del costo de operación. Todos estos resultados destacan la viabilidad y el potencial en la aplicación práctica del método propuesto.

Futuras investigaciones incluyen explorar nuevas estrategias para fijar variables binarias, lo cual podría llevar a una mayor eficiencia en la resolución del problema de UC. Asimismo, sería de gran interés llevar a cabo un estudio exhaustivo de distintos tipos de arquitecturas de ANN, acompañado de un análisis de sensibilidad de hiper-parámetros clave, como por ejemplo, el número de filtros en una TCN. Este análisis proporcionaría información valiosa sobre cómo la elección de la arquitectura y la complejidad del modelo afectan el rendimiento del método. Por ultimo, la incorporación de restricciones operativas adicionales durante el entrenamiento de las ANN podría ser beneficioso, mejorando la precisión de las predicciones y su aplicabilidad en contextos del mundo real.

En resumen, este estudio aborda de manera exitosa la aceleración y mejora de la resolución del problema de UC en sistemas hidrotérmicos. La integración de ANN en el proceso de toma de decisiones en el UC demuestra su efectividad para reducir significativamente los tiempos de cálculo sin comprometer la calidad de las soluciones obtenidas. La aplicación de este enfoque en el SEN no solo proporciona resultados prometedores, sino que también sienta las bases para futuras mejoras y refinamientos en la resolución de problemas similares en otras regiones y sistemas eléctricos.

Bibliografía

[1] "Coordinador eléctrico nacional presentó detalles de su hoja de ruta para una transición energética acelerada," *Coordinador Eléctrico Nacional*. [Online]. Available: https://www.coordinador.cl/novedades/coordinador-electrico-nacional-presento-detalles-de-su-hoja-de-ruta-para-una-transicion-energetica-acelerada/

[2] Ministerio de Energía, "Programa de descarbonización 2050." [Online]. Available: https://energia.gob.cl/noticias/nacional/ministro-de-energia-anuncia-que-para-el-2025-habremos-retirado-el-50-de-las-centrales-carbon

[3] N. P. Padhy, "Unit commitment-a bibliographical survey," *IEEE Transactions on power systems*, vol. 19, no. 2, pp. 1196–1205, 2004.

[4] S. Sen and D. Kothari, "Optimal thermal generating unit commitment: a review," *International Journal of Electrical Power & Energy Systems*, vol. 20, no. 7, pp. 443–451, 1998.

[5] R. Burns, "Optimization of priority lists for a unit commitment program," in *Proc. IEEE Power Eng. Soc. Summer Meeting, 1975*, 1975.

[6] P. Lowery, "Generating unit commitment by dynamic programming," *IEEE Transactions on Power Apparatus and Systems*, no. 5, pp. 422–426, 1966.

[7] A. Merlin and P. Sandrin, "A new method for unit commitment at electricite de france," *IEEE transactions on power apparatus and systems*, no. 5, pp. 1218–1225, 1983.

[8] Z. Ma, H. Zhong, Q. Xia, C. Kang, Q. Wang, and X. Cao, "A unit commitment algorithm with relaxation-based neighborhood search and improved relaxation inducement," *IEEE Transactions on Power Systems*, vol. 35, no. 5, pp. 3800–3809, 2020.

[9] X. Sun, P. B. Luh, M. A. Bragin, Y. Chen, J. Wan, and F. Wang, "A novel decomposition and coordination approach for large day-ahead unit commitment with combined cycle units," *IEEE Transactions on Power Systems*, vol. 33, no. 5, pp. 5297–5308, 2018.

[10] L. L. Garver, "Power generation scheduling by integer programming-development of theory," *Transactions of the American Institute of Electrical Engineers. Part III: Power Apparatus and Systems*, vol. 81, no. 3, pp. 730–734, 1962.

[11] B. Knueven, J. Ostrowski, and J.-P. Watson, "On mixed-integer programming formulations for the unit commitment problem," *INFORMS Journal on Computing*, vol. 32, no. 4, pp. 857–876, 2020.

[12] Z. Jin, K. Pan, L. Fan, and T. Ding, "Data-driven look-ahead unit commitment considering forbidden zones and dynamic ramping rates," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3267–3276, 2018.

[13] X. Lei, Z. Yang, J. Yu, J. Zhao, Q. Gao, and H. Yu, "Data-driven optimal power flow: A physics-informed machine learning approach," *IEEE Transactions on Power Systems*, vol. 36, no. 1, pp. 346–354, 2020.

[14] Y. Zhou, Q. Zhai, L. Wu, and M. Shahidehpour, "A data-driven variable reduction approach for transmission-constrained unit commitment of large-scale systems," *Journal of Modern Power Systems and Clean Energy*, 2022.

[15] R. Nayak and J. Sharma, "A hybrid neural network and simulated annealing approach to the unit commitment problem," *Computers & Electrical Engineering*, vol. 26, no. 6, pp. 461–477, 2000.

[16] M. Li, W. Wei, Y. Chen, M.-F. Ge, and J. P. Catalao, "Learning the optimal strategy of power system operation with varying renewable generations," *IEEE Transactions on Sustainable Energy*, vol. 12, no. 4, pp. 2293–2305, 2021.

[17] A. S. Xavier, F. Qiu, and S. Ahmed, "Learning to solve large-scale security-constrained unit commitment problems," *INFORMS Journal on Computing*, vol. 33, no. 2, pp. 739–756, 2021.

[18] Z. Lin, Y. Chen, J. Yang, C. Ma, H. Liu, L. Liu, L. Li, and Y. Li, "Accelerating transmission-constrained unit commitment via a data-driven learning framework," *Frontiers in Energy Research*, vol. 10, p. 1012781, 2023.

[19] Y. Wu, H. Tan, L. Qin, B. Ran, and Z. Jiang, "A hybrid deep learning based traffic flow prediction method and its understanding," *Transportation Research Part C: Emerging Technologies*, vol. 90, pp. 166–180, 2018.

[20] E. Jasmin, T. I. Ahamed, and T. Remani, "A function approximation approach to reinforcement learning for solving unit commitment problem with photo voltaic sources," in *2016 IEEE International Conference on Power Electronics, Drives and Energy Systems (PEDES)*. IEEE, 2016, pp. 1–6.

[21] W. Liu, P. Zhuang, H. Liang, J. Peng, and Z. Huang, "Distributed economic dispatch in microgrids based on cooperative reinforcement learning," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 6, pp. 2192–2203, 2018.

[22] Y. Yang, Z. Yang, J. Yu, B. Zhang, Y. Zhang, and H. Yu, "Fast calculation of probabilistic power flow: A model-based deep learning approach," *IEEE Transactions on Smart Grid*, vol. 11, no. 3, pp. 2235–2244, 2019.

[23] C. Ning and F. You, "Optimization under uncertainty in the era of big data and deep learning: When machine learning meets mathematical programming," *Computers & Chemical Engineering*, vol. 125, pp. 434–448, 2019.

[24] Z. Yan and Y. Xu, "Real-time optimal power flow: A lagrangian based deep reinforcement learning approach," *IEEE Transactions on Power Systems*, vol. 35, no. 4, pp. 3270–3273, 2020.

[25] Coordinador Eléctrico Nacional, "Programas de operación." [Online]. Available: https://www.coordinador.cl/operacion/documentos/programas-de-operacion-2021/

[26] A. J. Ardakani and F. Bouffard, "Acceleration of umbrella constraint discovery in

generation scheduling problems," *IEEE Transactions on Power Systems*, vol. 30, no. 4, pp. 2100–2109, 2014.

[27] A. S. Xavier, F. Qiu, F. Wang, and P. R. Thimmapuram, "Transmission constraint filtering in large-scale security-constrained unit commitment," *IEEE Transactions on Power Systems*, vol. 34, no. 3, pp. 2457–2460, 2019.

[28] Y. Yang, Z. Yang, J. Yu, K. Xie, and L. Jin, "Fast economic dispatch in smart grids using deep learning: An active constraint screening approach," *IEEE Internet of Things Journal*, vol. 7, no. 11, pp. 11 030–11 040, 2020.

[29] X. Li, Q. Zhai, J. Zhou, and X. Guan, "A variable reduction method for large-scale unit commitment," *IEEE Transactions on Power Systems*, vol. 35, no. 1, pp. 261–272, 2019.

[30] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[31] V. Nair, S. Bartunov, F. Gimeno, I. von Glehn, P. Lichocki, I. Lobov, B. O'Donoghue, N. Sonnerat, C. Tjandraatmadja, P. Wang *et al.*, "Solving mixed integer programs using neural networks," *arXiv preprint arXiv:2012.13349*, 2020.

[32] G. Ruan, H. Zhong, G. Zhang, Y. He, X. Wang, and T. Pu, "Review of learning-assisted power system optimization," *CSEE Journal of Power and Energy Systems*, vol. 7, no. 2, pp. 221–231, 2020.

[33] F. Hasan, A. Kargarian, and A. Mohammadi, "A survey on applications of machine learning for optimal power flow," in *2020 IEEE Texas Power and Energy Conference (TPEC)*. IEEE, 2020, pp. 1–6.

[34] G. Bebis and M. Georgiopoulos, "Feed-forward neural networks," *IEEE Potentials*, vol. 13, no. 4, pp. 27–31, 1994.

[35] Z. Zhang and J. Zhao, "A deep belief network based fault diagnosis model for complex chemical processes," *Computers & chemical engineering*, vol. 107, pp. 395–407, 2017.

[36] C. Shang, F. Yang, D. Huang, and W. Lyu, "Data-driven soft sensor development based on deep learning technique," *Journal of Process Control*, vol. 24, no. 3, pp. 223–233, 2014.

[37] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[38] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," *arXiv preprint arXiv:1511.08458*, 2015.

[39] P. Lara-Benítez, M. Carranza-García, J. M. Luna-Romera, and J. C. Riquelme, "Temporal convolutional networks applied to energy-related time series forecasting," *applied sciences*, vol. 10, no. 7, p. 2322, 2020.

[40] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*. Ieee, 2013, pp. 6645–6649.

[41] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[42] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional

and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[43] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[44] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *International conference on machine learning*. Pmlr, 2013, pp. 1310–1318.

[45] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[46] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[47] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[48] P. Dayan and C. Watkins, "Q-learning," *Machine learning*, vol. 8, no. 3, pp. 279–292, 1992.

[49] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[50] Y. Yang and L. Wu, "Machine learning approaches to the unit commitment problem: Current trends, emerging challenges, and new strategies," *The Electricity Journal*, vol. 34, no. 1, p. 106889, 2021.

[51] C. Wang and S. Shahidehpour, "Effects of ramp-rate limits on unit commitment and economic dispatch," *IEEE Transactions on Power Systems*, vol. 8, no. 3, pp. 1341–1350, 1993.

[52] M. Walsh and M. O'malley, "Augmented hopfield network for unit commitment and economic dispatch," *IEEE Transactions on Power Systems*, vol. 12, no. 4, pp. 1765–1774, 1997.

[53] A. M. Alvarez, Q. Louveaux, and L. Wehenkel, "A machine learning-based approximation of strong branching," *INFORMS Journal on Computing*, vol. 29, no. 1, pp. 185–195, 2017.

[54] A. V. Ramesh and X. Li, "Feasibility layer aided machine learning approach for day-ahead operations," *IEEE Transactions on Power Systems*, 2023.

[55] ——, "Spatio-temporal deep learning-assisted reduced security-constrained unit commitment," *arXiv preprint arXiv:2306.01570*, 2023.

[56] T. Ochoa, E. Gil, A. Angulo, and C. Valle, "Multi-agent deep reinforcement learning for efficient multi-timescale bidding of a hybrid power plant in day-ahead and real-time markets," *Applied Energy*, vol. 317, p. 119067, 2022.

[57] P. Hernandez-Leal, B. Kartal, and M. E. Taylor, "A survey and critique of multiagent deep reinforcement learning," *Autonomous Agents and Multi-Agent Systems*, vol. 33, no. 6, pp. 750–797, 2019.

[58] S. Gronauer and K. Diepold, "Multi-agent deep reinforcement learning: a survey," *Artificial Intelligence Review*, pp. 1–49, 2022.

[59] W. Du and S. Ding, "A survey on multi-agent deep reinforcement learning: from the perspective of challenges and applications," *Artificial Intelligence Review*, vol. 54, pp. 3215–3238, 2021.

[60] Y. Zhang and Q. Yang, "A survey on multi-task learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 12, pp. 5586–5609, 2021.

[61] T. Standley, A. Zamir, D. Chen, L. Guibas, J. Malik, and S. Savarese, "Which tasks should be learned together in multi-task learning?" in *International Conference on Machine Learning*. PMLR, 2020, pp. 9120–9132.

[62] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning (still) requires rethinking generalization," *Communications of the ACM*, vol. 64, no. 3, pp. 107–115, 2021.

[63] A. N. Tarekegn, M. Giacobini, and K. Michalak, "A review of methods for imbalanced multi-label classification," *Pattern Recognition*, vol. 118, p. 107965, 2021.

[64] Generadoras Chile, "Capacidad de generación en chile." [Online]. Available: http://generadoras.cl/generacion-electrica-en-chile

[65] Energy Examplar, "Plexos for power systems-power market simulation and analysis software." [Online]. Available: https://www.energyexemplar.com/plexos

[66] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2023. [Online]. Available: https://www.gurobi.com

[67] Coordinador Eléctrico Nacional, "Operación real." [Online]. Available: https://www.coordinador.cl/operacion/graficos/operacion-real/

[68] ——, "Generación programada." [Online]. Available: https://www.coordinador.cl/operacion/graficos/operacion-programada/generacion-programada/

[69] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[70] M. Hossin and M. N. Sulaiman, "A review on evaluation metrics for data classification evaluations," *International journal of data mining & knowledge management process*, vol. 5, no. 2, p. 1, 2015.