Universidad Nacional Autónoma de México

Facultad de Química

Uso de redes neuronales artificiales para la predicción de constantes termodinámicas en su estado critico y factor acéntrico basado en descriptores de reactividad química local

T E S I S

para obtener el título de:

Licenciado en Química

presenta:

Dante Alan Jiménez Mancilla

Tutor:

Dr. José Marco Antonio Franco Peréz

CIUDAD UNIVERSITARIA, CD. MX. 2023

JURADO ASIGNADO:

PRESIDENTE: **Profesor: Chavez Garcia Maria de Lourdes**

VOCAL: **Profesor: Juan Raúl Álvarez Idaboy**

SECRETARIO: **Profesor: José Marco Antonio Franco Pérez**

1er SUPLENTE: **Profesor: Flores Leonar Martha Magdalena**

2ndo SUPLENTE: **Profesor: Basurto Garcia German**

SITIO DONDE SE DESARROLLO EL TEMA:
Departamento de Física y Química Teórica.
Edificio B, Posgrado, Facultad de Química, UNAM.

ASESOR:
Dr. José Marco Antonio Franco Peréz

SUSTENTANTE:
Dante Alan Jiménez Mancilla

A mi familia y amigos por ser el pilar de mi vida, sin ellos no sería nada de lo que soy ahora. A mi madre . A mis hermanos: Emmanuel y Ángel. A mis padrinos. Alejandro y Asminda. A mis lomitos: Alfredo y Ricardo.

A la música por salvarme más de una vez.

Agradecimientos

A la Universidad Nacional Autónoma de México (UNAM) y en especial a la Facultad de Química, por la excelente formación académica, la enseñanza en cada asignatura y las oportunidades que recibí a lo largo de mis estudios de licenciatura.

Al Dr. Marco Franco , el director de la presente tesis, por brindarme la oportunidad de trabajar con él, la confianza, el apoyo y el tiempo dedicado en este proyecto.

Notación

- **BOA** Aproximación de Born-Oppenheimer (*Born-Oppenheimer Aproximation*)
- **DFT** Teoría de los Funcionales de la Densidad (*Density Functional Theory*)
- **ESP** Potencial electrostático (*Electrostatic Potential*)
- **MLP** Modelo de Perceptron Multicapa (*Multilayer Perceptron*)
- **GGA** Aproximación de Gradiente Generalizado (*Generalized Gradient Approximation*)
- **GS** Estado fundamental (*Ground State*)
- **HF** Hartree-Fock
- **QTAIM** Teoria Cuantica de Atomos y Moleculas (*Quantum Theory of Atoms In Molecules*)
- **AIMALL** Programa de computo basado en QTAIM
- **BDPC** Base de Datos de Propiedades Criticas
- **BDF** Base de Datos Final
- **BDDR** Base de Datos de Descriptores de Reactividad
- **LDA** Aproximación Local de la Densidad (*Local Density Approximation*)
- **PCA** Analisis de Componentes Principales(*Principal Component Analysis)*

- **SGD** El Descenso de Gradiente Estocástico(*Stochastic Gradient Descent)*
- **PES** Superficie de energía potencial (*Potential Energy Surface*)
- **ZPE** Energía de punto cero (*Zero Point Energy*)

Introducción

En este trabajo se diseño un modelo basado en redes neuronales artificiales orientado a predecir propiedades termodinámicas del estado crítico, como la temperatura, presión y volumen críticos.Los datos de entrada fueron índices de reactividad química utilizadas para describir la capacidad de una especie química para interaccionar con su entorno. Se considera que mediante estos descriptores, será posible capturar parte de la naturaleza no ideal de un gas puro. La red neuronal optimizada establecerá la mejor relación no lineal entre nuestros parámetros de reactividad moleculares con las propiedades críticas termodinámicas.

Es importante entender las propiedades de las sustancias en su estado crítico por varias razones: en primera instancia, mediante ellas es posible predecir y modificar el comportamiento de los fluidos en condiciones extremas, como en procesos de alta presión y alta temperatura, que se utilizan en la industria química, petroquímica y de energía, así como en el diseño y optimización de procesos de separación y purificación (destilación, extracción, absorción, etc.). Se ha mostrado que las propiedades críticas están relacionadas con los coeficientes del virial, por lo que mediante ellas sería posible establecer ecuaciones de estado generales y precisas, e incluso permitiría contribuir al desarrollo de nuevas sustancias con propiedades y comportamientos específicos, como en la síntesis de nuevos materiales supercríticos, y también, en la optimización de procesos industriales, por ejemplo, en la producción de biocombustibles, en la extracción de los depósitos de petróleo y gas, así como en la química atmosférica.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Marco Teórico

1.1 Ecuaciones de Estado y Constantes Criticas

La historia de las ecuaciones de estado se remonta al siglo XIX, cuando el colega científico Johannes Diderik van der Waals comenzó a estudiar las propiedades termodinámicas de los gases y líquidos. Uno de los primeros resultados relevantes fue la ecuación de estado de van der Waals, propuesta en 1873.[1]

Esta ecuación se basa en la idea de que las moléculas de un gas ocupan un volumen finito y experimentan fuerzas de atracción entre sí. La ecuación de van der Waals se expresa como:

$$(\mathrm{P}+\frac{a}{V_m^2})(\mathrm{V_m}-\mathrm{b}) = \mathrm{RT} \tag{1.1}$$

Donde `P` es la presión, `Vm` es el volumen molar, a y b son parámetros empíricos que representan a las fuerzas de atracción y el volumen ocupado por las moléculas, respectivamente,R `R` es la constante universal de los gases y `T` es la temperatura. La ecuación de van der Waals fue un avance en la comprensión de las propiedades termodinámicas de los gases y líquidos, ya que explicaba la existencia de una temperatura crítica por encima de la cual un gas no se puede licuar, así como la formación de puntos críticos en la curva de coexistencia de fase. Desde entonces, se han desarrollado ecuaciones de estado complejas y precisas para describir

las propiedades de diversas sustancias en diferentes condiciones termodinámicas.

Dado que las ecuaciones de estado son fundamentales en la termodinámica y en la ingeniería química, permiten predecir el comportamiento de los gases y líquidos en diferentes condiciones de presión, temperatura y volumen. Estas ecuaciones son útiles para el diseño y optimización de procesos industriales, como la producción de petróleo y gas, la destilación de mezclas, el diseño de reactores químicos entre otros.[2]

Además, las ecuaciones de estado ayudan en la investigación científica y tecnológica, ya que permiten entender y modelar fenómenos complejos en la química, la física y la ingeniería. También son útiles en áreas como la geología, la astrofísica y la biología, donde se estudian fenómenos relacionados con la, la disolución de gases en líquidos, la formación de estructuras moleculares, entre otros.[3, 4]

En resumen, las ecuaciones de estado son una herramienta esencial en la ciencia y la ingeniería, y su desarrollo y aplicación continúan siendo un área de investigación activa y relevante en la actualidad. A continuación se describen las mas relevantes desde un punto de vista histórico.

Ecuación de Estado de Redlich-Kwong

La ecuación de estado de Redlich-Kwong es una ecuación utilizada en termodinámica para describir el comportamiento de los gases. Fue propuesta en 1949 por los químicos Oscar Redlich y Joseph Kwong como una mejora de la ecuación de estado de van der Waals.[5]

La importancia de la ecuación de estado de Redlich-Kwong radica en que es capaz de predecir las propiedades termodinámicas de una amplia variedad de sustancias con una mayor precisión que la ecuación de van der Waals, en particular en lo que se refiere a la predicción del comportamiento de los gases a alta presión y temperatura.

La ecuación de estado de Redlich-Kwong se basa en la teoría cinética de los gases y en la ley de los gases ideales. La ecuación tiene en cuenta tanto la interacción atractiva como repulsiva entre las moléculas, lo que permite una descripción más precisa del comportamiento

de los gases en condiciones no ideales. matemáticamente podemos expresar la ecuación de Redlich-Kwong como:

$$\mathrm{P} = (\frac{RT}{V_m - b}) - (\frac{aT^{\frac{1}{2}}}{V_m(V_m + b)}) \tag{1.2}$$

La ecuación de estado de Redlich-Kwong ha sido utilizada en una variedad de áreas del conocimiento desde la química, la ingeniería química, la física hasta la geología. Cabe mencionar que ha sido utilizada en el diseño de procesos químicos y en la simulación de sistemas termodinámicos complejos, siendo una ecuación de estado con vigencia actual.[6, 7]

Ecuación de Estado de Peng-Robinson

La ecuación de Peng-Robinson es una de las ecuaciones de estado más utilizadas en la industria química y petroquímica para la predicción de propiedades termodinámicas de fluidos, incluyendo la presión de vapor, la densidad y la capacidad calorífica. Esta ecuación fue propuesta por los colegas Ding-Yu Peng y Donald B. Robinson en 1976 como una extensión de la ecuación de estado de Redlich-Kwong.

Su definición algebraica es:

$$\mathrm{P} = (\frac{RT}{V_m - b}) - (\frac{a\alpha}{V_m^2 + 2V_m b - b^2}) \tag{1.3}$$

Para tener en cuenta el volumen ocupado por las moléculas. La ecuación de Peng-Robinson incorpora un término adicional en comparación con la ecuación de Redlich-Kwong para tener en cuenta el tamaño molecular mediante la introducción del parámetro "a"que representa la interacción atractiva entre las moléculas del fluido o gas en la ecuación. También incluye un término de corrección a la presión para tener en cuenta las interacciones entre las moléculas a alta presión y un parámetro de ajuste "b", para tener en cuenta el volumen ocupado por las moléculas.

La ecuación de Peng-Robinson ha demostrado ser más precisa que la ecuación de Redlich-Kwong para la predicción de propiedades termodinámicas de una amplia gama de compues-

tos, desde gases nobles hasta hidrocarburos complejos.[8]

Ecuación de Estado de Benedict-Webb-Rubin (BWR)

La ecuación de estado de Benedict-Webb-Rubin (BWR) es una de las ecuaciones de estado más utilizadas en la industria petroquímica para calcular propiedades termodinámicas de fluidos a diferentes condiciones de temperatura y presión. Esta ecuación de estado se basa en una expansión en serie del factor de compresibilidad Z en términos de la relación entre la presión y la temperatura reducidas, lo que permite predecir el comportamiento de gases y líquidos. Dicho esto tenemos que:

$$\mathrm{P} = (\frac{RT}{v}) + (B_0RT - A_O - \frac{C_0}{T^2})(\frac{1}{v^2}) + \frac{bRT - a}{v^3} + \frac{a\alpha}{v^6} + \frac{c}{T^2v^3}(1 + \frac{\gamma}{v^2}exp\frac{-\gamma}{v^2}) \qquad (1.4)$$

En particular, la ecuación de estado de BWR se utiliza para calcular las propiedades críticas termodinámicas de una sustancia, como la temperatura crítica, la presión crítica y el volumen crítico. Estas propiedades son de importancia en la industria química ya que permiten diseñar y operar procesos a diferentes condiciones de temperatura y presión con una alta precisión y eficiencia. En la siguiente sección se aborda la importancia del estado crítico siendo esta ecuación la más utilizada para calcular propiedades a partir de estas condiciones. Esta relevante ecuación fue desarrollada por primera vez en la década de 1940 por los químicos George C. Benedict, John W. Webb y Keith S. Rubin, y ha sido objeto de numerosos estudios y mejoras desde entonces. Su uso se ha extendido a muchos campos de la ingeniería química, incluyendo la simulación y desarrollo de procesos.

Ecuación de Estado Virial

La ecuación de estado Virial es una ecuación matemática que se usa generalmente para describir el comportamiento de los gases reales. Esta ecuación es una expansión en serie que relaciona la presión, el volumen y la temperatura de un gas en términos de su composición molecular y sus interacciones.

La ecuación de estado Virial se puede escribir de la siguiente manera:

$$P = \frac{RT}{V} + \frac{B(T)}{V^2} + \frac{C(T)}{V^3} + \ldots \tag{1.5}$$

donde P es la presión, V es el volumen molar, T es la temperatura absoluta, R es la constante de los gases y $B(T)$, $C(T)$, etc, son los coeficientes de la expansión en serie llamada coeficientes viriales. Estos coeficientes están relacionados con las interacciones moleculares en el gas y varían con la temperatura.

La ecuación de estado virial se utiliza para gases que no se comportan idealmente, como los gases a altas presiones y bajas temperaturas. A diferencia de la ecuación de estado de los gases ideales, que asume que las moléculas no interactúan entre sí, la ecuación de estado virial tiene en cuenta las interacciones moleculares y predice de manera más precisa el comportamiento de los gases reales. La ecuación Virial puede derivarse de la mecánica estadística, si se hacen las suposiciones apropiadas para modelar las fuerzas intermoleculares; incluso se pueden desarrollar expresiones teóricas para cada uno de los coeficientes.

Cabe destacar que, la ecuación de estado no es adecuada para todos los gases y condiciones de temperatura y presión. Además, la precisión de la ecuación se verá afectada por la complejidad de las interacciones moleculares y la presencia de otros factores que pueden influir en el comportamiento del gas.

1.1.1. Constantes Termodinámicas Criticas

La Ecuación de BWR es de uso para el cálculo de constantes termodinámicas críticas,las constantes involucradas se obtienen a partir de las propiedades termodinámicas de una sustancia en su estado crítico, es decir, cuando se encuentra en una condición donde no hay diferencias entre las fases líquida y gaseosa. Esto es el estado critico. Las constantes críticas más importantes son la temperatura crítica *(Tc)*, la presión crítica *(Pc)*, el volumen crítico *(Vc)* y la compresibilidad crítica *(Zc)*. [9] Estas constantes proporcionan información importante sobre el comportamiento de la sustancia en condiciones extremas.

La temperatura critica y la presión critica para mostrar su ecuación según el enfoque que sea pertinente, es decir, por medio de que ecuación de estado se expresan estas constantes para que al lector se familiarice con las ecuaciones, por definición la temperatura crítica es la temperatura más alta a la que una sustancia existe en estado líquido, se define algebraicamente en su forma reducida mediante la ecuación de estado de Van Der Waals de la siguiente forma:

$$\mathrm{T_c} = \frac{8a}{27Rb} \tag{1.6}$$

Donde R es la constante de los gases, a es el parámetro de fuerza atractiva y b es el parámetro de volumen.

Otra expresión para definir la temperatura critica es por medio de la ecuación de estado de Redlich-Kwong en su forma reducida tenemos que:

$$\mathrm{T_c} = \frac{0.0778aR}{b} \tag{1.7}$$

La ecuación de estado del Virial para la temperatura critica definida de la siguiente manera

$$\mathrm{T_c} = \left[\frac{a}{Rb}\right](1 + \frac{2b}{a})(\frac{A}{RT_c}) \tag{1.8}$$

En la ecuación de estado virial para la temperatura crítica, las variables tienen los siguientes significados:*Tc*: temperatura crítica *a*: segundo coeficiente virial a una presión igual a cero, que es una medida de la atracción intermolecular entre las moléculas, *b*: tamaño molecular efectivo, que es una medida del volumen excluido por las moléculas, *R*: constante universal de los gases, *A*: segundo coeficiente virial como función de la temperatura, que tiene en cuenta la desviación del gas del comportamiento ideal a presiones no nulas, *T*: temperatura.

Por otro lado la presión crítica se define como la presión necesaria para comprimir una sustancia hasta su punto crítico. Matemáticamente se expresa según la ecuación de Van der Waals en su forma reducida de la siguiente manera:

$$\mathrm{P_c} = \frac{a}{27b^2} \tag{1.9}$$

La de Redlich-Kwong y la ecuación de estado Virial se definen respectivamente como:

$$\mathrm{P_c} = 0.42748\frac{RT_c}{V_c}(\frac{1-b}{V_c})^2 \tag{1.10}$$

$$\mathrm{P_c} = \frac{RT_c}{V_c Z_c} \tag{1.11}$$

El volumen crítico por definición es el volumen ocupado por una sustancia en su punto crítico.

La compresibilidad crítica es una medida de la facilidad con la que una sustancia puede ser comprimida cerca de su punto crítico.

El factor acéntrico es una propiedad física que se utiliza para describir la forma en que las moléculas de una sustancia interactúan entre sí en el estado líquido o gaseoso. Es un parámetro adimensional que se define como la relación entre la presión de vapor de una sustancia en su punto normal de ebullición y la presión de vapor que tendría una sustancia ideal con la misma temperatura crítica y presión crítica, esto proporciona información acerca de la no esfericidad de las moléculas en un gas.[1]

El factor acéntrico se utiliza en la descripción de las propiedades termodinámicas de las sustancias, como la capacidad calorífica, el calor latente de vaporización y la conductividad térmica. También se utiliza en la selección de sustancias para procesos de separación y en la predicción de propiedades críticas y de fase en la industria química y de procesos. Las constantes termodinámicas críticas son importantes porque permiten predecir el comportamiento de una sustancia en condiciones extremas, como en procesos de separación, extracción, síntesis y transformación de sustancias en industrias químicas y petroquímicas. Además, el conocimiento de estas constantes es importante en el diseño de equipos y procesos, y en la optimización de procesos químicos.

1.1.2. Punto Critico y Estado Critico

El punto crítico es una condición termodinámica donde las propiedades físicas de una sustancia experimentan cambios abruptos y drásticos. En el punto critico las fases líquida y gaseosa de la sustancia se vuelven indistinguibles entre sí, y se dice que la sustancia se encuentra en un estado crítico.

En el punto crítico, la diferencia entre las propiedades termodinámicas de la fase líquida y gaseosa se reduce a cero, lo que implica que la sustancia no puede ser comprimida a un volumen menor sin experimentar una transición de fase. Además, tanto la densidad como la compresibilidad de la sustancia son extremadamente altas en el punto crítico. Por otro lado, el estado crítico se refiere a las condiciones termodinámicas específicas en las cuales una sustancia se encuentra en su punto crítico. Estas condiciones incluyen la temperatura crítica, la presión crítica y el volumen crítico de la sustancia.[10]

1.1.3. Obtención de Constantes Termodinámicas Criticas

Existen diferentes métodos experimentales de obtención de las constantes termodinámicas en su estado crítico que se tienen en diferentes artículos científicos y libros importantes. Entre los métodos más citados y destacados tenemos: [11]

Método de expansión térmica

El método de expansión térmica es una técnica experimental utilizada para determinar las constantes termodinámicas críticas de una sustancia. En este método, se mide la expansión volumétrica de la sustancia mientras se calienta gradualmente a una presión constante ligeramente inferior a la presión crítica. La expansión térmica aumenta abruptamente en el punto crítico, lo que permite determinar la temperatura crítica de la sustancia.

Este método es útil para sustancias que no son volátiles y se manejan con seguridad a altas temperaturas y presiones. La precisión de las mediciones de expansión térmica depende

de la precisión de las mediciones de temperatura y volumen, y es importante tener en cuenta los efectos de las condiciones ambientales y las impurezas en los resultados experimentales. [11]

Método de velocidad del sonido

Se basa en la medición de la velocidad del sonido en una sustancia a diferentes temperaturas y presiones, para determinar su punto crítico. En el punto crítico, la velocidad del sonido alcanza un mínimo. La teoría detrás de este método se basa en que la velocidad del sonido en un gas está relacionada con compresibilidad adiabática y capacidad calorífica que a su vez están relacionadas con las propiedades críticas.

Para llevar a cabo este método, se mide la velocidad del sonido en la sustancia a diferentes temperaturas y presiones utilizando un transductor de ultrasonidos. La velocidad del sonido se mide a lo largo de una celda de medición de longitud conocida, y se obtiene la velocidad del sonido a diferentes presiones y temperaturas. A partir de estas mediciones se construye un gráfico de la velocidad del sonido en función de la presión a diferentes temperaturas, y se identifica el punto crítico como el punto en el que la velocidad del sonido es mínima.

Este método es útil para la determinación de puntos críticos de líquidos y líquidos comprimidos, y proporciona resultados precisos y confiables. Sin embargo, su aplicación es limitada en ciertos casos, como en la medición de puntos críticos de gases altamente compresibles o sustancias tóxicas o peligrosas para manipular. [11]

Método de conductividad térmica

En este método se mide la conductividad térmica de una sustancia cerca de su punto crítico y, a partir de esta medida, obtener las constantes críticas. Este método se basa en la medición de la conductividad térmica de una muestra de la sustancia a una serie de presiones y temperaturas cercanas al punto crítico. La conductividad térmica se mide en una celda que contiene la muestra y que está rodeada por un baño termostático para controlar la temperatura.

Existen equipos comerciales que utilizan este método, como el equipo de conductividad térmica de Anton Paar. Además, se encuentra en la literatura científica diferentes investigaciones que detallan los procedimientos y los resultados obtenidos mediante este método.[12]

Método de Refracción

El Método de Refracción se basa en medir el índice de refracción de la sustancia en cuestión a diferentes temperaturas y presiones, para luego calcular las constantes críticas a partir de los valores experimentales. El índice de refracción está relacionado con las propiedades termodinámicas de la sustancia, como la densidad y la compresibilidad, y cambia drásticamente cerca del punto crítico. Por lo tanto, este método es útil para determinar las constantes críticas de sustancias transparentes como gases, líquidos y algunos sólidos cristalinos.[13]

En este método, se mide el ángulo de refracción cuando un rayo de luz atraviesa la muestra y se desvía debido a la variación en el índice de refracción. La medición se realiza a diferentes temperaturas y presiones, y se obtiene un conjunto de datos que se ajusta a una ecuación matemática para calcular las constantes críticas.[14]

Este método se ha utilizado ampliamente en la determinación de las constantes críticas de sustancias orgánicas e inorgánicas.[15]

Método de Congelamiento

El método de punto de congelamiento es una técnica experimental para determinar la temperatura crítica y la presión crítica de una sustancia. El principio básico del método es que la temperatura de congelamiento de una sustancia líquida depende de la presión. A medida que la presión disminuye, la temperatura de congelamiento disminuye también, hasta que se alcanza una presión crítica y una temperatura crítica, a partir de la cual la sustancia no puede solidificarse sin una disminución adicional de la temperatura.[16]

Para utilizar este método, se mide la temperatura de congelamiento de la sustancia líquida a diferentes presiones, utilizando un dispositivo de congelación que permite registrar del

proceso de congelación. La presión crítica y la temperatura crítica se determinan a partir de una curva que muestra la relación entre la temperatura de congelamiento y la presión. Este método es particularmente útil para sustancias que tienen una temperatura de congelamiento relativamente alta.[13]

1.1.4. Usos y Aplicaciones

Usos y aplicaciones de las constantes termodinámicas en su estado critico. A continuación se mencionan algunos de los más importantes.

Diseño de equipos de procesos

Las constantes críticas se utilizan para diseñar equipos de procesos, como reactores y columnas de destilación, para garantizar que las condiciones de operación estén por encima de las condiciones críticas. Por ejemplo, en el diseño de ciertos reactores, se debe garantizar que la temperatura y la presión de operación sean mayores que las constantes críticas para evitar que la reacción se detenga o se produzcan condiciones inestables. De esta manera, se asegura una operación eficiente del equipo.

Extracción de petróleo y gas

Las constantes críticas se utilizan para determinar las propiedades de los fluidos que se encuentran en los yacimientos de petróleo y gas, lo que ayuda a optimizar la producción, extracción y el procesamiento.[17]

Producción de alimentos

Las constantes críticas se utilizan en la producción de alimentos para determinar las propiedades de los fluidos, como la solubilidad y la viscosidad, lo que ayuda a optimizar los procesos de producción.[18]

Industria química

Las constantes críticas se utilizan para diseñar y optimizar procesos químicos, como la síntesis de polímeros y la producción de productos químicos.[19]

SubsubsecciónInvestigación científica Las constantes críticas se utilizan en la investigación científica para estudiar el comportamiento de los fluidos en condiciones extremas.

Desarrollo de tecnología de energía renovable

Las constantes críticas se utilizan en el desarrollo de tecnologías de energía renovable, como la captura de dióxido de carbono y la producción de biocombustibles.[20]

Industria farmacéutica

Las constantes críticas se utilizan en la producción de medicamentos y productos farmacéuticos para determinar las propiedades de los fluidos utilizados en los procesos de fabricación. [21]

Producción de plásticos

Las constantes críticas se utilizan en la producción de plásticos para determinar las propiedades de los fluidos utilizados en los procesos de fabricación. [21]

Industria textil

Las constantes críticas se utilizan en la producción de textiles para determinar las propiedades de los fluidos utilizados en los procesos de fabricación. [21]

Diseño de motores y turbinas

Las constantes críticas se utilizan en el diseño de motores y turbinas para garantizar que las condiciones de operación sean aceptable.[21]

1.1.5. Principio de Los Estados Correspondientes

El Principio de Estados Correspondientes es un concepto importante en la termodinámica y la física de fluidos. Este teorema establece que diferentes sustancias exhiben un comportamiento similar cuando se encuentran en condiciones termodinámicas adecuadas. En otras palabras, el comportamiento de ciertas propiedades de sustancias diferentes, como la presión, la temperatura y el volumen, se vuelve similar en relación con ciertas condiciones específicas.

El Principio de Estados Correspondientes se basa en la idea de que los gases y líquidos a menudo se comportan de manera similar en términos de sus propiedades termodinámicas cuando se escalan apropiadamente. Esto significa que si dos sustancias tienen la misma relación de propiedades, como la relación entre la presión y la temperatura, entonces sus comportamientos termodinámicos pueden ser muy similares en ciertas condiciones.

El Principio de Estados Correspondientes se ha utilizado para desarrollar ecuaciones de estado que describen el comportamiento de los gases y líquidos en una amplia variedad de condiciones. Un ejemplo es la ecuación de estado de Van der Waals, que se basa en el concepto de estados correspondientes y se utiliza para describir el comportamiento de los gases reales en condiciones no ideales. [22]

1.1.6. Principio de Correspondencia

El Principio de Correspondencia es un concepto fundamental en la física estadística que establece una relación entre los parámetros termodinámicos macroscópicos y las propiedades microscópicas de un sistema termodinámico. En particular, el principio de correspondencia establece que las propiedades termodinámicas macroscópicas de un sistema en equilibrio deben corresponder a las propiedades microscópicas de las partículas que lo conforman.

En el caso del estado crítico, el principio de correspondencia es especialmente importante, ya que las propiedades termodinámicas de un fluido en su punto crítico están influenciadas por la estructura microscópica del sistema. En particular, en el punto crítico, el comportamiento del fluido se vuelve no lineal y se produce una transición de fase de segunda orden. La comprensión de estas propiedades críticas es crucial para entender la física de los procesos termodinámicos y para el diseño de sistemas y procesos que involucren fluidos críticos.

En resumen, el principio de correspondencia establece una relación entre las propiedades macroscópicas y microscópicas de un sistema termodinámico, y es esencial para comprender las propiedades termodinámicas de un fluido en su estado crítico. [23].

1.2 Aprendizaje Profundo de Maquina

Se tomó como punto de partida el "deep learning", aprendizaje profundo ya que al enfrentar problemas complejos en el área de la química teórica estas técnicas permiten aprender patrones y características de datos de una manera eficiente y precisa que otras herramientas de la ciencia de datos. Esto es útil cuando los datos son complejos y no tienen una estructura clara o cuando hay variables interrelacionadas que influyen en los resultados. La respuesta tiene dependencia no lineal con las variables de entrada

El aprendizaje profundo se basa en una arquitectura de red neuronal que se inspira en la forma en que el cerebro humano procesa la información. La red neuronal tiene múltiples capas ocultas que identifican patrones y características en los datos de entrada. A medida que la red se entrena con más datos, se ajusta y mejora continuamente su capacidad para identificar patrones mejorando la calidad de las predicciones proporcionadas[24].

1.2.1. Algoritmos de Redes Neuronales Artificiales

Los algoritmos de redes neuronales artificiales son técnicas de aprendizaje automático. Estas redes están compuestas por un conjunto de nodos interconectados (neuronas artificiales) que procesan y transmiten información a través de conexiones ponderadas. [25]

A continuación mencionaré algunos de los algoritmos de redes neuronales artificiales más comunes:

1. Perceptrón: es un modelo de red neuronal con una única capa de neuronas. Es utilizado para resolver problemas de clasificación binaria.[25]

2. Redes neuronales multicapa (MLP): son redes con múltiples capas de neuronas interconectadas. Estas redes se utilizan para problemas complejos como el reconocimiento de patrones, la clasificación de imágenes y el procesamiento del lenguaje natural.[25]

3. Redes neuronales recurrentes (RNN): son redes en las que la salida de una capa se

alimenta como entrada en la siguiente capa. Estas redes son útiles para problemas en los que se requiere un análisis temporal de los datos, como el reconocimiento de voz y la predicción del mercado financiero.[25]

4. Redes neuronales convolucionales (CNN): son redes que utilizan capas convolucionales para detectar patrones en imágenes y otros datos espaciales. Estas redes son ampliamente utilizadas en tareas de visión por computadora y reconocimiento de objetos.[25]

5. Redes neuronales generativas adversariales (GAN): son redes que generan muestras de datos sintéticos a través de una competencia entre dos redes: una red generadora y una red discriminadora. Estas redes se utilizan para la generación de imágenes, música y otras formas de datos complejos.[25]

1.2.2. Perceptron Primer Modelo De Aprendizaje de Maquina

Existen variados tipos de modelos de redes neuronales según el problema o análisis que se quiera llevar acabo como ejemplo simple pero representativo abordaremos la arquitectura del perceptrón que historicamente fue el primer algoritmo en su tipo propuesto por Frank Rosenblatt en el año 1957.[26]

El esquema de esta arquitectura de red se muestra en la figura 1

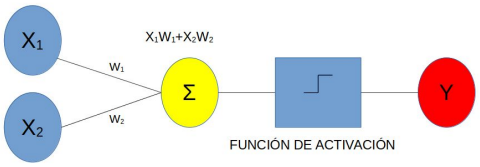

Figura 1.1: Diagrama Estructural Modelo Perceptron

Donde X_1, X_2 son inputs o variables de entrada a los cuales se les asigna un peso w_1, w_2, el peso es la influencia relativa para cada valor de entrada esto se multiplica y después se hace

la suma ponderada para cada valor de entrada con su respectivo peso, por ultimo se pasa por la función de activación que para este modelo es la función escalón o en ingles "Heaviside", esta función introduce no linealidad al modelo y determina si el perceptron o neurona se activa o no. [25]

$$F(x) = \begin{cases} 1 & x \cdot w + b \\ 0 & diferente \end{cases} \tag{1.12}$$

Aplicando la función escalón al modelo de perceptrón si el algoritmo cumple con la ecuación 1.12 el valor de salida será verdadero, es decir el perceptrón funciona como un clasificador binario donde la función mapea su valor de entrada x (un vector de valor real) a un valor de salida $f(x)$(un único valor binario).

$$f = \sum_{i=1}^{m} w_i x_i \tag{1.13}$$

w es un vector de los pesos asignados por valor de entrada, $x \cdot w$ es el producto punto, m es el número de entradas al perceptrón y b es el sezgo o bias. Entonces el perceptrón nos dará un único valor de salida que se interpreta como la clasificación del ejemplo de entrada en una de las dos clases posibles, generalmente 0 y 1, o -1 y 1, dependiendo de la codificación utilizada.

Durante el proceso de entrenamiento, los pesos del perceptrón se ajustan para reducir el error de clasificación. Se utilizan algoritmos de aprendizaje, como el algoritmo de descenso de gradiente, para actualizar los pesos en función del error cometido en la clasificación de los ejemplos de entrenamiento, como ultima parte del proceso tenemos la iteración y convergencia: El proceso de ajuste de pesos se repite iterativamente para múltiples ejemplos de entrenamiento hasta que se alcanza un criterio de convergencia, como un número máximo de iteraciones o cuando el error de clasificación es lo suficientemente bajo.

A continuación, algunas ventajas y desventajas del perceptrón:

Ventajas del perceptrón:

1. Sencillez: el perceptrón es un modelo de aprendizaje muy simple y fácil de entender.

Su estructura básica consiste en una única capa de neuronas, lo que lo hace relativamente fácil de implementar y entrenar.

2. Eficiencia computacional: debido a su simplicidad, el perceptrón es eficiente computacionalmente. El proceso de entrenamiento y predicción es rápido, especialmente para conjuntos de datos pequeños y simples.

3. Interpretabilidad: dado que el perceptrón es un modelo lineal, los pesos asignados a cada característica proporcionan información sobre su importancia relativa para la clasificación. Esto hace que el modelo sea interpretable y útil para extraer conocimiento sobre los datos.[27]

Desventajas del perceptrón:

1. Limitaciones en la capacidad de representación: el perceptrón sólo aprende si los datos son intrínsecamente no lineales, el perceptrón no podrá clasificarlos de manera precisa.[27]

2. Sensibilidad a valores atípicos y ruido: El perceptrón es sensible a valores atípicos y ruido en los datos. Incluso un solo valor atípico afecta el proceso de entrenamiento y hacer que el modelo se equivoque en sus predicciones.[27]

3. No garantiza convergencia para problemas no linealmente separables: Si los datos de entrada no son linealmente separables, es decir, no separa completamente con una línea recta, el perceptrón no convergerá en su entrenamiento. En tales casos, se requieren modelos, como redes neuronales multicapa, para abordar estos problemas.[27]

Es importante tener en cuenta que estas ventajas y desventajas se refieren específicamente al perceptrón en su forma básica. A lo largo de los años, se han desarrollado diversas variantes y mejoras del perceptrón para superar algunas de estas limitaciones, como las redes neuronales multicapa o los algoritmos de aprendizaje profundo.[27]

1.2.3. Modelo Multicapa

Los modelos multicapa son una extensión del perceptrón básico que permiten superar las limitaciones de representación de funciones no lineales. Estos modelos también se conocen como redes neuronales artificiales o redes neuronales multicapa.

El funcionamiento de los modelos multicapa se basa en la composición de múltiples capas de neuronas artificiales interconectadas. Cada capa está compuesta por un conjunto de neuronas, y las conexiones entre las neuronas están asociadas con pesos.[27]

A continuación una descripción básica del funcionamiento de los modelos multicapa:

1. Capa de entrada: la capa de entrada recibe los datos de entrada, que son características o atributos que sirven de ejemplos para el aprendizaje del algoritmo. Cada nodo en la capa de entrada representa una característica y transmite la información a la siguiente capa.

2. Capas ocultas: Las capas ocultas se encuentran entre la capa de entrada y la capa de salida. Estas capas contienen neuronas intermedias que procesan y transforman la información. Habrá una o varias capas ocultas eso depende la arquitectura del modelo, cada neurona en una capa oculta está conectada a todas las neuronas de la capa anterior y de la siguiente capa.

3. Pesos y funciones de activación: Cada conexión entre neuronas tiene asociado un peso, que determina la importancia de la señal transmitida por esa conexión. Además, cada neurona en una capa oculta y en la capa de salida aplica una función de activación no lineal a la suma ponderada de las señales recibidas.

4. Propagación hacia adelante: La información fluye desde la capa de entrada hacia las capas ocultas y finalmente hacia la capa de salida. Cada neurona en una capa oculta calcula la suma ponderada de las salidas de las neuronas en la capa anterior, aplica la función de activación y transmite la salida a las neuronas en la capa siguiente.

5. Capa de salida: La capa de salida produce la salida final del modelo. tiene una o varias neuronas dependiendo del tipo de problema, como clasificación multiclase o regresión. La

salida de la capa de salida se interpreta como la predicción del modelo.

6. Ajuste de pesos: Durante el entrenamiento, los pesos de las conexiones se ajustan utilizando algoritmos de aprendizaje, como el algoritmo de retropropagación (backpropagation). Este algoritmo calcula el gradiente de una función de pérdida o error y propaga el error hacia atrás a través de las capas de la red para ajustar los pesos de manera iterativa.

Los modelos multicapa son capaces de aprender representaciones y relaciones no lineales entre las características de entrada, lo que los hace más flexibles y poderosos que el perceptrón básico. Sin embargo, su entrenamiento es más complejo y requiere más datos y capacidad computacional.[27]

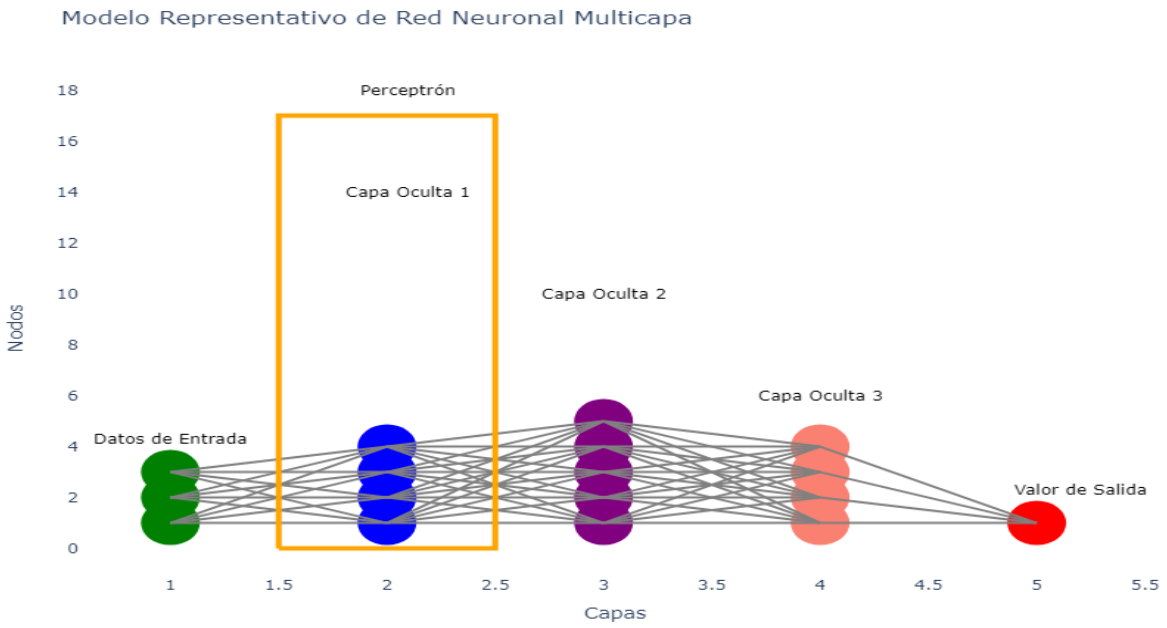

Figura 1.2: Diagrama de Red Neuronal Modelo Multicapa (MLP) Datos de entrada en color verde, primera capa oculta color azul mostrando el modelo de perceptrón como un componente del MLP, segunda capa oculta color morado, tercera capa color salmón y valor de salida en color rojo.

Figura 1.3: Diagrama de Flujo: Funcionamiento de un Modelo MLP

1.2.4. Hiperparámetros de Red

En el contexto del aprendizaje profundo y las redes neuronales, los hiperparámetros de red son parámetros configurables que no se aprenden directamente durante el proceso de entrenamiento del modelo. En cambio, determinan aspectos fundamentales de la arquitectura y el comportamiento de la red. los hiperparámetros utilizados en el proyecto son la cantidad de capas y neuronas en la red, las funciones de activación utilizadas en cada capa, la tasa de aprendizaje del optimizador, el algoritmo de optimización, el *batch size*, el número de épocas de entrenamiento. Elegir los valores adecuados para los hiperparámetros es esencial para lograr un buen rendimiento del modelo y evitar el sobre-ajuste.[28]

1.2.5. Herramientas de Análisis de Red

Las herramientas de análisis de red se refieren a técnicas y métodos utilizados para comprender y evaluar el funcionamiento y el rendimiento de las redes neuronales y otros modelos de aprendizaje automático. Estas herramientas pueden abarcar desde técnicas visuales, como

la visualización de gráficos de pérdida y precisión durante el entrenamiento, hasta métodos más avanzados, como el análisis de salidas intermedias de capas para comprender cómo el modelo procesa la información. Además, las herramientas de análisis pueden incluir la evaluación de métricas de rendimiento como la precisión. El análisis de errores y la interpretación de las decisiones del modelo también son componentes importantes de estas herramientas, A continuación menciono algunas de las herramientas mas importantes en la evaluación y eficiencia de Redes Neuronales Artificiales.[28]

1. **MSE (Mean Squared Error):** El Error Cuadrático Medio es una métrica que mide el promedio de los errores al cuadrado entre las predicciones del modelo y los valores reales. Cuanto más bajo sea el valor del MSE, mejor se ajustará el modelo a los datos.[28] El MSE penaliza los errores grandes de manera más significativa que los errores pequeños debido a la elevación al cuadrado.

2. **RMSE (Root Mean Squared Error):** La Raíz Cuadrada del Error Cuadrático Medio es simplemente la raíz cuadrada del MSE. Proporciona una medida del error promedio en las mismas unidades que los datos originales, lo que facilita la interpretación. Un RMSE más bajo indica un ajuste más preciso del modelo.[28]

3. **MAE (Mean Absolute Error):** El Error Absoluto Medio mide el promedio de las diferencias absolutas entre las predicciones del modelo y los valores reales.[28] A diferencia del MSE, el MAE trata los errores de manera lineal, lo que significa que todos los errores tienen el mismo peso en la métrica.

4. **R Cuadrado (Coeficiente de Determinación):** El Coeficiente de Determinación, comúnmente conocido como R cuadrado, es una medida que indica la proporción de la variabilidad total de la variable dependiente que es explicada por el modelo. Un R cuadrado cercano a 1 sugiere que el modelo explica el %100 de la variabilidad en los datos.[28]

5. **Gráfico de Pérdida:** Un gráfico que representa cómo cambia la función de pérdida a medida que el modelo se entrena con más datos. [28]Generalmente, la pérdida dismi-

nuye a medida que avanza el entrenamiento. Observar este gráfico ayuda a evaluar si el modelo se está ajustando correctamente.

En el ámbito del modelo de aprendizaje automático, la curva de pérdida representa cómo cambia la función de pérdida (en este caso, el Error Cuadrático Medio, MSE) a medida que el modelo se entrena en diferentes épocas. Es una herramienta útil para evaluar el rendimiento del modelo a lo largo del proceso de entrenamiento.

Cuando observamos las curvas de pérdida de entrenamiento, validación y prueba, podemos hacer las siguientes observaciones:

Curva de Pérdida de Entrenamiento: - La curva de pérdida de entrenamiento muestra cómo el MSE disminuye a medida que el modelo se entrena en el conjunto de entrenamiento durante las épocas. Inicialmente, la pérdida al principio es alta, pero debería disminuir a medida que el modelo ajusta sus parámetros para hacer predicciones más precisas en el conjunto de entrenamiento. Esto se debe a que el modelo se está "memorizando"los datos de entrenamiento y optimizando sus parámetros para reducir la pérdida en ese conjunto específico.[28] - Sin embargo, una característica importante de esta curva es que generalmente tenderá a disminuir continuamente con cada época y generalmente muestra una reducción significativa en las primeras etapas del entrenamiento. Esto se debe a que el modelo se ajusta a los datos de entrenamiento y aprende a representar los patrones y relaciones presentes en ese conjunto de datos específico[28]

Curva de Pérdida de Validación: - La curva de pérdida de validación muestra cómo el MSE cambia a medida que el modelo se evalúa en el conjunto de validación durante las épocas. Este conjunto de datos no se utiliza para entrenar el modelo y, por lo tanto, proporciona una evaluación imparcial del rendimiento del modelo en datos "nuevos"que no ha visto durante el entrenamiento.[28] - Lo que esperamos ver en esta curva es que la pérdida inicialmente disminuya a medida que el modelo generaliza bien a los datos de validación y mejora su rendimiento. Sin embargo, en algún momento, la pérdida podría empezar a aumentar nuevamente si el modelo comienza a sobreajustarse (overfitting) a los datos de entrenamiento y no generaliza bien a datos nuevos.[28]

Curva de Pérdida de Prueba: - La curva de pérdida de prueba muestra cómo el MSE

cambia a medida que el modelo se evalúa en el conjunto de prueba durante las épocas. El conjunto de prueba también es independiente del entrenamiento y la validación y se utiliza para una evaluación final del rendimiento del modelo en datos completamente nuevos. La expectativa es que esta curva sea similar a la de validación y muestre un rendimiento general estable o una ligera disminución en la pérdida a medida que el modelo generaliza bien a los datos de prueba. La convergencia de esta curva a valores bajos indica que el modelo está generalizando bien y puede hacer predicciones precisas en nuevos datos.[28]

6. **Gráfico de Calibración:** Un gráfico que compara las probabilidades pronosticadas por el modelo con las frecuencias observadas. En un gráfico de calibración ideal, las probabilidades pronosticadas coinciden perfectamente con las frecuencias observadas, lo que indica que el modelo es confiable en sus predicciones de probabilidad.[28]

7. **Gráfico de Residuales:** Un gráfico que muestra las diferencias entre los valores reales y las predicciones del modelo. Si los residuales están distribuidos aleatoriamente alrededor de cero y no muestran un patrón discernible, esto sugiere que el modelo está haciendo predicciones razonablemente precisas. En la gráfica de residuales, es importante buscar lo siguiente:

   Homocedasticidad: Los residuales deberían tener una dispersión constante en todo el rango de predicciones. Si hay una variación no constante en la dispersión, puede indicar que el modelo tiene problemas de heterocedasticidad. Linealidad: Los residuales deberían estar distribuidos aleatoriamente alrededor del eje y = 0, sin una tendencia clara hacia arriba o hacia abajo. Si hay una tendencia, podría sugerir que el modelo no está capturando adecuadamente la relación entre las variables y los valores reales.

8. **Comparación de Datos Reales y Predicciones:** Un gráfico que superpone los valores reales y las predicciones del modelo. Este gráfico permite visualizar cómo se ajustan las predicciones del modelo a los datos reales. Un buen ajuste se reflejará en puntos que están cerca de una línea diagonal.[28]

9. **Incertidumbre por Modelo:** la incertidumbre se refiere a cuánta confianza se tiene

en las predicciones del modelo. Se mide utilizando intervalos de confianza, desviaciones estándar u otras métricas estadísticas. Una mayor incertidumbre indica que las predicciones pueden ser menos confiables.[28]

1.2.6. Consideraciones Éticas y de Validación

En el entorno del aprendizaje automático y la química computacional, es crucial abordar las **consideraciones éticas** asociadas con la implementación de modelos predictivos en aplicaciones del mundo real. Estos modelos tienen un impacto significativo en diversas áreas, como la industria, la medicina y la investigación, lo que resalta la necesidad de asegurar que su uso sea responsable y beneficioso para la sociedad.

La **transparencia y la interpretabilidad** de los modelos son aspectos clave desde un punto de vista ético. A medida que los modelos de redes neuronales se vuelven más complejos, puede ser desafiante comprender cómo toman decisiones. Es importante implementar enfoques que permitan desglosar y explicar las predicciones del modelo, especialmente en aplicaciones.

En cuanto a la **validación y la confiabilidad** de los resultados, es esencial realizar una evaluación rigurosa de los modelos. La **validación cruzada** es una técnica común para evaluar el rendimiento de un modelo en datos no vistos. Esto ayuda a evitar el sobreajuste y proporciona una estimación más precisa del rendimiento del modelo en situaciones reales.

Adicionalmente, la **evaluación adecuada de la incertidumbre** es esencial para comprender los límites de las predicciones del modelo. Los modelos de redes neuronales, si bien en ocasiones son altamente precisos, también pueden tener limitaciones y errores. La estimación de la incertidumbre ayuda a cuantificar la confianza en las predicciones y proporciona información valiosa para la toma de decisiones informadas.

El desarrollo y la implementación de modelos predictivos en química requieren una consideración ética y rigurosa validación. Abordar aspectos como la transparencia, la equidad y la confiabilidad es esencial para garantizar un uso responsable y beneficioso de la tecnología

en aplicaciones del mundo real.

El análisis conjunto de las métricas y la incertidumbre proporciona una visión completa del desempeño de cada modelo y sienta las bases para futuros desarrollos en esta área de estudio.

Capítulo 2

Teoría cálculos de estructura electrónica

La ecuación de Schrödinger, dependiente del tiempo, desempeña un papel central e indispensable en la teoría de la mecánica cuántica u operador hamiltoneano. [29]:

$$i\frac{\partial}{\partial t}\Phi(\vec{r},t) = \widehat{H}\Phi(\vec{r},t) \tag{2.1}$$

La ecuación anterior resume las posiciones de todas las partículas que componen el sistema, con t denotando el tiempo y $\widehat{H}$ representando un operador de energía.

Cuando se trabaja con átomos y moléculas libres de perturbaciones externas, estos poseen propiedades bien definidas que no varían con el tiempo. En este caso, se realiza una separación de variables $\Phi(\vec{r},t) = f(t)\Psi(\vec{r})$, donde $f(t)$ es la parte temporal y $\Psi(\vec{r})$ satisface la ecuación de Schrödinger independiente del tiempo.

La ecuación de Schrödinger independiente del tiempo se expresa como:

$$\widehat{H}\Psi_n = E_n\Psi_n \tag{2.2}$$

donde $\widehat{H}$ es el operador hamiltoniano asociado a la energía del sistema E_n.

Los sistemas químicos que consisten en múltiples electrones, la función de onda para sis-

temas polielectrónicos se obtiene resolviendo la ecuación de Schrödinger mediante la correcta formulación de los operadores para las partículas que componen el sistema. [30] Para un sistema molecular electrónico, el hamiltoniano en unidades atómicas se expresa de la siguiente manera:[30]:

$$\widehat{H} = -\frac{1}{2}\sum_{i=1}^{N}\nabla_i^2 - \sum_{i=1}^{N}\sum_{A=1}^{M}\frac{Z_A}{r_{iA}} + \sum_{i=1}^{N}\sum_{i<j}^{N}\frac{1}{r_{ij}} - \frac{1}{2}\sum_{A=1}^{M}\frac{1}{m_A}\nabla_A^2 + \sum_{A=1}^{M}\sum_{A<B}^{M}\frac{Z_A Z_B}{r_{AB}} \tag{2.3}$$

El primer término abarca la contribución de la energía cinética asociada al movimiento de los N electrones en el sistema. El segundo término engloba la interacción electrostática entre los N electrones y los M núcleos, considerando las cargas opuestas. El tercer término representa la repulsión electrostática entre los electrones, ya que al ser partículas cargadas negativamente, se repelen entre sí. El cuarto término describe la energía cinética asociada al movimiento de los núcleos en el sistema. Por último, el quinto término corresponde a la interacción potencial entre los núcleos, que depende de sus posiciones relativas en el sistema.[29]:

Estos términos constituyen los componentes esenciales del hamiltoniano completo y su consideración permite describir y comprender la dinámica y las propiedades de sistemas químicos a escala microscopica.[29]:

Dado lo complejo que resulta encontrar una solución para el hamiltoniano completo 2.3, ya que implica términos nucleares y electrónicos acoplados, se recurre a aproximaciones para simplificar su tratamiento. Una de estas aproximaciones es la conocida como aproximación de Born-Oppenheimer (BOA).[29].

La idea principal de la aproximación de Born-Oppenheimer se basa en el hecho de que los núcleos son más pesados que los electrones, lo que implica que su velocidad es menor en comparación con la velocidad de los electrones. Por lo tanto, se considera que los núcleos permanecen estáticos mientras los electrones se mueven en su campo.[29]

Esta aproximación permite separar los grados de libertad de los electrones y los núcleos, lo que simplifica el problema. Bajo esta consideración se puede resolver la ecuación de Schrödinger para los electrones en el campo de los núcleos, obteniendo así la distribución electró-

nica y las propiedades asociadas. Luego, teniendo en cuenta esta distribución electrónica, se calcula la energía potencial total del sistema y se resuelve las ecuaciones de movimiento para los núcleos, determinando así su dinámica.[30]

La aproximación de Born-Oppenheimer ha demostrado ser muy exitosa en la descripción de sistemas moleculares, ya que permite tratar los movimientos de los electrones y los núcleos de manera separada, simplificando el problema general permitiendo la resolución de la Ecuacion de Schrödinger.[30] Dicho esto tenemos que el hamiltoniano representado en la ecuación 2.3 se puede reescribir de la siguiente forma:

$$\widehat{H} = -\frac{1}{2}\sum_{i=1}^{N}\nabla_i^2 - \sum_{i=1}^{N}\sum_{A=1}^{M}\frac{Z_A}{r_{iA}} + \sum_{i=1}^{N}\sum_{i<j}^{N}\frac{1}{r_{ij}} + E_{nuc} = \widehat{H}_{elec} + E_{nuc}, \tag{2.4}$$

En la ecuación (2.4), se observa una distinción entre las coordenadas nucleares y electrónicas en el hamiltoniano total. Se desprecia inicialmente la contribución de la energía cinética de los núcleos, y la interacción repulsiva entre ellos se considera como una constante adicional.

La aproximación de tratar las coordenadas nucleares y electrónicas por separado simplifica en gran medida el tratamiento del sistema, ya que se desacoplan los grados de libertad electrónicos de los nucleares. El hamiltoniano electrónico[31] ($\widehat{H}elec$) se compone de varios términos, siendo el primero de ellos la energía cinética de los electrones. Esta energía cinética se representa como ($-\frac{1}{2}\sum\nabla_i^2$), donde N es el número de electrones y ∇_i^2 es el operador laplaciano aplicado al i-ésimo electrón. [30] El segundo término del hamiltoniano electrónico involucra la interacción coulómbica entre los electrones y los núcleos. En esta interacción, se considera la suma de las contribuciones de cada electrón (i) y cada núcleo (A), y sigue la ley de Coulomb. Matemáticamente, se expresa como $-\sum_{i=1}^{N}\sum_{A=1}^{M}\frac{Z_A}{r_{iA}}$, donde Z_A representa el numero atómico del núcleo A, y r_{iA} es la distancia relativa al electrón i respecto al núcleo A. Este término refleja la interacción electrostática atractiva entre los electrones y los núcleos, donde los electrones son atraídos hacia los núcleos debido a la diferencia de carga. [31] La suma se realiza sobre todos los electrones y todos los núcleos presentes en el sistema. La interacción coulómbica juega un papel fundamental en la estabilidad y las propiedades de los sistemas atómicos y moleculares. Es responsable de la formación de enlaces químicos y

de la estructura electrónica de los átomos y moléculas. El tercer término del hamiltoniano electrónico corresponde a la repulsión electrostática entre los electrones. En este término, se realiza una suma que contempla todas las posibles combinaciones de pares de electrones (i y j) presentes en el sistema. Matemáticamente, se expresa como $\sum_{i=1}^{N} \sum_{i<j}^{N} \frac{1}{r_{ij}}$, donde r_{ij} representa la distancia entre los electrones i y j. Al ser partículas con carga del mismo signo, experimentan una fuerza de repulsión entre sí.

Por otro lado, la energía nuclear (E_{nuc}) se trata como una constante en el hamiltoniano electrónico y tiene en cuenta la repulsión entre los núcleos. Dado que los núcleos también tienen misma carga entre sí, experimentan una fuerza de repulsión electrostática.

El hamiltoniano electrónico, denotado como $\widehat{H}_{elec}$ en la ecuación (2.4), se refiere a la parte del hamiltoniano que se utiliza ampliamente en cálculos de estructura electrónica. La ecuación de Schrödinger correspondiente a la parte electrónica se expresa como:

$$\widehat{H}_{elec}\Psi_{elec} = E_{elec}\Psi_{elec} \tag{2.5}$$

$$\Psi_{elec} = \Psi_{elec}(r_i; R) \tag{2.6}$$

Aquí, Ψ_{elec} es la función de onda electrónica y E_{elec} es la energía electrónica asociada a dicha función de onda.

En la ecuación (2.6), se observa que la función de onda electrónica, Ψ_{elec}, depende de las coordenadas electrónicas r_i (que representan las posiciones de los electrones) y de las coordenadas nucleares R (que indican las posiciones de los núcleos) paramétricamente. Esta parametrización refleja cómo la función de onda electrónica se modifica en respuesta a las posiciones de los núcleos en el sistema.[30]

Un concepto inherente a la BOA es el de la superficie de energía potencial (PES) donde se representa la relación entre la energía de una molécula y su geometría molecular.

2.1 Método de Hartree-Fock

Dado que la resolución analítica de la ecuación 2.2 utilizando el Hamiltoniano de la ecuación 2.4 resulta intratable para sistemas químicos constituidos por más de un electrón su resolución se realiza mediante enfoques aproximados. Entre ellos, el método de Hartree-Fock se destaca por su base teórica y su influencia en metodologías posteriores [30, 31].

Se deben resaltar algunas de las principales características de este enfoque, ya que serán relevantes al abordar la teoría de los funcionales de la densidad (DFT). El método de Hartree-Fock opera dentro de la Aproximación BOA que se ha discutido previamente.[31] Es importante destacar que la energía total del sistema está influenciada por las posiciones de los M nucleos. La expresión de la energía total E_{tot} es la siguiente:

$$E_{tot} = \varepsilon_{elec} + \varepsilon_{nuc} \tag{2.7}$$

Aquí, ε_{nuc} representa el valor esperado del operador $\widehat{V}_{NN}$.

El método es de naturaleza variacional, lo que significa que, para el estado fundamental, la energía E, calculada utilizando una función de onda aproximada Ψ, actúa como un límite superior para la energía del sistema (E_0), que se obtendría utilizando la función de onda exacta Ψ_0. Esto refleja cómo el método progresivamente mejora la función de onda del sistema de manera autoconsistente:

$$\langle \Psi_0 | \widehat{H} | \Psi_0 \rangle \equiv E_0 \leq E \equiv \langle \Psi | \widehat{H} | \Psi \rangle \tag{2.8}$$

La función de onda de un sistema polielectrónico se aproxima utilizando determinantes de Slater:

$$\psi(r,s) = \frac{1}{\sqrt{N!}} \begin{vmatrix} \chi_i(x_1) & \chi_j(x_1) & \dots & \chi_k(x_1) \\ \chi_i(x_2) & \chi_j(x_2) & \cdots & \chi_k(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \chi_i(x_N) & \chi_j(x_N) & \cdots & \chi_k(x_N) \end{vmatrix} \tag{2.9}$$

Donde N representa el número de electrones en el sistema y las χ son las funciones monoelectronicas Es fundamental señalar que la igualdad en la ecuación 2.8 ($E_0 = E$) se logra cuando $\Psi = \Psi_0$, es decir, cuando la función de onda utilizada en el cálculo es exactamente igual a la función de onda del estado fundamental del sistema.[31]

Los determinantes de Slater son combinaciones lineales con antisimetría de productos de funciones de onda monoelectrónicas espaciales que también incluyen la propiedad de espín de los electrones, conocidas como espín-orbitales $\chi_k(x_i)$. Estos espín-orbitales se definen como el producto de un orbital espacial $\phi_j(r_i)$ y una función de espín $\alpha(s_i)$ para el espín positivo y $\beta(s_i)$ para el espín negativo.[31]

$$\chi_k(x_i) = \begin{cases} (\phi_j(r_i)\alpha(s_i) & \text{para el espín positivo,} \\ \phi_j(r_i)\beta(s_i) & \text{para el espín negativo.} \end{cases} \tag{2.10}$$

Esta definición asegura que el intercambio de dos electrones cambie el signo de la función de onda. Esto es esencial para describir partículas fermiónicas, como los electrones, ya que siguen el principio de exclusión de Pauli, donde dos electrones con el mismo espín no pueden ocupar el mismo espacio espín-orbital.[31]

Dentro del método Hartree-Fock El operador de Fock $\widehat{f}$ se define como:

$$\widehat{f}(1) = \widehat{h}(1) + \widehat{v}_{eff}(1) \tag{2.11}$$

Donde $\widehat{h}(1)$ es el Hamiltoniano monoelectrónico y el potencial efectivo $\widehat{v}_{eff}(1)$ se refiere al potencial generado por todos los electrones excepto el electrón 1. Este operador de Fock es crucial en el método de Hartree-Fock, ya que encapsula las contribuciones monoelectrónicas

y el efecto del entorno electrónico en la energía del sistema.[31]

$$\widehat{v}_{eff}(1) = \sum_{b}^{N} \widehat{J}_b(1) + \widehat{K}_b(1) \tag{2.12}$$

Los operadores $\widehat{J}_b(1)$ y $\widehat{K}_b(1)$ son operadores coulómbicos y de intercambio, respectivamente. Estos operadores actuan sobre espín-orbitales centrados en las coordenadas del electrón 1,

$$\widehat{J}_b(1)\chi_a(1) = \left(\int dx_2 \chi_b^*(2) \frac{1}{r_{ij}} \chi_b(2) \right) \chi_a(1) \tag{2.13}$$

$$\widehat{K}_b\chi_a(1) = \left(\int dx_2 \chi_b^*(2) \frac{1}{r_{ij}} \chi_a(2) \right) \chi_b(1) \tag{2.14}$$

Si se toma el producto interno de 2.13 y 2.14 con $\chi_a(1)$ a la izquierda,

$$\langle \chi_a(1) | \widehat{J}_b(1)\chi_a(1) \rangle \equiv \langle ab|ab \rangle = J_{ab} \tag{2.15}$$

$$\langle \chi_a(1) | \widehat{K}_b\chi_a(1) \rangle \equiv \langle ab|ba \rangle = K_{ab} \tag{2.16}$$

Se obtiene la relación entre los operadores monoelectrónicos coulómbico e de intercambio, y las respectivas integrales bielectrónicas definidas al calcular el valor esperado del operador de interacción electrostática interelectrónica.

La dependencia de los operadores coulómbico y de intercambio sobre los espín-orbitales, que son a su vez soluciones de la ecuación 2.12, permite resolverla de manera iterativa.

Cada espín-orbital $\chi_k(x)$ es una solución de una ecuación de valores propios,

$$\widehat{f}\chi_k(x) = \varepsilon_k \chi_k(x) \tag{2.17}$$

Esta ecuación se resuelve mediante el método de campo autoconsistente (SCF). Un conjunto de espín-orbitales se utiliza para expandir la primera aproximación de la función de onda del sistema y, por lo tanto, el potencial efectivo de Hartree-Fock.[31]

2.2 Teoría de los Funcionales de la Densidad (DFT)

La Teoría de los Funcionales de la Densidad (DFT) es una formulación alternativa de la mecánica cuántica, propuesta por Hohenberg y Kohn [32] en 1964 y desarrollada posteriormente por Kohn y Sham [33]. Esta teoría reemplaza la función de onda de N electrones y la ecuación de Schrödinger por la densidad electrónica, lo que conduce a una reducción de la complejidad de los cálculos y un escalado aproximadamente lineal de los recursos computacionales con el tamaño del sistema.

2.2.1. Teoremas de Hohenberg y Kohn

Los teoremas de Hohenberg y Kohn establecen que la energía total de un sistema de N electrones es un funcional de la densidad electrónica $\rho(\vec{r})$. Esto significa que todas las propiedades observables del sistema se calculan a partir de esta densidad.

Los teoremas se enuncian de la siguiente manera:

- **Teorema 1:** Cualquier observable de un estado estacionario fundamental no degenerado se calcula exactamente a partir de la densidad electrónica del estado fundamental.

- **Teorema 2:** La densidad electrónica de un estado fundamental no degenerado se calcula exactamente minimizando la energía del sistema.

El teorema 1 establece que cualquier propiedad (observable) puede expresarse como un funcional de la densidad electrónica del estado fundamental. El teorema 2 proporciona el principio variacional para determinar la densidad electrónica que minimiza a la energía total. Estos teoremas sugieren que la DFT es en principio exacta, en la práctica se requieren

aproximaciones para resolverla.

2.2.2. Aproximación de Kohn y Sham

Kohn y Sham propusieron una aproximación práctica para implementar la DFT. Introdujeron el concepto de un sistema de referencia de partículas no interactuantes, construido a partir de espín-orbitales, donde la energía cinética podía calcularse de manera exacta. La contribución restante, la energía de interacción electrón-electrón, mas la diferencia en la energía cinética respecto al sistema interactuar suele ser pequeña.[34]

El sistema ficticio propuesto se conecta al sistema real al elegir un potencial v_s, de manera que la densidad de este sistema sea igual a la densidad del estado fundamental del sistema interactuante. Para minimizar la energía orbital, se introduce el operador de Kohn-Sham.

Este método establece una ecuación de tipo Hartree-Fock para resolver el sistema no interactuante, y luego introduce un funcional de intercambio-correlación E_{XC} para tomar en cuenta las interacciones electrón-electrón. La energía total se calcula como la suma de la energía cinética, la energía de Coulomb, la energía de intercambio-correlación y la energía electrón-núcleo.

La DFT proporciona una formulación alternativa a la mecánica cuántica tradicional, reemplazando la función de onda por la densidad electrónica. Aunque los teoremas de Hohenberg y Kohn sugieren que es teóricamente exacta, en la práctica se utilizan aproximaciones para resolverla, como la aproximación de Kohn y Sham, que divide el problema en uno de partículas no interactuantes más un funcional de intercambio-correlación.[34, 35]

La ecuación de Kohn-Sham original es la siguiente:

$$\left(-\frac{1}{2}\nabla^2 + V_{\text{eff}}(\mathbf{r})\right)\psi_i(\mathbf{r}) = \epsilon_i\psi_i(\mathbf{r}) \tag{2.18}$$

Donde: - ∇^2 es el operador laplaciano. - $V_{\text{eff}}(\mathbf{r})$ es el potencial efectivo de Kohn-Sham. - $\psi_i(\mathbf{r})$ son los orbitales de Kohn-Sham. - ϵ_i son las energías de los orbitales.

El potencial efectivo de Kohn-Sham se calcula como:

$$V_{\text{eff}}(\mathbf{r}) = V_{\text{ext}}(\mathbf{r}) + V_{\text{Hartree}}(\mathbf{r}) + V_{\text{XC}}(\mathbf{r}) \tag{2.19}$$

Donde: - $V_{\text{ext}}(\mathbf{r})$ es el potencial externo que proviene de las interacciones núcleo-electrón. - $V_{\text{Hartree}}(\mathbf{r})$ es el potencial de Hartree que representa la interacción electrón-electrón promediada. - $V_{\text{XC}}(\mathbf{r})$ es el potencial de intercambio-correlación que captura los efectos cuánticos y de correlación electrónica.

La densidad electrónica $\rho(\mathbf{r})$ se calcula a partir de los orbitales de Kohn-Sham:

$$\rho(\mathbf{r}) = \sum_i |\psi_i(\mathbf{r})|^2 \tag{2.20}$$

2.3 Reactividad Química

La reactividad química de una molécula se refiere a su capacidad de responder a diferentes reactivos. Para entender esta reactividad, se recurre a la estructura electrónica de la molécula aislada y se consideran los efectos que tienen los reactivos atacantes en este estado.

Una herramienta fundamental para describir la reactividad química es la teoría de los funcionales de densidad conceptual (C-DFT). Esta teoría utiliza funciones de respuesta que se expresan en términos de derivadas de la densidad electrónica y la energía total del sistema con respecto al número de electrones y al potencial externo.[36]

En el contexto de DFT, la energía total de un sistema se expresa en función de la densidad electrónica y el potencial externo. El funcional universal de Hohenberg-Kohn (F_{HK}) es clave en esta formulación y está compuesto por dos términos esenciales: la energía cinética electrónica (T) y la energía de interacción electrón-electrón (V_{ee}).

Cuando se minimiza la energía total del sistema manteniendo constante el número de electrones (N), se obtiene la ecuación de Euler-Lagrange. En esta ecuación, el potencial quí-

mico (μ) actúa como un multiplicador de Lagrange. Es importante destacar que el potencial externo desempeña un papel crucial al mantener a los electrones confinados en una región del espacio.

La relación entre el potencial químico (μ) y la electronegatividad (χ) es un pilar en la teoría de la reactividad química, siendo la primera derivada de E respecto a N La dureza química (η), es la segunda derivada de la energía total E con respecto al número de electrones N.

Para comprender la regioselectividad, se introduce la función de Fukui (f), que es la primera derivada de que $\rho(r)$) respecto al número de electrones N, es decir, cuantifica la respuesta de la densidad electrónica ante cambios en el número de electrones. Además, el descriptor dual (Δf) identifica regiones anfifilicas en las moleculas.[37]

La teoría de DFT se extiende para abarcar sistemas con números fraccionarios de electrones, lo que permite abordar procesos en los que se añaden o remueven cantidades fraccionarias de carga. Esta extensión es esencial para modelar correctamente procesos de transferencia de carga entre especies químicas.[38]

Además, varios principios de reactividad han sido enunciados o han encontrado sustento teórico en la (C-DFT) El principio de ácidos duros y blandos (HSAB) establece que los ácidos duros tienen preferencia por interactuar con bases duras, mientras que los ácidos blandos prefieren reaccionar con bases blandas. Esta regla proporciona una base para predecir interacciones ácido-base en química.[39] A partir de este marco, las reacciones químicas pueden analizarse en términos de transferencia de carga, considerando que las interacciones entre especies químicas de similar dureza son favorecidas. La energía de reacción resultante de estas interacciones es generalmente exotérmica, lo que refuerza la importancia de la regla HSAB.El principio de máxima dureza propuesto por Pearson sostiene que las moléculas tienden a organizarse de manera que sean lo más duras posible. La evolución hacia un estado de máxima dureza se fundamenta en conceptos de energía total y dureza química.[40]

Tabla 2.1: Ecuaciones clave en la teoría de reactividad química

Ecuación	Nombre	Descripción
$E[\rho] = F_{HK}[\rho] + \int dr\rho(r)v(r)$	Energía del estado fundamental	Energía total del sistema, incluyendo energía cinética y de interacción
$F_{HK}[\rho] = T[\rho] + V_{ee}[\rho]$	Funcional universal de Hohenberg-Kohn	Función que encapsula la energía cinética electrónica y la interacción electrón-electrón
$\mu = \left(\frac{\delta E}{\delta\rho(r)}\right)_v = v(r) + \frac{\delta F}{\delta\rho(r)}$	Potencial químico electrónico	Potencial que mide la disposición de un sistema químico a intercambiar electrones
$P(r,r') = \left(\frac{\delta^2 E}{\delta v(r)\delta v(r')}\right)_N$	Función de respuesta lineal	Respuesta del sistema a un cambio en el potencial externo
$f(r) = \left(\frac{\delta\mu}{\delta v(r)}\right)_N$	Función de Fukui	Cuantifica la capacidad de donación o aceptación de electrones de un sitio molecular
$\Delta f(r) = \left(\frac{\delta\eta}{\delta v(r)}\right)_N$	Descriptor dual	Indica cómo cambia la función de Fukui ante cambios en el número de electrones
$\mu = -\chi \approx -\frac{I+A}{2}$	Potencial químico	Potencial químico electrónico relacionado con la electronegatividad
$\eta \approx I - A$	Dureza química	Medida de la resistencia de un sistema a cambios en su electrón
$\omega = \frac{\mu^2}{2\eta}$	Índice de electrofilicidad	Cantidad que mide la capacidad de aceptar electrones de una especie química
$\Delta E = \mu\Delta N + \frac{1}{2}\eta(\Delta N)^2$	Ecuación de estructura de línea recta	Relación entre cambios en energía, potencial químico y dureza

2.3.1. Descriptores de Reactividad Química Local Dependientes de la Temperatura Electrónica

Fundamento

El presente enfoque, conocido como Chemical Reactivity of Atoms in Molecules (CRAIM), se centra en el análisis de la reactividad química y la estabilidad electrónica de especies químicas en su configuración de equilibrio. Para lograrlo, se considera cada cuenca atómica como un sistema abierto que es capaz de interactuar con su entorno, permitiendo el intercambio de energía y densidad electrónica.[41]

La caracterización de las cuencas atómicas se lleva a cabo mediante la Teoría Cuántica de Átomos en Moléculas (QTAIM), que define estas cuencas en función de la topología de la densidad electrónica. El número de electrones y la energía en cada cuenca se obtienen mediante la integración de la densidad electrónica sobre el volumen encerrado por la superficie de densidad.[42]

El núcleo del enfoque CRAIM se basa en el Potencial Gran de Fermiones (GPF), el cual describe el estado de equilibrio de fermiones no interactuantes en un campo medio. Derivando el GPF con respecto a sus variables naturales, se generan coeficientes de respuesta que proporcionan información esencial sobre la reactividad química local. Estos coeficientes indican cómo un átomo en una molécula reacciona ante la eliminación de restricciones del ensamble.

El enfoque CRAIM considera tanto el número de electrones como la energía en las cuencas atómicas como funciones dependientes de la densidad electrónica correspondiente y la temperatura local. Los coeficientes de reactividad derivados de CRAIM ofrecen percepciones precisas sobre propiedades químicas y estabilidad electrónica de los átomos. Además, la superficie de densidad de flujo cero actúa como una barrera restrictiva, limitando el transporte de energía y partículas hacia el entorno circundante y contribuyendo así a la estabilidad del sistema. En total, existen 10 descriptores en esta teoría, los cuales se muestran en la siguiente tabla:

Tabla 2.2: Coeficientes de Reactividad, Nombre, Descripción y Ecuación

Coeficiente	Descripción	Fórmula
Potencial Químico Local	Medida de la tendencia de los electrones a fluir en una dirección (acidez o basicidad) en el átomo A de la molécula.	$\mu_A = \left.\frac{\partial E_A}{\partial N_A}\right\|_T$
Suavidad Química Atómica	Fluctuación en el número de electrones en cada cuenca atómica, indica la capacidad de participar en una respuesta química cuando se somete a una perturbación.	$S_A = -\left.\frac{\partial N_A}{\partial \mu_A}\right\|_T$
Hiper-Suavidad Química	Indicador de la estabilidad electrónica del átomo en una molécula, proporciona información sobre su estabilidad electrónica.	$\kappa_A = \left.\frac{\partial^2 E_A}{\partial N_A^2}\right\|_T$
Entropía Electrónica	Medida de la probabilidad de que un átomo evolucione hacia un nuevo estado con mayor entropía.	$S_A = k_B \ln \Omega_A$
Temperatura Electrónica	Relacionada con las características de dureza del átomo en una molécula, indica su resistencia a perturbaciones.	$\beta_A = \frac{1}{k_B T_A}$
Indice Electrofilicidad	Índice de electronegatividad utilizado en la Teoría de la Densidad Funcional Conceptual (C-DFT).	$\omega_A = \frac{1}{2}(E_{\text{int}}^A - E_A)$
Indice Termofilicidad	Índice de termofilicidad para procesos donde se intercambia energía en lugar de electrones.	$\Lambda_\Omega = -\left.\frac{\partial E_A}{\partial T}\right\|_{N_A}$
Orbital Donador en HOMO	Variación de energía al donar carga electrónica desde el orbital más alto ocupado (HOMO).	$\epsilon_{\text{HOMO}} = \left.\frac{\partial E_A}{\partial N_i}\right\|_{V,T,N_i \to 1}$
Orbital Aceptor en LUMO	Variación de energía al aceptar carga electrónica en el orbital más bajo desocupado (LUMO).	$\epsilon_{\text{LUMO}} = \left.\frac{\partial E_A}{\partial N_i}\right\|_{V,T,N_i \to 0}$
Capacidad Calorífica Electrónica	Capacidad de un átomo o molécula para intercambiar energía con el entorno a temperatura constante.	$C_A = -T\left.\frac{\partial^2 E_A}{\partial T^2}\right\|_{N_A,V}$

Capítulo 3

Objetivos

3.1 Objetivos General

- Diseñar modelos basados en redes neuronales artificiales que permitan predecir propiedades termodinámicas del estado crítico, como acéntrico, temperatura, presión y volumen críticos, así como el factor acéntrico.

3.2 Objetivos particulares

- Construir una base de datos de propiedades criticas.
- Realizar el calculo de los descriptores de reactividad que serán utilizados como datos de entrada para los modelos de predicción.
- Realizar el entrenamiento de las diferentes Redes Neuronales Multicapa para cada propiedad termodinámica y factor acéntrico
- Optimizar cada red neuronal para establecer la mejor relación no lineal entre los parámetros de reactividad quimica locales y las propiedades críticas termodinámicas.

Capítulo 4

Metodología

En el presente trabajo construimos un modelo de predicción de propiedades termodinámicas criticas. En primera instancia realizamos una búsqueda exhaustiva de los datos de interés consultando en fuentes como NIST (Instituto Nacional de Estándares y Tecnología) por sus siglas ingles, PubChem, ChemSpider, etc, de igual forma extrajimos datos del Handbook of Thermodynamic and Physical Properties of Chemical Compounds. Con estos datos generamos una base de propiedades termodinamicas(BDPTC) que funcionaria como prueba para el modelo. El conjunto de datos de entrada fueron calculados utilizando métodos de química computacional particularmente los descriptores químicos expuestos, para así preparar una base de datos con esta base se realizaron los procesos de entrenamiento, validación y calibración en la serie de modelos expuestos en este escrito, cada paso del proceso se describirá en este capitulo.

4.1 Preparación de Datos

Se tenían 1703 moléculas con sus respectivas propiedades de interés como la Temperatura Critica, Presión critica, Volumen Critico, Densidad Critica, Factor de Comprensibilidad en su estado critico y el factor acéntrico , 10218 datos para la base. No obstante, los datos pasaron por un proceso de depuración, este proceso fue necesario ya que había que verificar datos y

eliminar otros que no estaban en las consideraciones del proyecto como los átomos donde se carecía de enlace químico, por ejemplo. Al final se tuvo un conjunto de 1512 moléculas se le asigno como Base de Datos de Propiedades Termodinámicas Criticas (BDPTC) Este conjunto de entrada se dividió en tres subconjuntos de entrenamiento, validación y prueba 70 %, 15 % y 15 %, respectivamente.

Tabla 4.1: Tamaños de los conjuntos de entrenamiento, validación y prueba.

Conjunto	**Tamaño**
Conjunto de Entrenamiento	1058 muestras
Conjunto de Validación	226 muestras
Conjunto de Prueba	226 muestras

4.2 Cálculos Química Cuántica

4.2.1. Gaussian16

La geometría de cada molécula de la BDPT se modela con ayuda del software Gaussian 16 se utiliza para realizar una optimización de la geometría molecular, lo que nos permitió disponer de la estructura molecular más estable.

Cuando se realiza una optimización de geometría con la keyword 'opt', Gaussian 16 ajustará las posiciones de los átomos de la molécula para minimizar la energía. Durante este proceso, el programa calculará iterativamente las fuerzas y gradientes moleculares, y ajustará las posiciones de los átomos hasta que se alcance un mínimo local (o un punto estacionario cercano). Una vez que se encuentra esta estructura, el programa imprimirá información sobre la geometría optimizada y la energía asociada. También se hizo un calculo de frecuencias vibracionales utilizando con la finalidad de asegurar un mínimo en la superficie de energía potencial.

keywords1 = ['opt freq','wb97xd','6-31+g*']

Para el paso de optimización y frecuencia, se uso el funcional wb97xd y el conjunto de

bases 6-311++g(d,p) que es grande y flexible, lo que lo hace adecuado para cálculos precisos de propiedades moleculares.

4.2.2. AIMALL

AIMALL proporciona la energía electrónica de los átomos y el número de electrones para cada molécula necesarios para el algoritmo de reactividad local de átomos y moléculas.

Se usaron como archivos de entrada los archivos con extensión *.wfn* proporcionados por Gaussian16, AIMALL al finalizar los cálculos, su output es una serie de archivos, la información necesaria la extrajimos de los archivos con extensión *.sum* que sirvieron de datos de entrada para correr el algoritmo de Reactividad Local en Átomos y moléculas.

4.2.3. Algoritmo de Reactividad Local en Átomos y moléculas

En el grupo de investigación del Dr Marco Franco Perez se ha desarrollado un algoritmo para calcular descriptores de reactividad química local basado en la topología de la densidad electrónica del sistema. Se utiliza la información proporcionada por AIMALL para iniciar el algoritmo y obtener los valores de cada descriptor por molecula que son la columna vertebral de este proyecto. Con estos valores se creo la BDDR (base de datos de Descriptores de Reactividad)

4.3 Construcción de Base de Datos Final

Las bases de datos BDPTC y BDDR se unificaron y se agregaron las variables de momento dipolar y volumen molecular para cada especie del conjunto de datos. Estas variables se extrajeron de los cálculos Gaussian16 y AIMALL respecticamente , en total disponemos de un conjunto de datos constituido por 40824 registros. A continuación daré un ejemplo para una molécula al azar de este conjunto final de datos. 4.2

En la base de datos final se registró para cada molécula las siguientes variables: Identificador por molécula, formula química , nombre del compuesto, masa molecular, temperatura de ebullición, propiedades termodinámicas criticas y factor acéntrico (6 variables) momento dipolar, volumen molecular y 10 descriptores de reactividad donde se tomo el valor máximo de cada uno de ellos, adicionalmente para los descriptores *Optimum Chemical Potential* y *Theta Inverse* se tomo su valor mínimo para la construcción final de la base esta constituida por 28 columnas y 1512 filas.

Tabla 4.2: Datos para una molecula en la base de datos final

ID	DJ01
FORMULA	CBrCIF2
NAME	BROMOCHLORODIFLUOROMETHANE
MW(g/mol)	165.365
TB(K)	269.141
Tc(K)	426.15
Pc(bar)	42.542
Vc(cm3/mol)	246
RHOC(g/cm3)	0.6722
Zc	0.295
OMEGA	0.187
Dipole Moment	0.6401
molecular volume	885.548485
Optimum Chemical Potential Max	-68.47
Optimum Chemical Potential Min	-3260.313
Theta Inverse Max	798.07
Theta Inverse Min	30.99
Intensive Atomic Softness Max	0.02231
Intensive Atomic Hypersoftness Max	0.00051
Intensive Heat Capacity Max	16.362
Intensive Entropy Max	4057.444
ODD at HOMO Max	181.807
OAD at HOMO Max	-51.425
ODD at LUMO Max	181.656
OAD at LUMO Max	-51.699
electrophilicity intensive MAX	29.36
thermophilicity intensive MAX	5.73

4.4 Modelos de Red Neuronal Multicapa

En esta etapa se crearon 6 redes neuronales, una para cada propiedad termodinámica de interés. Inicialmente, se pensaba que un solo modelo sería suficiente para la predicción consistente de cada una de las variables de estudio. Sin embargo, al proponer diferentes arquitecturas y capas, los resultados no fueron convincentes. Por lo tanto, se planteó realizar un análisis estadístico ampliamente usado en el área del aprendizaje automático llamado matriz de correlación y PCA (Principal Component Analysis). De esta manera, se podría determinar qué variables podrían funcionar de manera adecuada para el entrenamiento y obtener un mejor rendimiento de cada red.La matriz de correlación es una herramienta que se utiliza para evaluar la relación entre diferentes variables en un conjunto de datos.

Se utiliza la matriz de correlación para identificar las relaciones entre las propiedades termodinámicas de interés y otras variables que podrían influir en el comportamiento del modelo. Al calcular la matriz de correlación, se puede obtener una visión más clara de cómo estas variables se relacionan entre sí y determinar qué características tienen un impacto significativo en las propiedades de estudio (ver Figura 4.1).

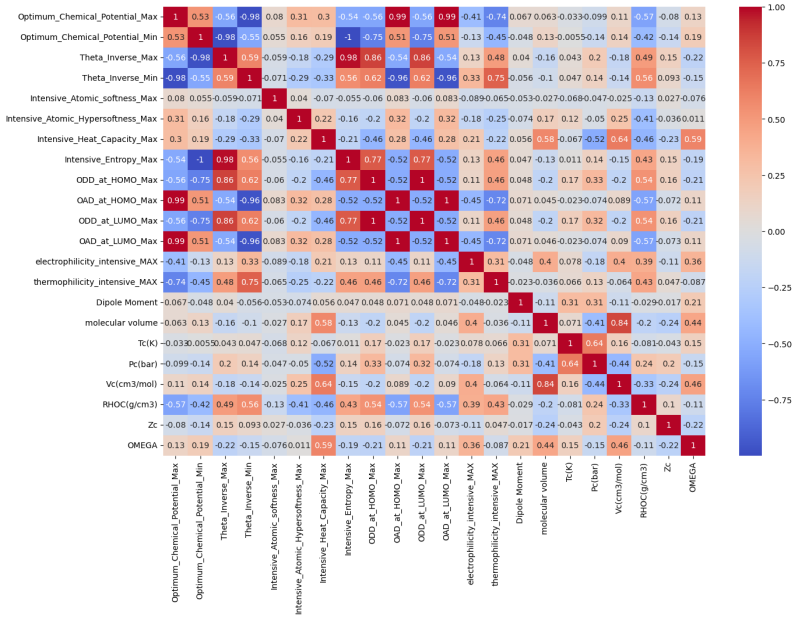

Figura 4.1: Mapa de calor de la matriz de correlación general.

En el mapa de calor con la correlación general de cada uno de los datos se puede notar que era necesario hacer un análisis exhaustivo que considerara cada variable de interés para poder determinar para cada propiedad termodinámica que variables son mas afines y como ejemplo se toma la temperatura critica para la descripción metodológica realizada, para dicha variable se tienen los siguientes valores.

Tabla 4.3: Matriz de Correlación para Temperatura Critica

Variable	Correlación con Tc(K)
Tc(K)	1.000000
Pc(bar)	0.641931
Dipole Moment	0.306129
ODD_at_LUMO_Max	0.171877
ODD_at_HOMO_Max	0.171777
Vc(cm3/mol)	0.158645
OMEGA	0.152125
Intensive_Atomic_Hypersoftness_Max	0.117585
electrophilicity_intensive_MAX	0.078300
molecular volume	0.071245
thermophilicity_intensive_MAX	0.065889
Theta_Inverse_Min	0.047433
Theta_Inverse_Max	0.042701
Intensive_Entropy_Max	0.011135
Optimum_Chemical_Potential_Min	-0.005456
OAD_at_LUMO_Max	-0.022764
OAD_at_HOMO_Max	-0.022771
Optimum_Chemical_Potential_Max	-0.032951
Zc	-0.042887
Intensive_Heat_Capacity_Max	-0.067131
Intensive_Atomic_softness_Max	-0.068037
RHOC(g/cm3)	-0.081361

Se considera solo las variables con correlación positiva en la matriz ya que es importante para simplificar el modelo, eliminar información redundante, evitar problemas de multicolinealidad y mejorar la eficiencia y la interpretabilidad del modelo. Es pertinente recordar que la multicolinealidad es un fenómeno en el que dos o más variables están altamente correlacionadas entre sí. Esto puede causar problemas en modelos de regresión, ya que la presencia de variables altamente correlacionadas hace que las estimaciones de los coeficientes de re-

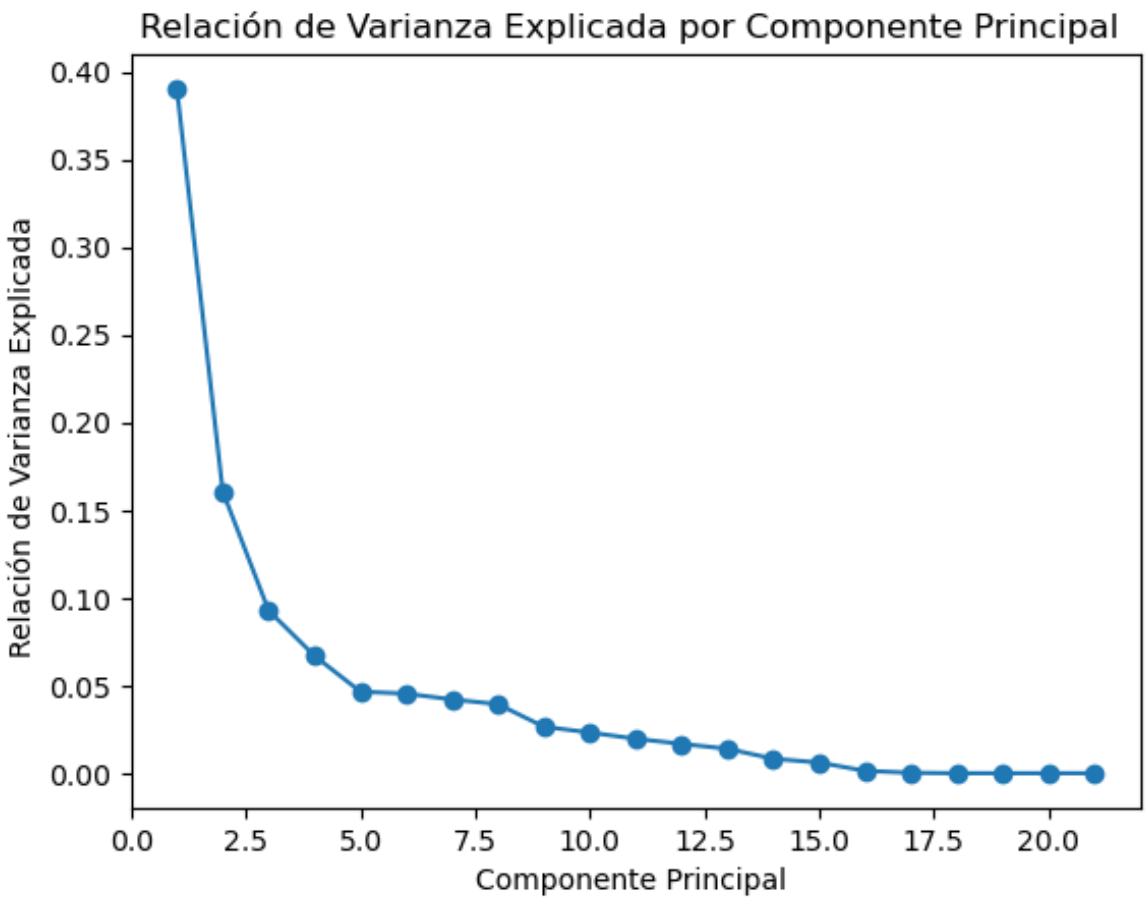

Figura 4.2: PCA

gresión sean inestables.

El PCA ayuda a simplificar y resumir la información de alta dimensionalidad en un número menor de dimensiones, lo que facilita el análisis y la interpretación de los datos.

En el contexto de estudio, se aplica el PCA a las variables del conjunto de datos para identificar las variables más importantes y descartar aquellas que tienen menos impacto en las propiedades termodinámicas de interés. Al utilizar PCA, se representan las variables originales en un espacio nuevo, definido por los componentes principales, donde cada componente representa una combinación lineal de las variables originales.

La Figura 4.2 muestra la *Relación de Varianza Explicada* para cada uno de los Componentes Principales resultantes del análisis. El eje x representa el número de Componentes Principales y el eje y representa la Varianza Explicada por cada uno de ellos.

La Varianza Explicada indica cuánta información o variabilidad de los datos originales es capturada por cada Componente Principal. El primer componente principal captura la mayor cantidad de varianza, y a medida que avanza hacia los siguientes componentes principales, la cantidad de varianza explicada disminuye progresivamente.

El número de Componentes Principales seleccionado se basa en el nivel de varianza que se considera relevante para el análisis. En este caso, se ha seleccionado un umbral del 98 % de varianza explicada, por lo tanto conservamos aquellos componentes que capturan al menos el 98 % de la variabilidad total de los datos originales.

El análisis de componentes principales permite reducir la dimensionalidad de los datos al transformar las variables originales en un espacio de menor dimensión definido por los componentes principales. Al igual que la matriz de correlación la meta es seleccionar las variables más relevantes para construir modelos de red neuronal más eficientes y precisos.

Tabla 4.4: Principales variables para el primer componente principal de Tc(K)

Principales variables para PCA
Theta_Inverse_Min
Theta_Inverse_Max
ODD_at_HOMO_Max
ODD_at_LUMO_Max
Intensive_Entropy_Max
thermophilicity_intensive_MAX
electrophilicity_intensive_MAX
Dipole Moment
Intensive_Atomic_softness_Max

Para tomar la decisión de que variables son las idóneas para el modelo de red se crea una tabla que compare las variables seleccionadas por PCA y las variables con valores positivos en la matriz de correlación:

Tabla 4.5: Comparación de Variables seleccionadas por PCA y Variables con Correlación Positiva

Variables PCA	**Variables con Correlación Positiva**
Theta_Inverse_Min	Tc(K)
Theta_Inverse_Max	Pc(bar)
ODD_at_HOMO_Max	Dipole Moment
ODD_at_LUMO_Max	ODD_at_LUMO_Max
Intensive_Entropy_Max	ODD_at_HOMO_Max
thermophilicity_intensive_MAX	Vc(cm3/mol)
electrophilicity_intensive_MAX	OMEGA
Dipole Moment	Intensive_Atomic_Hypersoftness_Max
Intensive_Atomic_softness_Max	electrophilicity_intensive_MAX
	molecular volume
	thermophilicity_intensive_MAX

En esta tabla, Se observa que al menos 8 variables eran indispensables para poder ejecutar un modelo de red aceptable, ya que se repitieron en ambas técnicas de reducción de variables, se tomaron las variables por el método PCA y por el método de Matriz de Correlación que se consideraron pertinentes como datos de entrada de la red particularmente 12 variables mostradas en la tabla 4.5

4.5 Descripción de la Red Neuronal

En esta sección, se presenta la arquitectura y entrenamiento de la red neuronal utilizada para propiedad temperatura critica cabe mencionar que el proceso descrito en las secciones anteriores se aplico para cada una de las propiedades criticas de estudio. La red fue diseñada con el objetivo de resolver un problema multivariable no lineal, donde se busca predecir el valor de la temperatura crítica ($Tc(K)$) a partir de 12 variables de entrada. A continuación se describe todo el proceso de funcionamiento de esta red utilizando vocabulario y términos de ciencia de la computación y lenguaje de programación Python.[43]

La arquitectura de la red neuronal se definió mediante la clase `Net`, que se hereda de la clase `nn.Module`, ambas clases provienen de la librería especializada de inteligencia artificial *PyTorch*.

La red consta de tres capas completamente conectadas (*fully connected layers*) y dos capas de activación ReLU. La primera capa (`fc1`) tiene 12 neuronas de entrada y 12 neuronas de salida, seguida de la segunda capa (`fc2`) con 6 neuronas de entrada y 6 neuronas de salida. Finalmente, la tercera capa (`fc3`) tiene 3 neuronas de entrada y 1 neurona de salida, que representa la predicción final de la red. 4.3

Durante el entrenamiento de la red, se utilizo la función de pérdida de error cuadrático medio (`MSELoss`) para calcular la diferencia entre las predicciones de la red y las etiquetas reales. Los parámetros de la red se ajustan utilizando el optimizador Adam con una tasa de aprendizaje de 0.001.

El entrenamiento se llevo a cabo a través de un bucle de 1000 épocas, donde se ajustaron gradualmente los parámetros de la red para minimizar la pérdida. Después de cada época, se realiza una validación y prueba de la red en los conjuntos de datos de validación y prueba, respectivamente.

Los resultados del entrenamiento se registraron en las listas `train_losses`, `val_losses` y `test_losses`, donde se almacenan las pérdidas de entrenamiento, validación y prueba, respectivamente. Estas pérdidas se utilizan para evaluar el rendimiento de la red y determinar su capacidad predictiva.[43]

La utilización de PCA así como de la matriz de correlación permitió identificar las variables más relevantes y realizar una reducción de dimensionalidad para mejorar la capacidad predictiva de la red neuronal. Es importante destacar que todo este proceso es artesanal ya que implica ajustar cada uno de los hiperparámetros conforme se ejecuta el modelo. Para ello, se realizaron múltiples ajustes en la arquitectura de la red neuronal, las variables de entrada y sus hiperparámetros, como la Tasa de aprendizaje (*Learning rate*), Épocas (*Epochs*), Tamaño de lote (*Batch size*), Unidades ocultas (*Hidden units*), Función de activación (*Activation function*), Regularización (*Regularization*), Optimizador (*Optimizer*), y Función de pérdida (*Loss function*), así como el Conjunto de validación (*Validation set*).

El proceso de forward propagation consiste en propagar los datos de entrada a través de la red neuronal para generar predicciones. Todos los modelos de red neuronal implementados

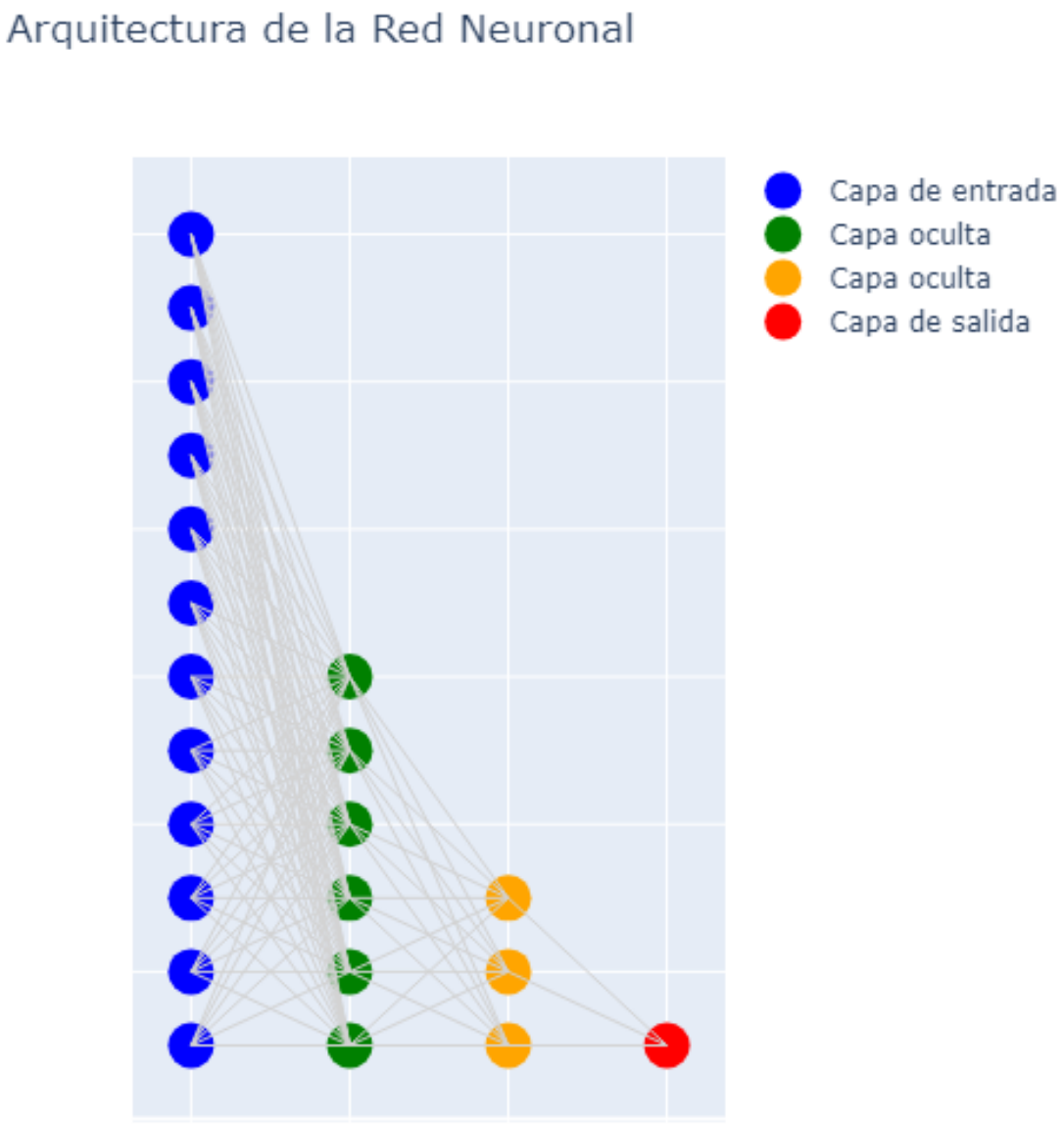

Figura 4.3: Arquitectura de Red Neuronal para Temperatura Critica, donde cada color muestra cada capa y sus nodos

en este trabajo realizan forward propagation durante el proceso de predicción. En cada capa de la red, las entradas se combinan con los pesos de las conexiones para calcular los valores de activación, que se utilizan como entradas para la siguiente capa. Este proceso se repite hasta llegar a la capa de salida, donde se obtienen las predicciones finales.

En la siguiente tabla se ilustran los detalles de cada una de las redes que se lograron en este proyecto.4.6

4.6 Detalles por Modelo de Red Neuronal Artificial

Tabla 4.6: Detalle de cada red

Nombre de red	**Arquitectura**	**Tasa de aprendizaje**	**Función de activación**
RED-Tc	(12, 6, 3, 1)	0.001	ReLU
RED-Pc	(10, 15, 5, 1)	0.0001	ReLU
RED-Vc	(8, 12, 4, 1)	0.001	ReLU
RED-Dc	(8, 4, 8, 1)	0.01	ReLU
RED-Fa	(9, 9, 9, 1)	0.001	ReLU
RED-Zc	(6, 9, 6, 1)	0.001	ReLU

Tabla 4.7: Detalle de cada red, continuación

Nombre de red	**Épocas**	**Tamaño de lote**	**Optimizador**
RED-Tc	1500	256	Adam
RED-Pc	1500	128	Adam
RED-Vc	1200	32	Adam
RED-Dc	1000	64	SDG
RED-OMEGA	1500	128	Adam
RED-Zc	500	64	Adam

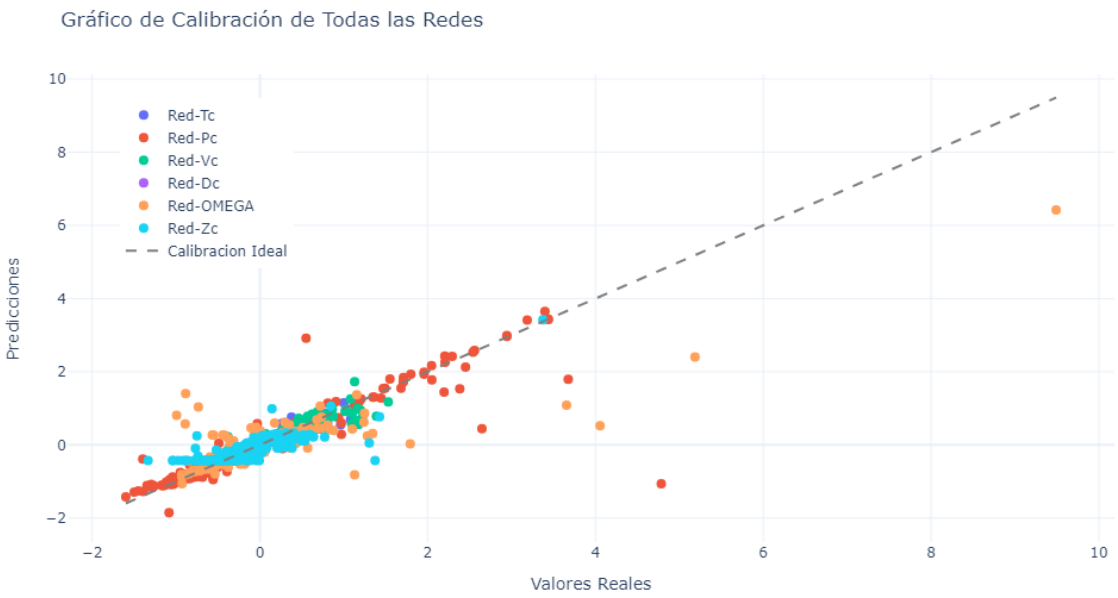

Figura 4.4: Calibración de Redes Neuronales, Predicciones vs Valores Reales

Capítulo 5

Resultados y discusión

5.1 Análisis de Resultados

En esta sección, se presentarán y analizarán los resultados obtenidos de los seis modelos desarrollados para predecir las propiedades termodinámicas críticas. Se iniciará destacando las similitudes comunes entre cada una de las redes neuronales utilizadas. Posteriormente, se profundizará en los resultados específicos obtenidos por cada modelo y se discutirán las métricas de rendimiento.

5.1.1. Características Comunes de los Modelos

Antes de analizar los resultados específicos, es relevante destacar las características comunes compartidas por todos los modelos de red neuronal desarrollados. Estas características incluyen:

- **Función de Activación ReLU**: En todos los modelos, se utilizó la función de activación ReLU (Rectified Linear Unit) en las capas ocultas. Esta función permite la activación de neuronas solo si la entrada es positiva, lo que ayuda a superar el problema de desvanecimiento de gradientes y agiliza el proceso de entrenamiento.

- **Función de Optimización Adam**: Para el proceso de optimización durante el entrenamiento, se empleó el algoritmo Adam (Adaptive Moment Estimation). Esta técnica combina la adaptabilidad de los métodos RMSprop y el momento de Nesterov, lo que resulta en un rendimiento eficiente y rápido.

Estas decisiones de diseño permitieron una implementación eficiente y una convergencia exitosa durante el proceso de entrenamiento en todos los modelos. También hay que destacar que el tamaño de los conjuntos de entrenamiento, validación y prueba son homogéneos en cada modelo de red con 70 %, 15 % y 15 % respectivamente Algo interesante es analizar la frecuencia de las variables de entrada para la construcción de cada modelo, en la metodología que se ocuparon las técnicas de PCA y la de matriz de correlación para elegir cada variable por modelo. En la figura 5.1 se muestra un gráfico de barras que nos indica la frecuencia de variable de entrada por modelo 4.7.

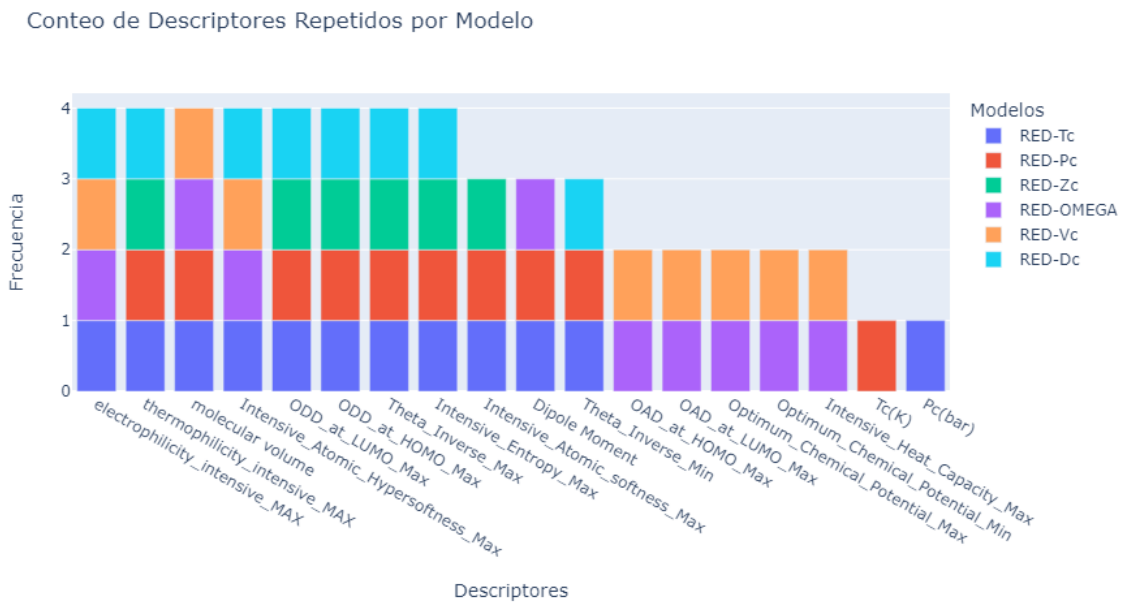

Figura 5.1: Conteo de Descriptores Repetidos por Modelo

Hay 8 variables que se repiten en 4 modelos de los 6 realizados y 7 variables que se repiten en los otros dos, es decir tenemos que los descriptores para el factor acéntrico (RED-OMEGA) y para el volumen critico (RED-Vc) están intrínsecamente relacionadas se hace la inferencia lógica de que hay descriptores mas afines a determinada propiedad. En la figura

5.2 hay una visualización de las conexiones entre las variables por modelo donde los nodos azules representan las variables y los nodos verdes las redes neuronales. Este diagrama nos permite dilucidar la relación de variable de entrada con el modelo, en la 5.1 tenemos el conteo general de la frecuencia de variables de entrada con los respectivas variables por modelo.

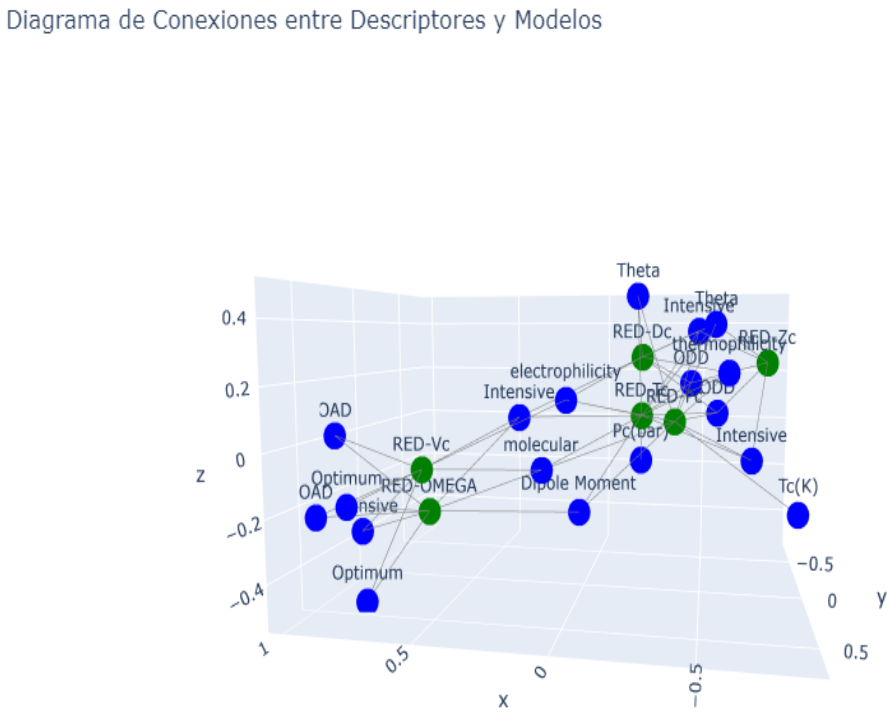

Figura 5.2: Diagrama de conexiones entre Variables de entrada y Modelos de Red

El grafo de correlación permite visualizar como las variables de entrada se distribuyen en cada modelo, donde se visualiza las conexiones que existen entre los modelos para analizar una posible creación de modelo único.

Tabla 5.1: Conteo de Frecuencia de variables de entrada por modelo

Descriptor	**Freq.**	**Model**
Intensive_Entropy_Max	4	RED-Tc, RED-Pc, RED-Zc, RED-Dc
Theta_Inverse_Max	4	RED-Tc, RED-Pc, RED-Zc, RED-Dc
ODD_at_HOMO_Max	4	RED-Tc, RED-Pc, RED-Zc, RED-Dc
ODD_at_LUMO_Max	4	RED-Tc, RED-Pc, RED-Zc, RED-Dc
Intensive_Atomic_Hypersoftness_Max	4	RED-Tc, RED-Dc, RED-Vc, RED-OMEGA
molecular_volume	4	RED-Tc, RED-Pc, RED-Vc, RED-OMEGA
thermophilicity_intensive_MAX	4	RED-Tc, RED-Pc, RED-Zc, RED-Dc
electrophilicity_intensive_MAX	4	RED-Tc, RED-Pc, RED-Vc, RED-Dc
Intensive_Atomic_softness_Max	3	RED-Tc, RED-Pc, RED-Zc
Dipole Moment	3	RED-Tc, RED-Pc, RED-OMEGA
Theta_Inverse_Min	3	RED-Tc, RED-Pc, RED-Dc
Intensive_Heat_Capacity_Max	2	RED-OMEGA, RED-Vc
Optimum_Chemical_Potential_Min	2	RED-OMEGA, RED-Vc
Optimum_Chemical_Potential_Max	2	RED-OMEGA, RED-Vc
OAD_at_LUMO_Max	2	RED-OMEGA, RED-Vc
OAD_at_HOMO_Max	2	RED-OMEGA, RED-Vc
Pc(bar)	1	RED-Tc
Tc(K)	1	RED-Pc

5.1.2. Resultados Específicos

Los resultados específicos obtenidos por cada uno de los seis modelos para predecir las propiedades termodinámicas críticas. Se detallan las métricas de rendimiento evaluadas en el conjunto de prueba y se incluyen gráficos para visualizar la calidad de las predicciones.

Modelo RED-Zc: Propiedad factor de comprensibilidad crítico

La Red utiliza la función de activación ReLU en todas las capas ocultas. La red tiene una capa de salida con una sola neurona que predice el valor del factor de comprensibilidad crítico (Zc) basado en las características de entrada. En la siguiente tabla se observan las métricas de rendimiento.

Tabla 5.2: Métricas de rendimiento del modelo.

Métrica	Valor
MSE (Error Cuadrático Medio)	0.0010
RMSE (Raíz del Error Cuadrático Medio)	0.0313
MAE (Error Absoluto Medio)	0.0184

- **Arquitectura del Modelo RED-Zc**: La arquitectura de la red neuronal está compuesta por tres capas ocultas. La primer capa oculta tiene 6 neuronas, la segunda tiene 9 neuronas y la tercera capa tiene 6 neuronas.

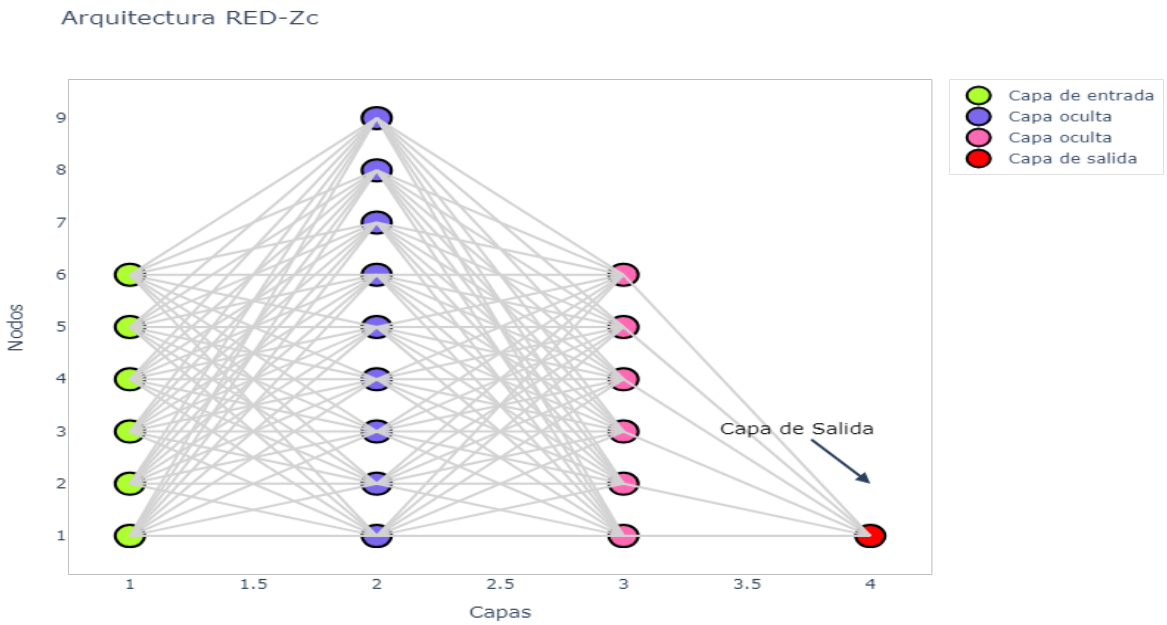

Figura 5.3: Arquitectura RED-Zc

- **Curvas de Pérdida en Entrenamiento, Validación y Prueba**(Figura 5.4): Se muestra la función de pérdida durante el entrenamiento, validación y prueba del modelo RED-Zc. Se observa cómo la pérdida disminuye a medida que el modelo se entrena, y cómo se comporta en los conjuntos de validación y prueba.

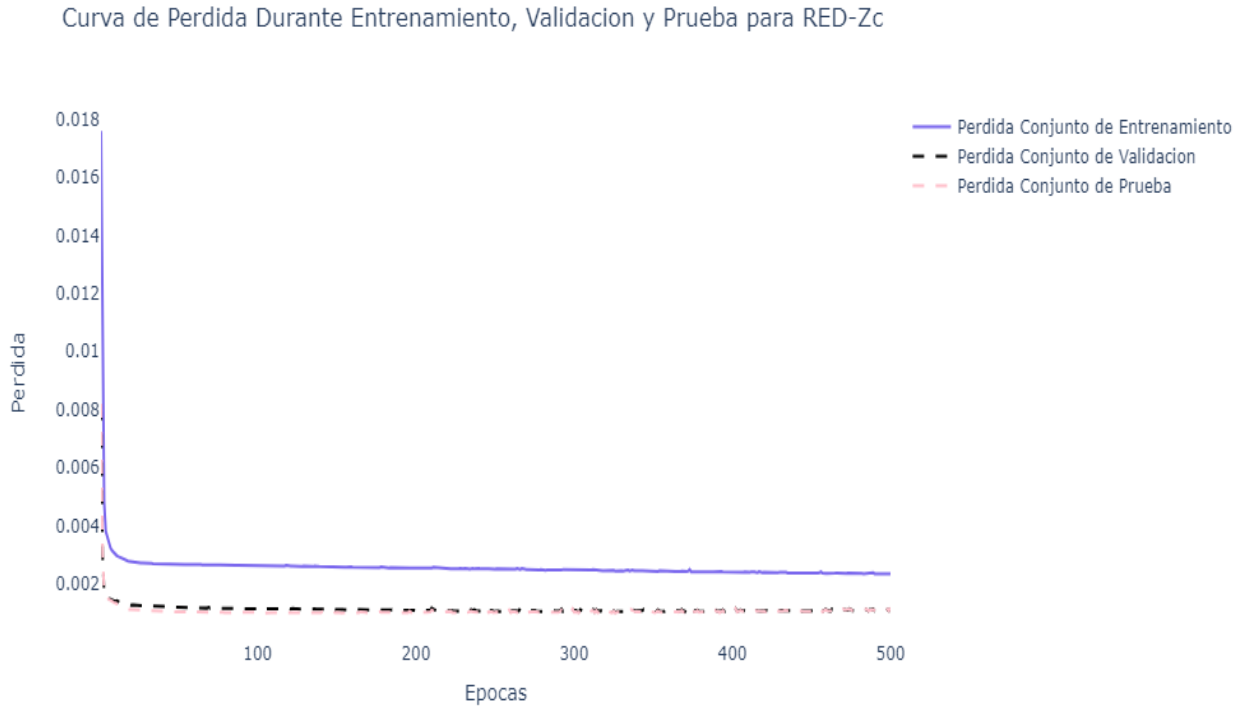

Figura 5.4: Función de Pérdida durante Entrenamiento, Validación y Prueba RED-Zc

- **Gráfica de Calibración RED-Zc**(Figura 5.5): La línea de puntos representa la calibración ideal, donde las predicciones del modelo deberían estar alineadas con la recta, el modelo presenta subestimación ya que todas las predicciones se aglomeran en la parte inferior de la recta.

- **Gráfica de Residuales Análisis**(Figura5.6): Una distribución aleatoria de los residuales en torno al valor cero indica que el modelo está realizando buenas predicciones.

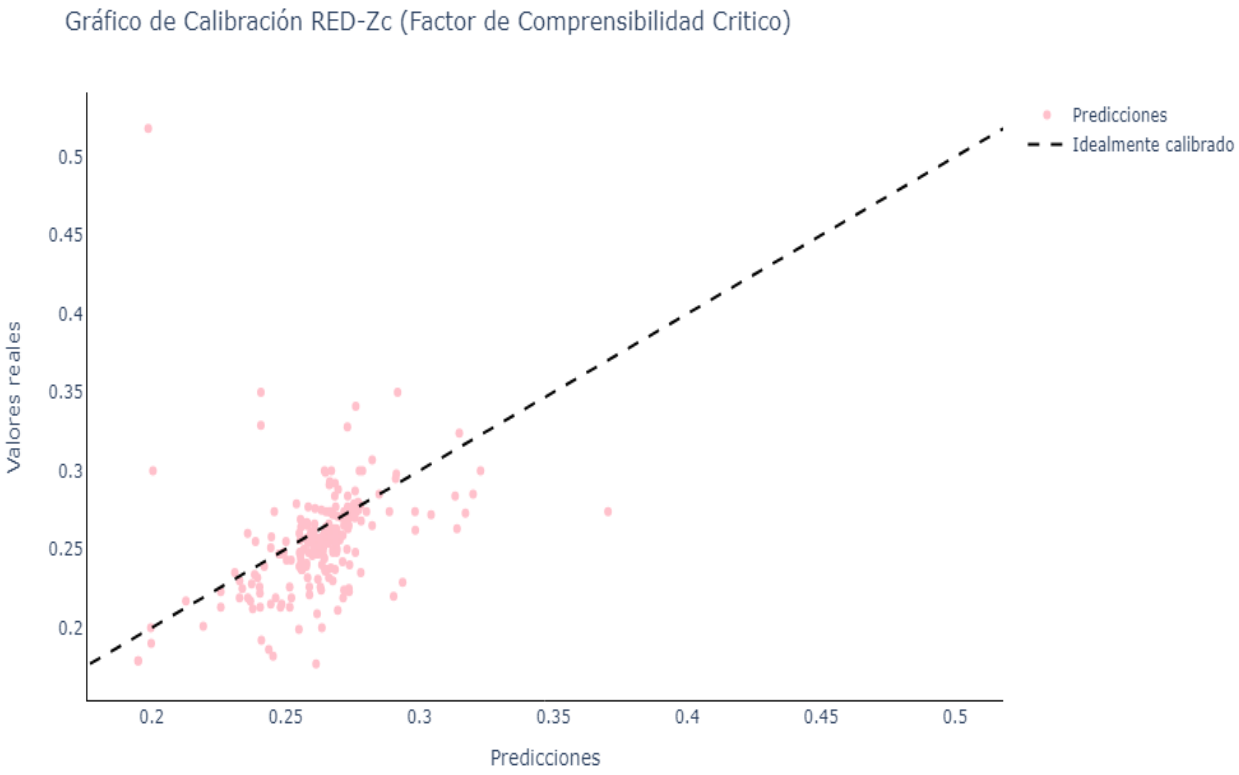

Figura 5.5: Gráfica de Calibración RED-Zc

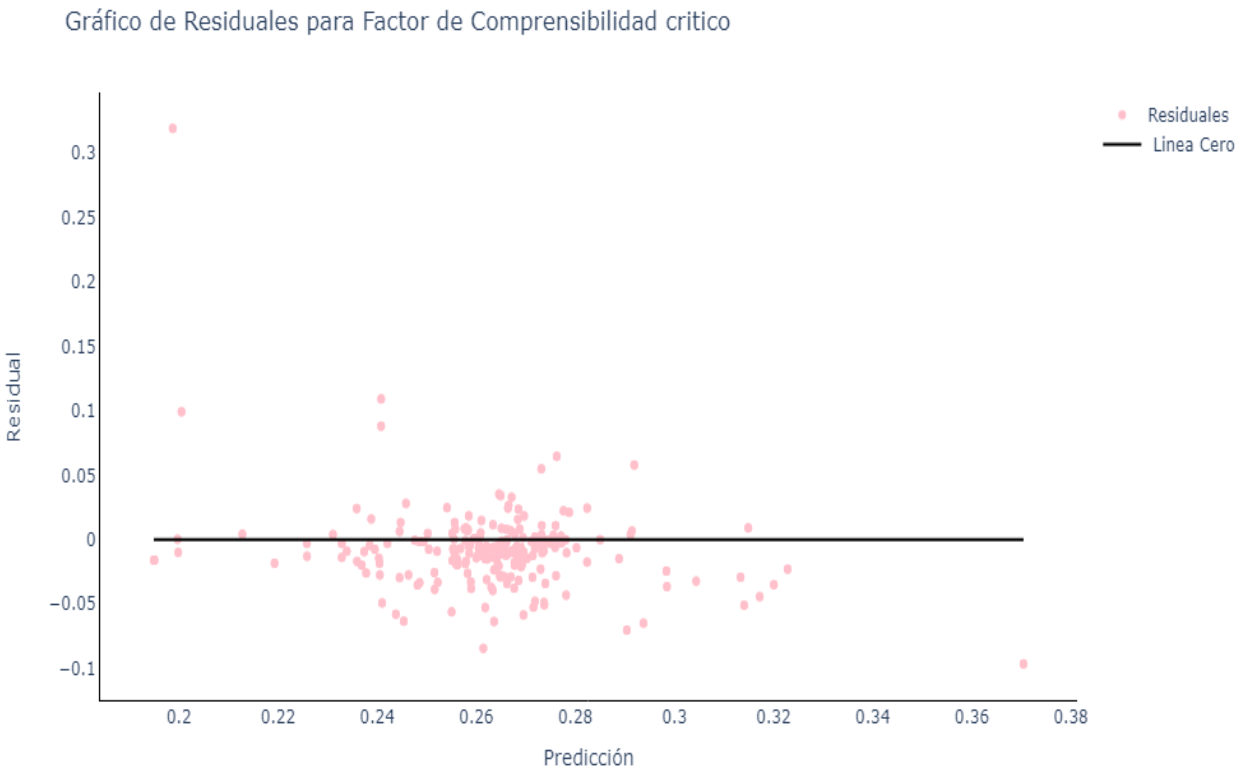

Figura 5.6: Gráfica de Residuales RED-Zc

- **Gráfica Valores Reales y Valores Predecidos en una muestra Aleatoria del conjunto de datos para Prueba de Eficiencia del Modelo RED-Zc**(Figura5.7): Se muestra una gráfica que compara los valores reales y los valores predecidos por el modelo RED-Zc para una muestra aleatoria del conjunto de prueba.

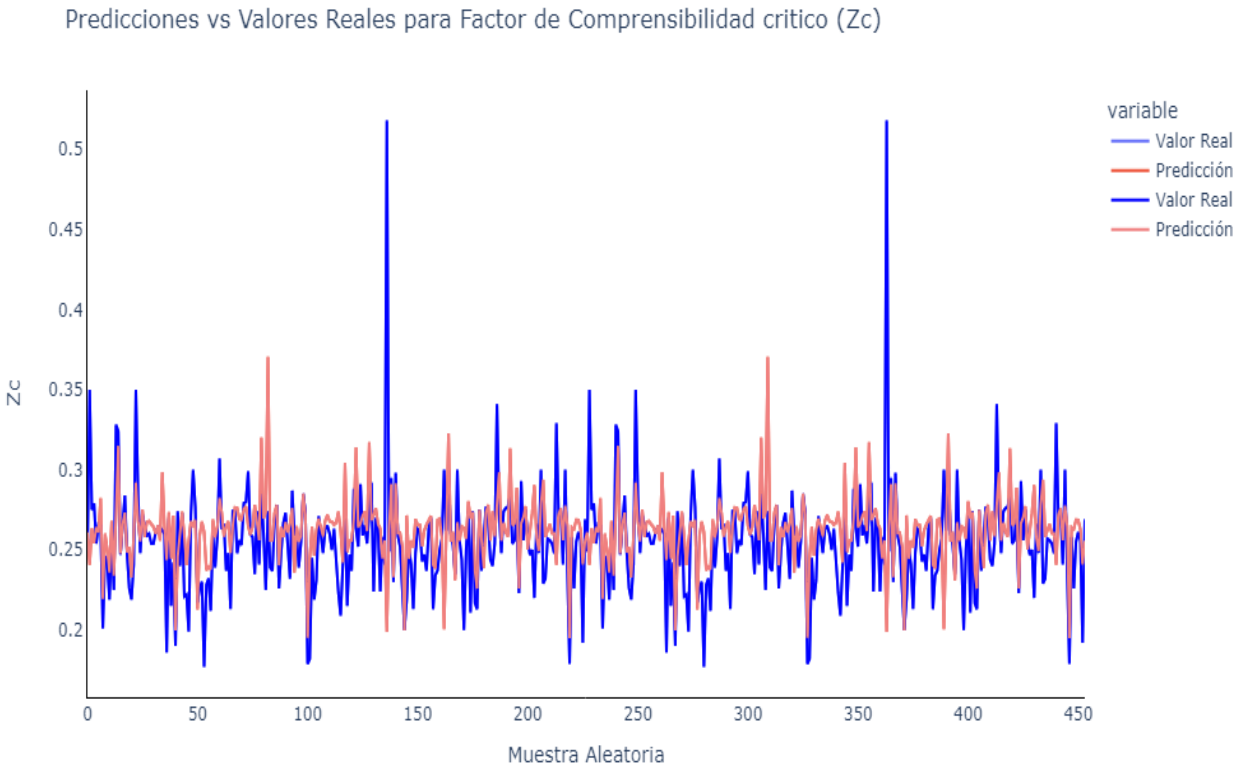

Figura 5.7: Valores Reales vs Predicciones conjunto de prueba RED-Zc

- **Resultados de la predicción y la incertidumbre en el conjunto de prueba para Zc**(Tabla 5.3): Se presenta una tabla con los resultados de las predicciones del modelo RED-Zc en el conjunto de prueba. Se incluye el valor real de Zc, la predicción del modelo, la diferencia entre ambos y el porcentaje de incertidumbre. La incertidumbre se calcula como el porcentaje de diferencia entre el valor real y la predicción con respecto al valor real.

La incertidumbre general de la RED-Zc 7.92 % en general es una red que funciona adecuadamente a pesar de subestimación en la gráfica de calibración y en los residuales.

Tabla 5.3: Resultados de la predicción y la incertidumbre en el conjunto de prueba para Zc.

Sustancia Química	Valor Real Zc	Predicción Zc	Diferencia	Incertidumbre (%)
ISOPENTIL ISOVALERATO	0.249	0.26099154	0.011991546	4.82 %
m-TERFENIL	0.35	0.2408135	0.1091865	31.20 %
CLORURO DE ETILO	0.275	0.26338205	0.011617959	4.22 %
TRICLOROFLUOROMETANO	0.279	0.2540852	0.024914801	8.93 %
2-ETILNAFTALENO	0.254	0.26406512	0.010065109	3.96 %
SULFURO DE DIMETILO	0.266	0.26088852	0.005111486	1.92 %
1,2-DIBROMOETANO	0.265	0.28235832	0.017358333	6.55 %
ÁCIDO ACÉTICO	0.201	0.21929106	0.018291056	9.10 %
1-UNDECENO	0.246	0.26001146	0.014011458	5.70 %
2,3-DIMETILOCTANO	0.243	0.2504068	0.007406801	3.05 %
SULFURO DE BUTILO-NONILO	0.219	0.2359412	0.016941205	7.74 %
ACENAPHTHENE	0.257	0.2678943	0.010894299	4.24 %
SULFURO DE BUTILO-OCTILO	0.225	0.23386577	0.008865774	3.94 %
DI-n-PROPILEMINA	0.328	0.27312422	0.05487579	16.73 %
TRIBROMOSILANO	0.324	0.3148937	0.009106308	2.81 %
N-BUTILCICLOHEXANO	0.247	0.24759781	0.00059782	0.24 %
SULFURO DE DIMETILO	0.266	0.2662159	0.000215888	0.08 %
ETANO	0.284	0.27322254	0.010777473	3.79 %
N-BUTILCICLOHEXANO	0.247	0.24851727	0.001517281	0.61 %

Modelo RED-Vc: Propiedad Volumen Critica

La RED-Vc es una red neuronal de regresión Utiliza la función de activación ReLU en las capas ocultas para introducir no linealidades y mejorar el aprendizaje de relaciones complejas entre descriptores químicos y la propiedad Vc (volumen crítico). El optimizador Adam y la función de pérdida MSE se utilizan durante el entrenamiento de 1200 épocas. Se escalaron los datos para tener media cero y varianza unitaria, y el modelo se evalúo en conjuntos de validación y prueba para verificar su rendimiento.

- **Arquitectura del Modelo**(Figura 5.8): La arquitectura de la red neuronal La RED-Vc cuenta con 8 neuronas de entrada, seguida de dos capas ocultas con 12 y 4 neuronas respectivamente, y una capa de salida con 1 neurona.

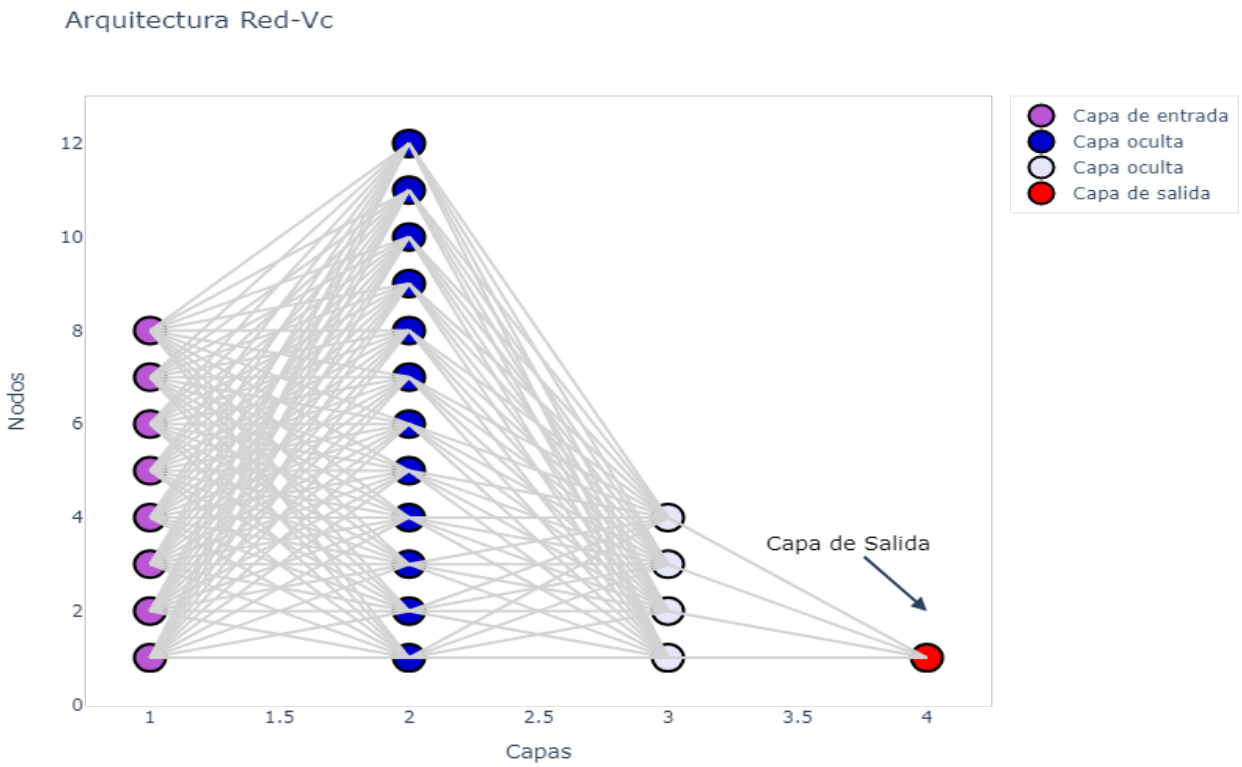

Figura 5.8: Arquitectura RED-Vc

La Tabla 5.4 muestra las métricas de rendimiento del modelo RED-Vc en el conjunto de prueba. El Error Cuadrático Medio (MSE) de 0.2398 indica la magnitud promedio de los errores al cuadrado, mientras que el RMSE de 0.4897 mide la dispersión de los errores. El MAE de 0.1704 representa la magnitud promedio de los errores absolutos. El modelo tiene un buen ajuste global con un R^2 de 0.8058, reflejando su capacidad para predecir el volumen crítico de manera precisa.

Tabla 5.4: Métricas de Rendimiento en el Conjunto de Prueba para RED-Vc

MSE (Error Cuadrático Medio)	0.2398
RMSE (Raíz del Error Cuadrático Medio)	0.4897
MAE (Error Absoluto Medio)	0.1704
R^2 (Coeficiente de Determinación)	0.8058

- **Curvas de Pérdida en Entrenamiento, Validación y Prueba**(Figura 5.9) Aunque los perfiles no se empalman puede observar que las 3 curvas tienden convergencia después de las 1000 épocas, lo que nos indica un sobre ajuste mínimo en nuestros datos.

- **Gráfica de Calibración RED-Vc**(Figura 5.10) En la RED-Vc se observa que el gráfico de calibración encaja bien en la línea de idealidad, significa que las predicciones del modelo están calibradas y son cercanas a los valores reales en promedio. Esto indica que el modelo está produciendo estimaciones precisas y confiables para la propiedad de interés

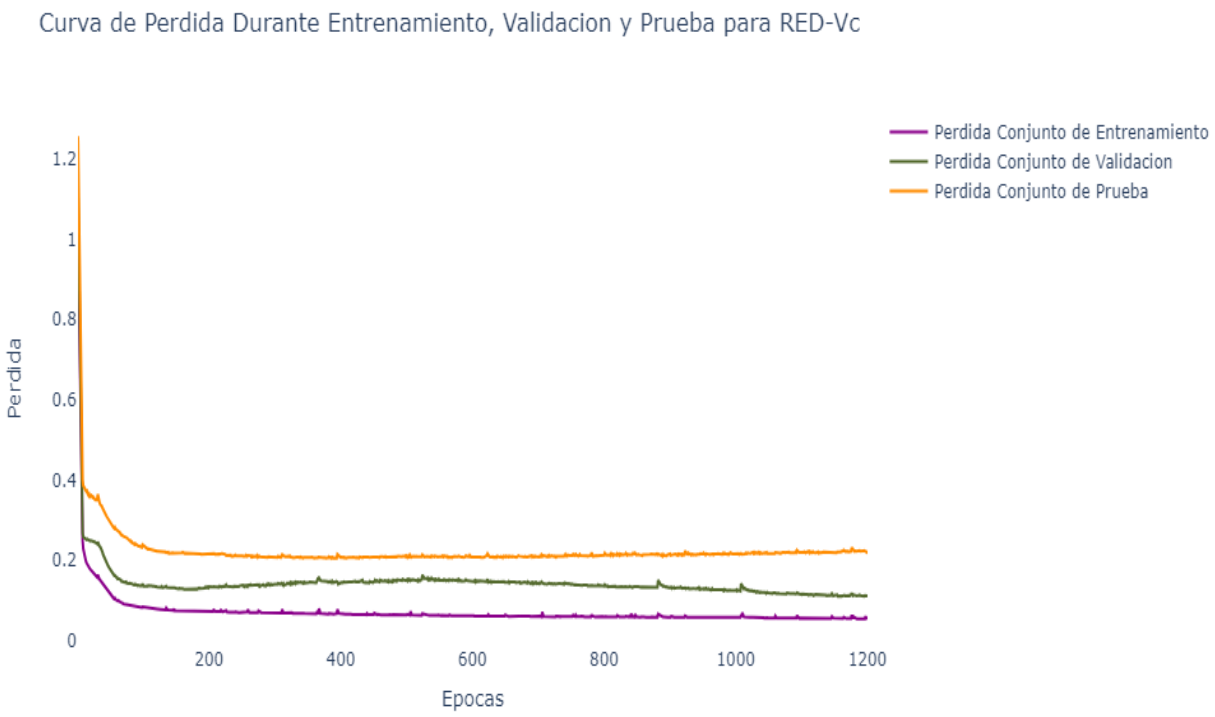

Figura 5.9: Función de perdida con Épocas RED-Vc

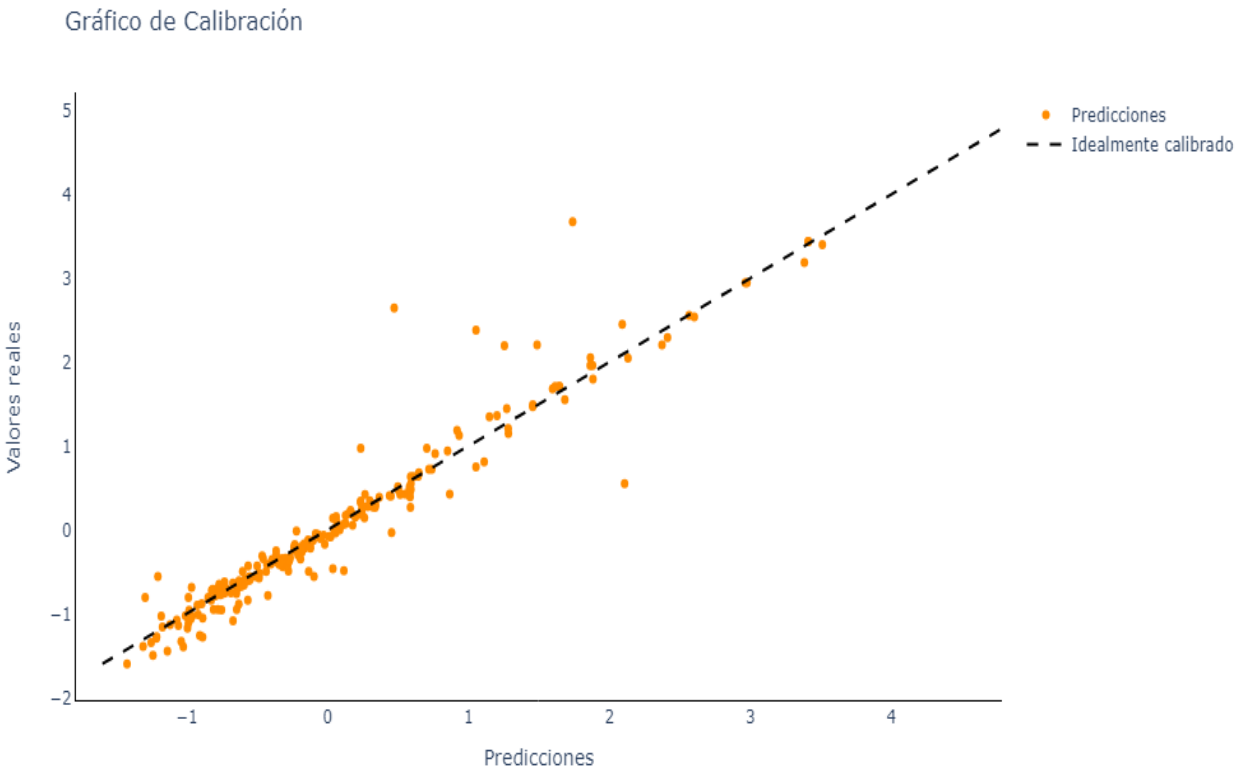

Figura 5.10: Gráfica de Calibración RED-Vc

- **Gráfica de Residuales Análisis**(Figura 5.11) la gráfica de residuales distribuye de manera homogénea los puntos de diferencia entre los valores reales y las predicciones sobre la linea cero esto indica que el modelo hace buenas predicciones.

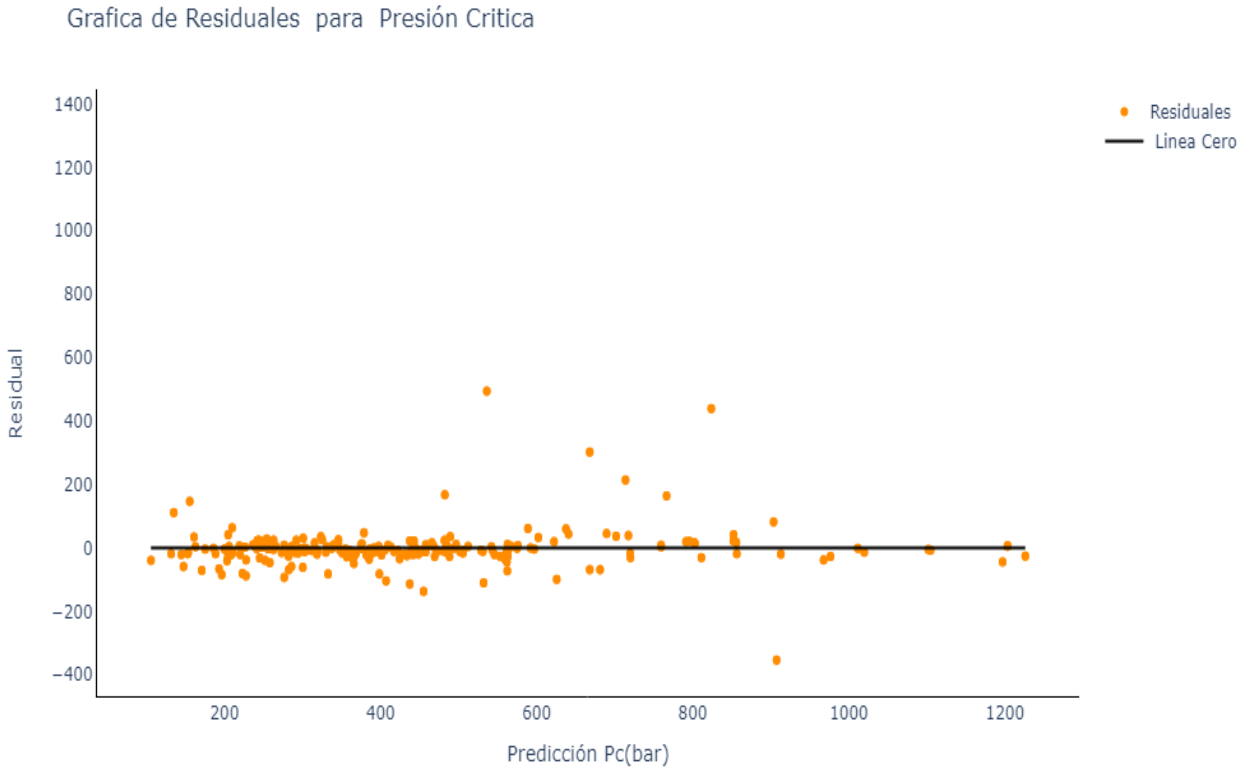

Figura 5.11: Grafica de Residuales RED-Vc

- **Grafica Valores Reales y Valores Predecidos en una muestra Aleatoria del conjunto de datos para Prueba de Eficiencia del Modelo RED-Vc** (Figura 5.12) El gráfico muestra como las predicciones del modelo presentan un comportamiento similar a los valores la muestra.

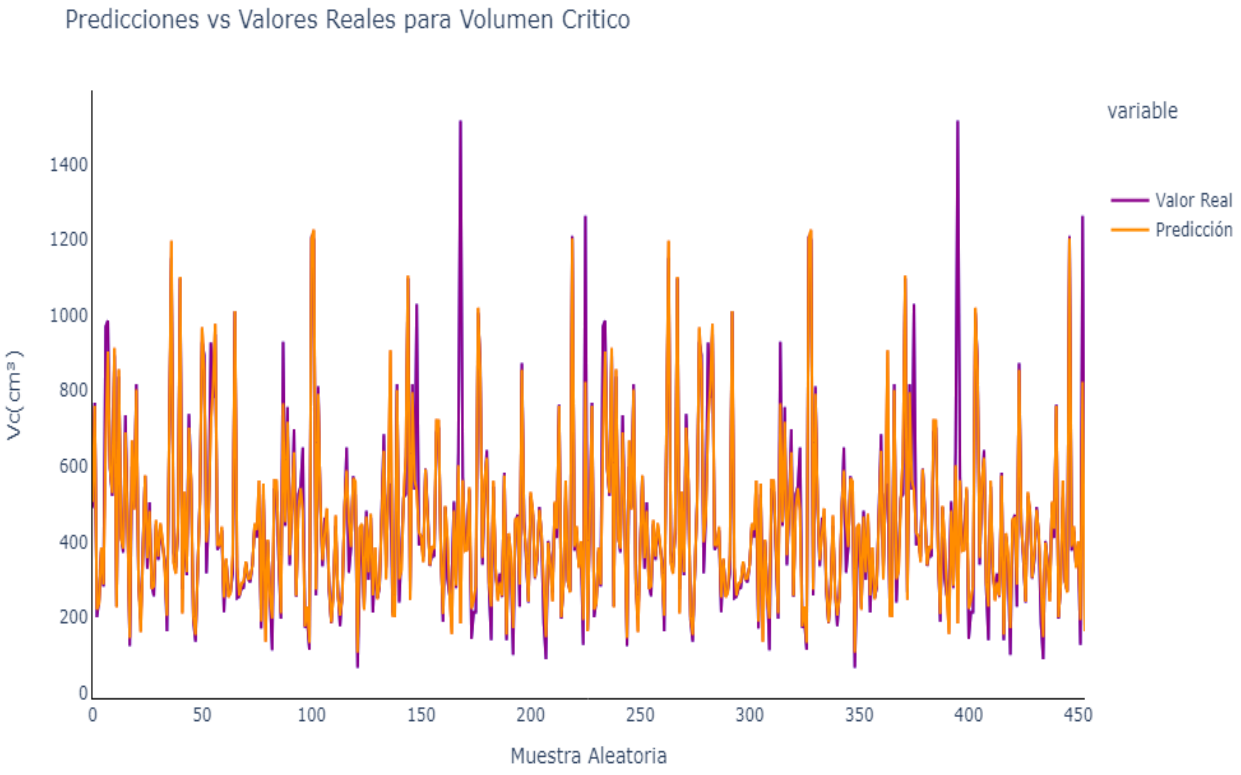

Figura 5.12: Grafica Valores Reales y Valores Predecidos en una muestra Aleatoria del conjunto de datos para Prueba de Eficiencia del Modelo RED-Vc

La incertidumbre general del modelo RED-Vc es de 9.55 %. Lo que indica que la red realiza predicciones con un alto nivel de confianza.

Tabla 5.5: Resultados de la predicción y la incertidumbre en el conjunto de prueba para Vc (cm^3/mol).

Molecula	**Valor Real Vc**	**Predicción Vc**	**Incertimdumbre (%)**
ÓXIDO DE CARBONILO	135.1	152.95964	13.22 %
BOROHIDRURO DE BERILIO	302.5	311.6205	3.02 %
2,2,3,3,4-PENTAMETILPENTANO	508	496.81146	2.20 %
SULFURO DE BUTILO-UNDECILO	929.5	967.71674	4.11 %
1-PENTADECANOL	894.5	852.329	4.71 %
1,2-PROPANODIAMINA	316	398.49033	26.10 %
TRIETANOLAMINA	472	468.349	0.77 %
DISULFURO DE HEPTILO	927.5	713.83856	23.04 %
2,3-DIMETIL-2,3-DIFENILBUTANO	781	811.1309	3.86 %
I-HEPTADECINO	949.5	976.56635	2.85 %
PARA-XILENO	379.1	392.04495	3.41 %
ETILACETOACETATO	391	391.24802	0.06 %
ACETANILIDA	430	439.02686	2.10 %
TETRAFLUOROHIDRAZINA	213	252.07732	18.35 %
M-DICLOROBENCENO	351	355.17365	1.19 %
CLOROPRENO	273	253.58887	7.11 %
CLOROACETATO DE METILO	270	264.77094	1.94 %
P-FENILENDIAMINA	317	365.79196	15.39 %

Modelo RED-Tc: Propiedad Temperatura Critica

Dado el tamaño relativamente pequeño del conjunto de datos y la complejidad de los descriptores químicos, se optó por una arquitectura de red neuronal simple. Esto mejora la eficiencia computacional y ayuda a evitar el sobreajuste en un conjunto de datos limitado, lo que facilita la generalización a datos nuevos o de prueba.

- **Arquitectura del Modelo**(Figura 5.13) La arquitectura del Modelo Red-Tc se compone de tres capas neuronales: una capa de entrada con 12 neuronas, seguida por dos capas ocultas con 6 y 3 neuronas respectivamente, y finalmente una capa de salida con 1 neurona. Las funciones de activación ReLU se aplican en las capas ocultas para introducir no linealidades en el modelo. La Figura ilustra esta arquitectura.

- **Métricas de Rendimiento** En el conjunto de prueba, se evaluaron varias métricas de

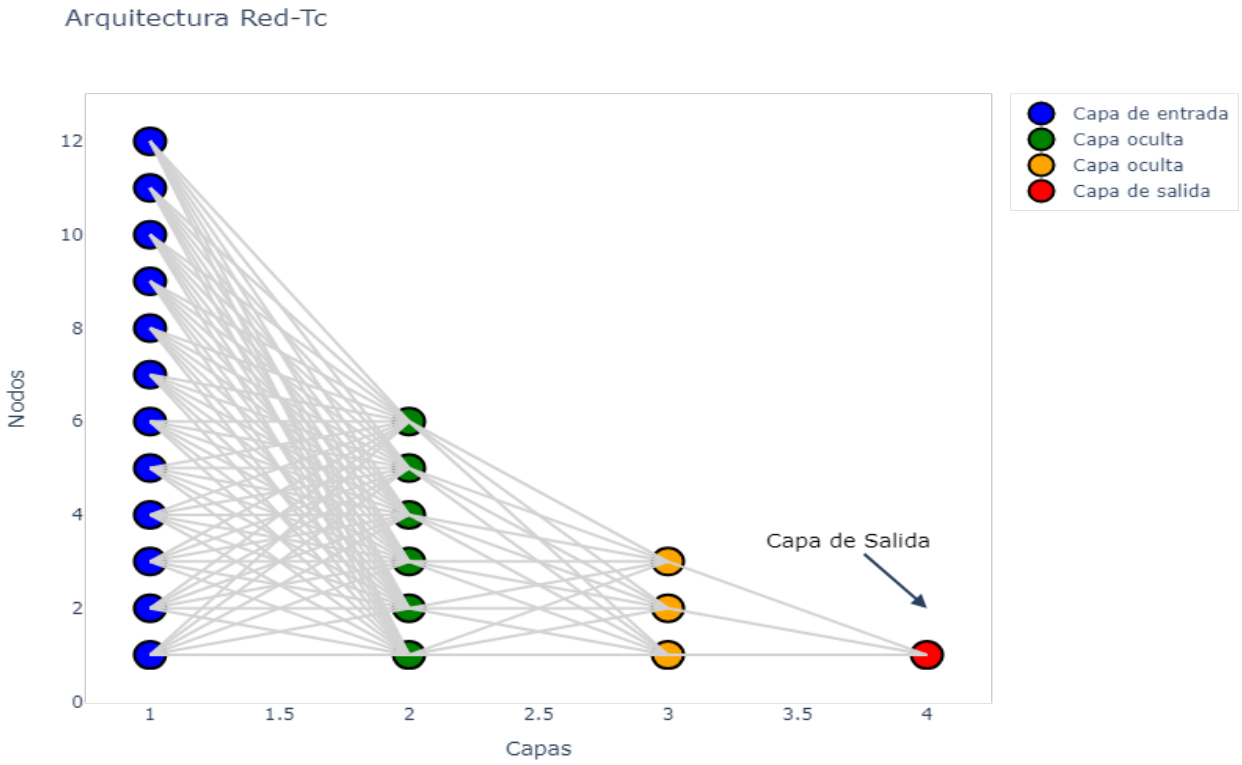

Figura 5.13: Arquitectura de Red-Tc

rendimiento para medir la calidad del Modelo Red-Tc.

Tabla 5.6: Métricas de Rendimiento en el Conjunto de Prueba

MSE (Error Cuadrático Medio)	0.0731
RMSE (Raíz del Error Cuadrático Medio)	0.2703
MAE (Error Absoluto Medio)	0.1704
R^2 (Coeficiente de Determinación)	0.6479

Curvas de Pérdida en Entrenamiento, Validación y Prueba(Figura 5.14)

Las curvas de pérdida en entrenamiento, validación y prueba proporcionan información sobre el proceso de entrenamiento del modelo. La convergencia de las curvas de validación y prueba, junto con la curva de entrenamiento por encima, indica la posibilidad de sobreajuste, donde el modelo mostrara un desempeño ligeramente superior con los datos de entrenamiento, que con los de validación y prueba.

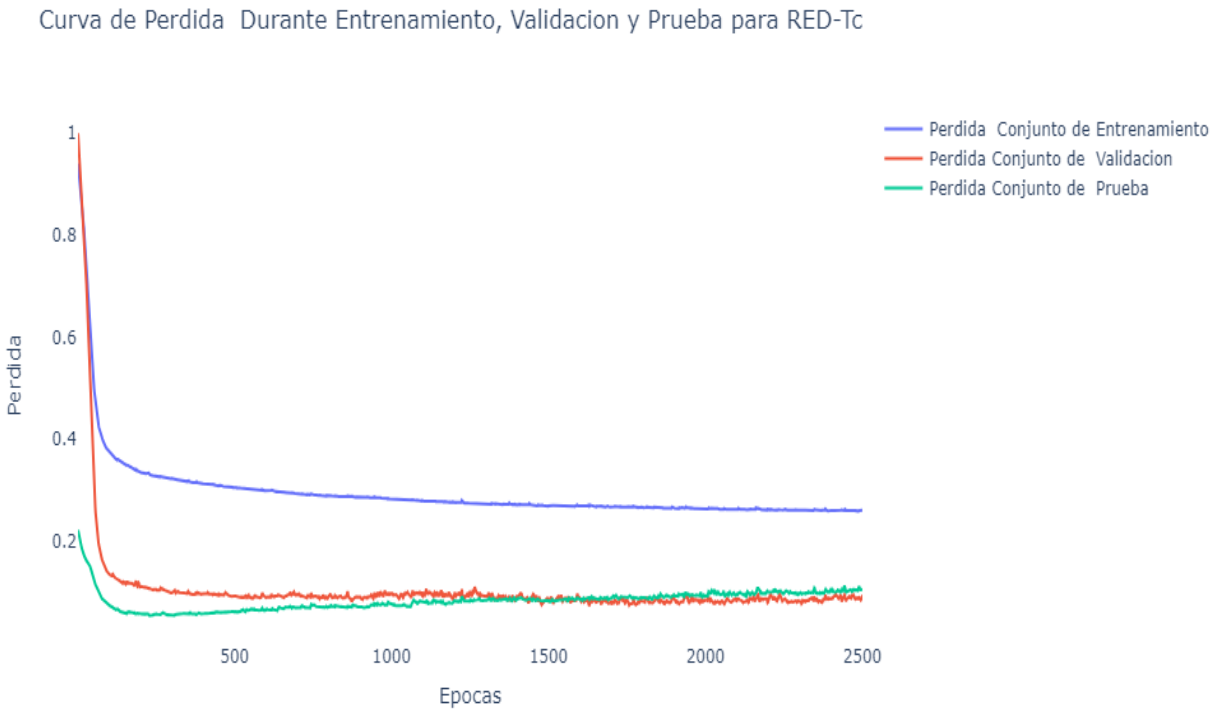

Figura 5.14: Arquitectura de Red-Tc

Gráfica de Calibración(Figura 5.15) La gráfica de calibración muestra cómo las predicciones del Modelo Red-Tc se comparan con los valores reales en el conjunto de prueba. La subdispersión en la gráfica sugiere que el modelo está subestimando la incertidumbre, debido a la aglomeración y poca distribución de los datos sobre la linea ideal de calibración esto indica que las predicciones están más agrupadas de lo esperado.

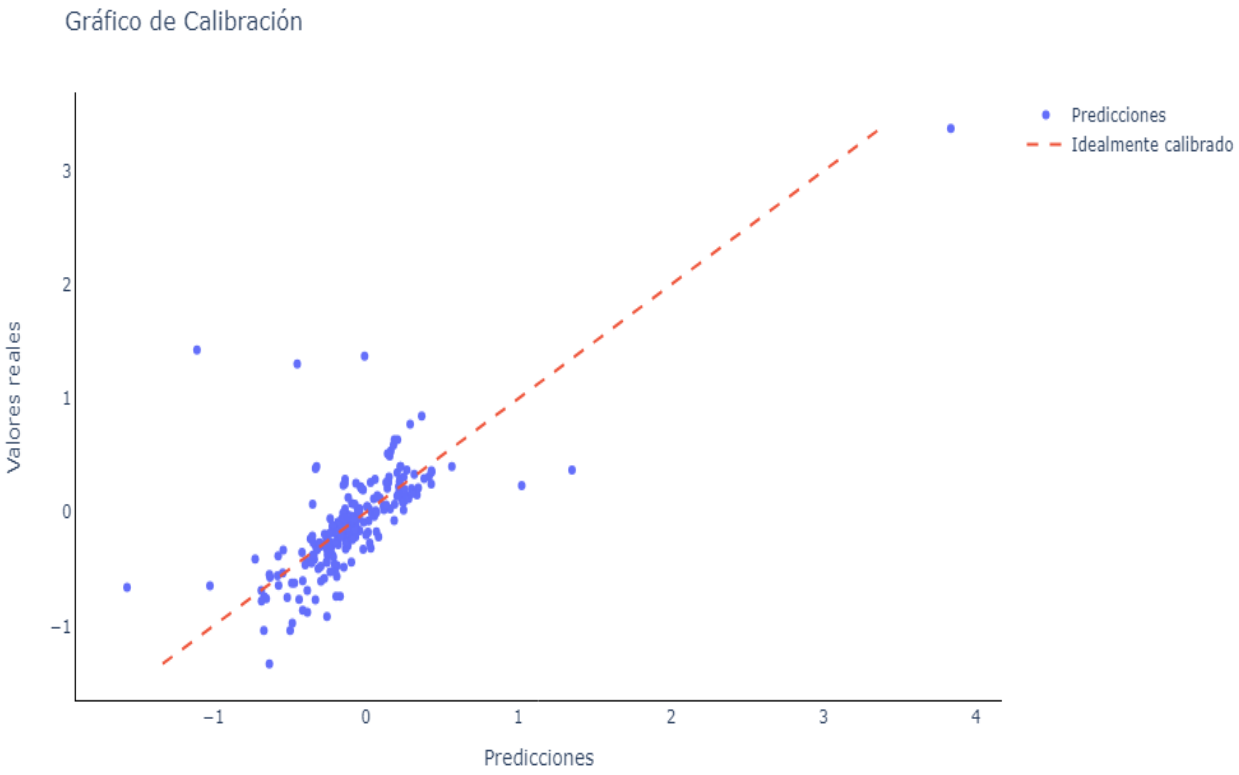

Figura 5.15: Grafica de Calibracion

Gráfica de Residuales Análisis(Figura 5.16) Es importante interpretar la gráfica de residuales en el contexto del modelo y el dominio de los datos. Si se encuentran problemas en la gráfica de residuales, Para la RED-Tc en particular tenemos 8 outliers que podrían explicar los resultados en la eficiencia del modelo, se deberían eliminar y entrenar el modelo nuevamente pero en este caso no era conveniente disminuir los datos de entrada.

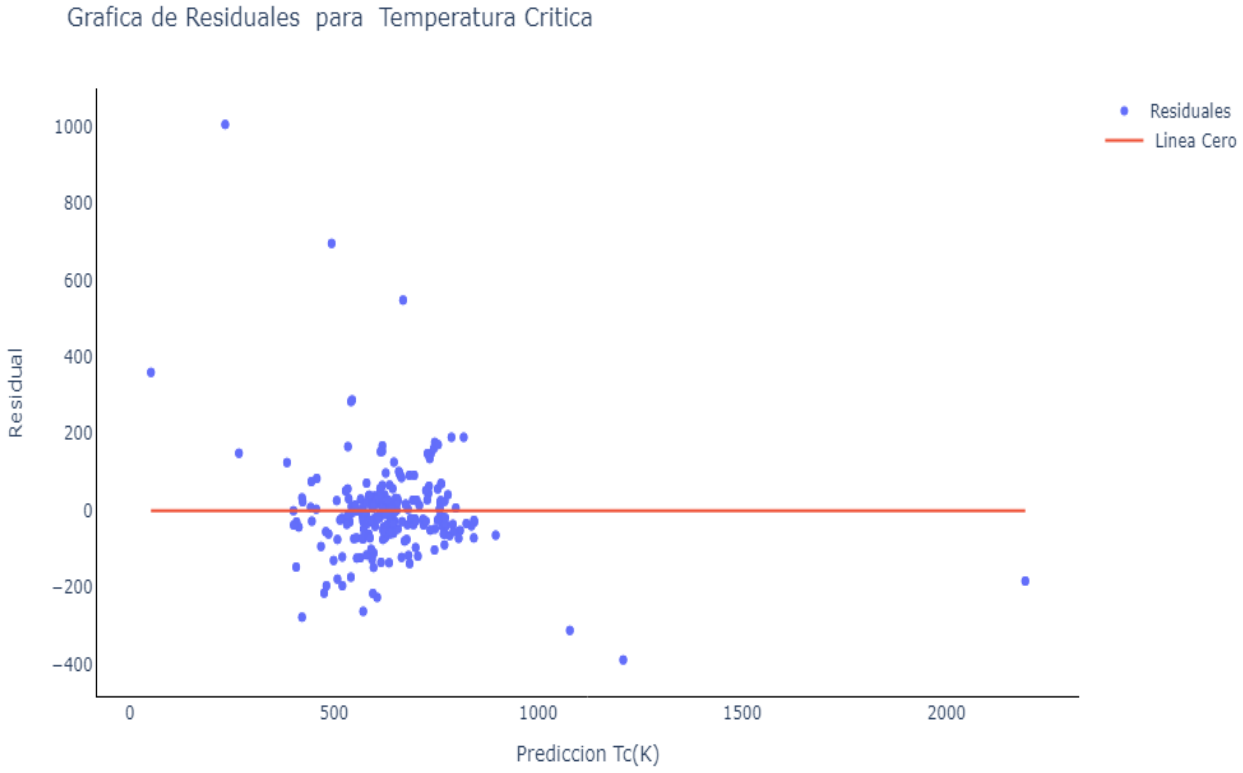

Figura 5.16: Grafica de Residuales para RED-Tc

Gráfica de Calibración Valores Reales vs Predicciones RED-Tc(Figura 5.17

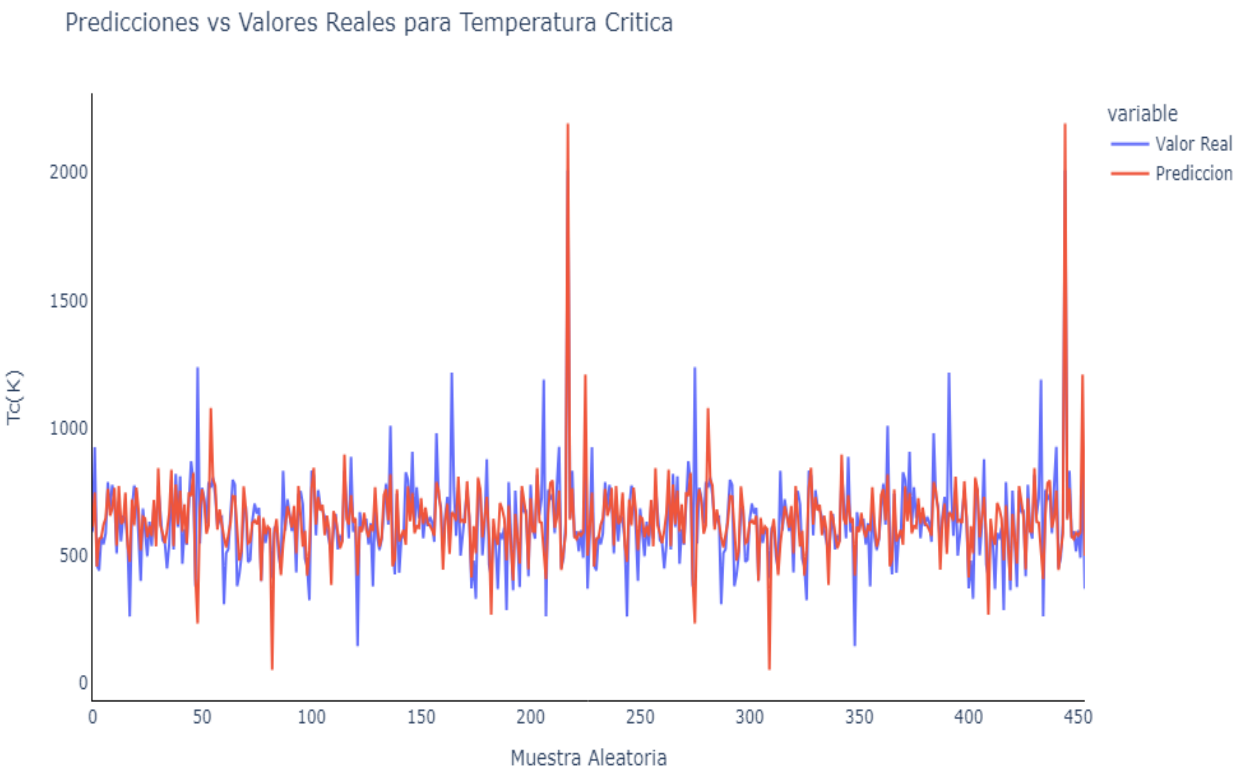

Figura 5.17: Grafica Valores Reales vs Predicciones

En este gráfico se observa como el modelo tiene dos picos que se salen del umbral esperado en las predicciones, en la muestra 217 y en la muestra de 444 del conjunto de 450 valores de prueba. **Valores Reales y Valores Predecidos en una muestra Aleatoria del conjunto de datos para Prueba de Eficiencia del Modelo RED-Tc** (Figura 5.7)

Tabla 5.7: Resultados de las predicciones y la incertidumbre en el conjunto de prueba.

Sustancia Química	Valor Real Tc(K)	Predicción Tc(K)	Incertidumbre (%)
3-ETIL-4-METILHEXANO	593.7	609.67786	3 %
M-TERFENILO	924.85	739.2324	20 %
CLORURO DE ETILO	460.35	521.46783	13 %
CLOROSILANO DE METILO	442	494.92	12 %
2,3-DICLOROPROPENO	577	638.4816	11 %
CLORURO DE SULFORILO	545	528.68933	3 %
OCTAMETILCICLOTETRASILOXANO	586.5	782.8459	33 %
SULFURO DE BUTILO-DODECILO	787.27	738.9552	6 %
SULFURO DE BUTILO-PENTILO	681.56	652.92163	4 %
1,4-DIMETILNAFTALENO	776.77997	708.3297	9 %
1-HEXADECINO	724.26	763.47546	5 %
FORMIATO DE ETILO	508.4	521.8537	3 %
1-PENTADECINO	711.41	759.8294	7 %
DI-N-PROPILOAMINA	555.8	568.6514	2 %
2,3-DIBROMOBUTANO	656.96	713.90924	9 %
ISODECANOL	644	742.9268	15 %
ACRILATO DE ETILO	553	592.6866	7 %
TRIFLUORURO DE BORO	682.28	771.9259	13 %
ÁCIDO MALEICO	773	734.0275	5 %
SULFURO DE BUTILO-NONILO	748.42	718.42975	4 %

El modelo tiene una incertidumbre para el modelo RED-Tc de 12.82 %. un intervalo de incertidumbre aceptable del 5 %. al 20 %. Es importante destacar que el análisis de incertidumbre en las predicciones de una red neuronal es una parte crucial del proceso de modelado y debe ser considerado para garantizar que el modelo sea confiable y adecuado para su despliegue en aplicaciones. Los resultados específicos para esta red sugiere que se necesario realizar un entrenamiento, preferentemente incrementando el tamaño de la base de datos.

Modelo RED-Dc: Densidad critica

La RED-Dc es una red neuronal de regresión que se ha entrenado para predecir la propiedad de densidad crítica (Dc) de una sustancia química. Utiliza la función de activación ReLU en las capas ocultas para introducir no linealidades y mejorar el aprendizaje de relaciones complejas entre los descriptores químicos y la propiedad Dc. Se ha utilizado el optimizador SGD y la función de pérdida MSE durante el entrenamiento de 1000 épocas. Los datos se dividieron en conjuntos de entrenamiento, validación y prueba, y se aplicó escalado estándar para normalizar los datos. A continuación, se presenta un análisis detallado de los resultados obtenidos.

- **Arquitectura del Modelo** (Figura 5.18) : La arquitectura de la red neuronal La ar-

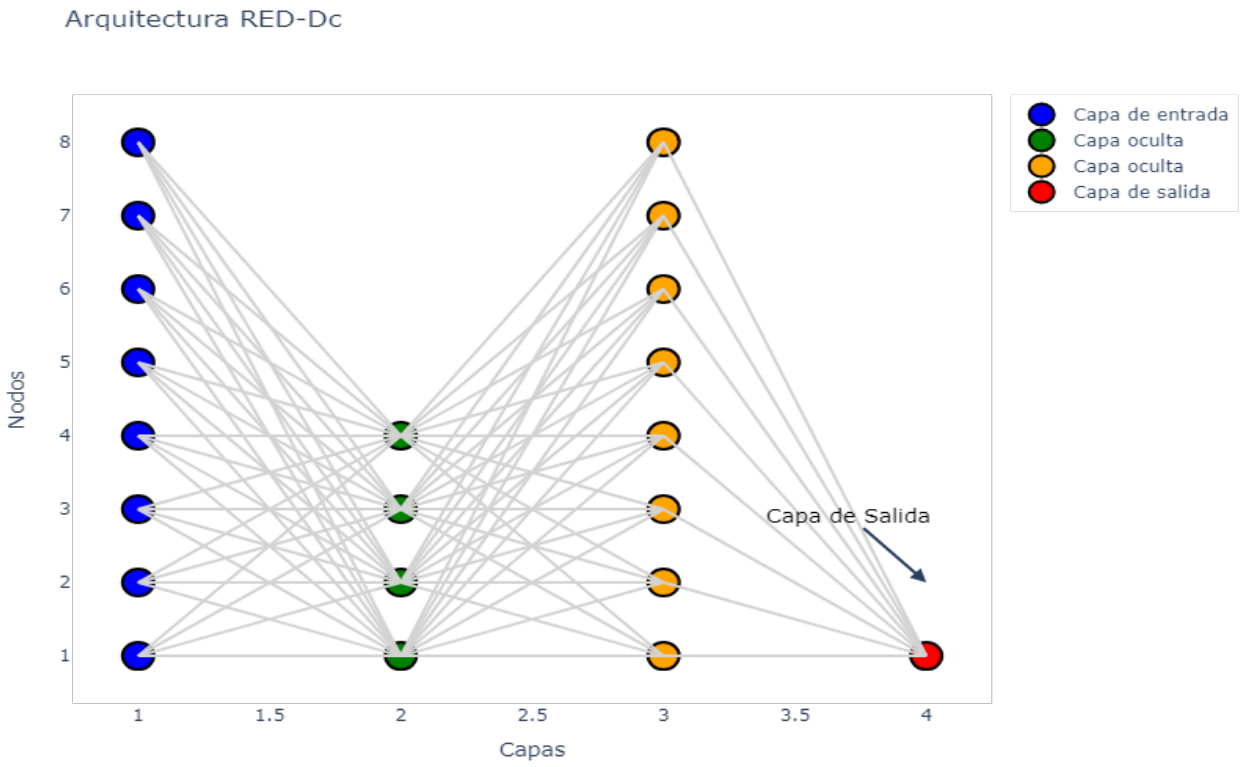

Figura 5.18: Arquitectura RED-Dc

quitectura de la red neuronal RED-Dc consta de 3 capas totalmente conectadas (fully connected layers). La primera capa tiene 8 neuronas de entrada, seguida de dos capas

ocultas con 4 y 8 neuronas respectivamente, y finalmente una capa de salida con 1 neurona que produce la predicción de densidad crítica.

- **Metricas de Rendimiento RED-Dc**:

Tabla 5.8: Métricas de Rendimiento en el Conjunto de Prueba para RED-Dc

MSE (Error Cuadrático Medio)	0.0073
RMSE (Raíz del Error Cuadrático Medio)	0.0852
MAE (Error Absoluto Medio)	0.0470
R^2 (Coeficiente de Determinación)	0.6196

- **Curvas de Pérdida en Entrenamiento, Validación y Prueba** El gráfico de pérdida en la Figura 5.19 muestra la evolución de la función de pérdida en función de las épocas de entrenamiento. En esta red se uso una funcion de optimizacion diferente denominada El Descenso de Gradiente Estocástico (SGD) es un optimizador que ajusta los parámetros de un modelo iterativamente utilizando el gradiente de la función de pérdida. A diferencia de Adam, que adapta las tasas de aprendizaje individualmente y mantiene estimaciones de momentos de gradientes, SGD utiliza una tasa de aprendizaje fija para todos los parámetros en cada iteración. Aunque SGD es más simple, en ocasiones se acopla mejor a los hiperparametros para lograr en la convergencia y adaptación a diferentes escenarios en comparación con el algoritmo complejo y adaptable de Adam. Al probar con Adam los resultados para el grafico de perdida no eran consistentes ni satisfactorios por lo que se opto por usar este optimizador.

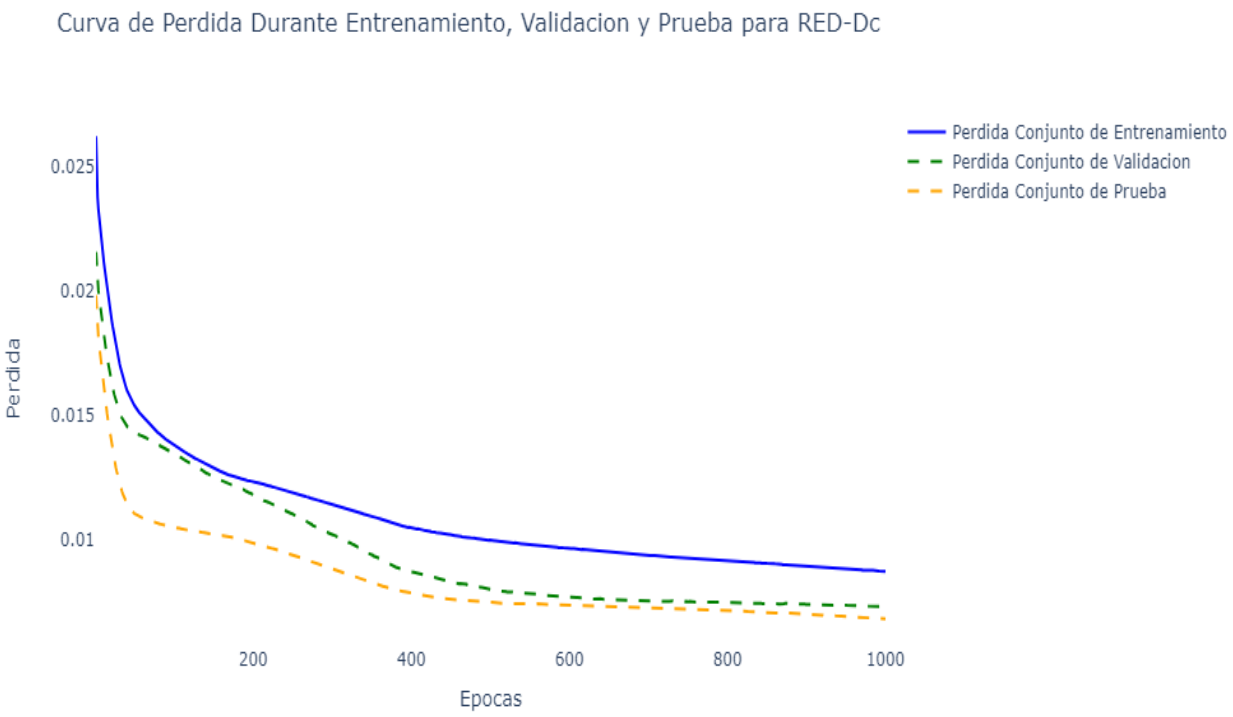

Figura 5.19: Función de Pérdida vs Épocas en Entrenamiento, Validación y Prueba

- **Grafica de Calibración**(Figura 5.20) En la evaluación de la red neuronal RED-Dc en la propiedad de densidad crítica (Dc), se observa una distribución homogénea en la gráfica de calibración. Aunque las predicciones realizadas por el modelo exhiben cierta dispersión en comparación con los valores reales, es importante destacar que esta dispersión se encuentra alineada con la línea de idealidad.

  La alineación con la línea de idealidad sugiere que el modelo RED-Dc logra un equilibrio satisfactorio entre precisión y generalización. Aunque hay variaciones puntuales en las predicciones, el hecho de que la dispersión siga la tendencia ideal implica que las estimaciones realizadas por el modelo están razonablemente calibradas en relación con los valores reales. Por lo tanto el modelo RED-Dc produce predicciones consistentes y, en promedio, cercanas a la realidad, a pesar de la variabilidad inherente en los datos.

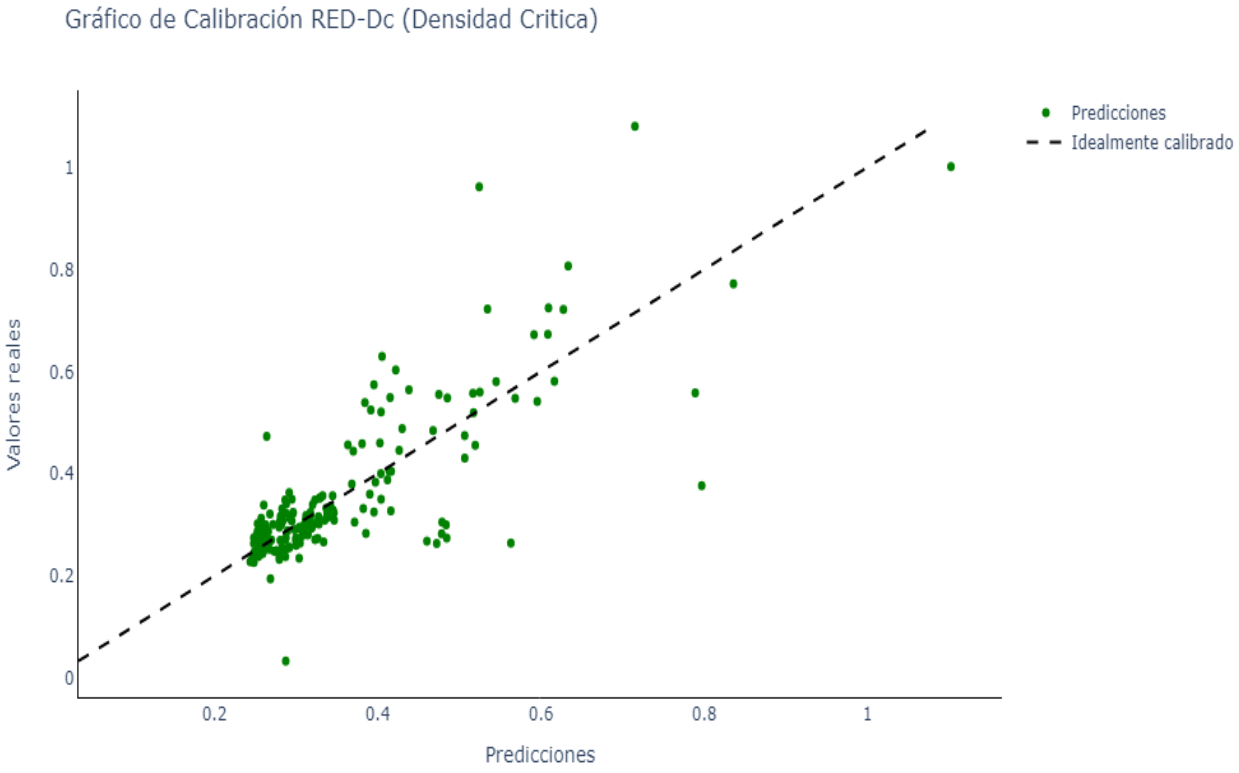

Figura 5.20: Calibracion RED-Dc

- **Grafica de Residuales Analisis** (Figura 5.21) La gráfica de residuales en RED-Dc pone de manifiesto una tendencia hacia la sobreestimación de los valores de densidad crítica, con una amplia acumulación de residuales positivos debido a la naturaleza de la propiedad. A pesar de esta tendencia, el modelo RED-Dc logra mantener un equilibrio entre precisión y manejo de casos excepcionales. Esta observación sugiere que el modelo es efectivo para proporcionar estimaciones razonables de densidad crítica, incluso si se inclinan ligeramente hacia valores más altos.

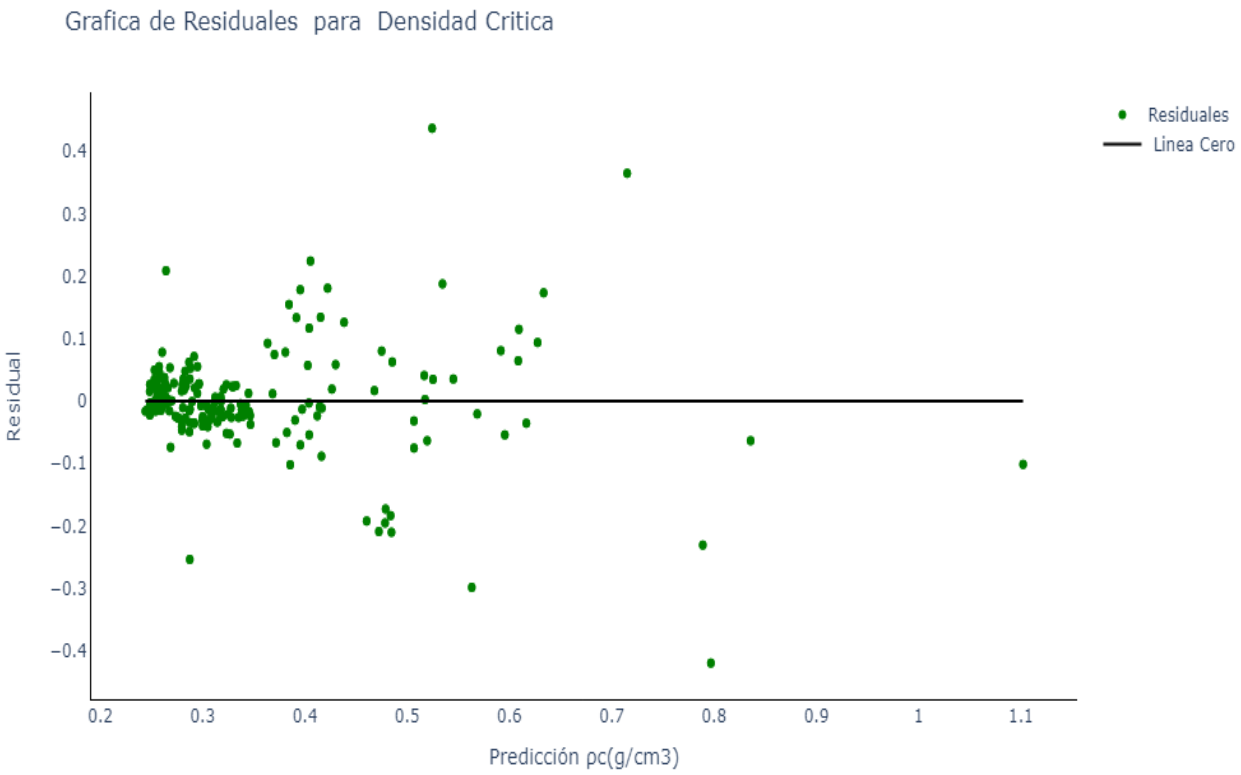

Figura 5.21: Grafica de Residuales RED-Dc

- **Grafica Valores Reales y Valores Predecidos en una muestra Aleatoria del conjunto de datos para Prueba de Eficiencia del Modelo RED-Dc** Figura 5.22)

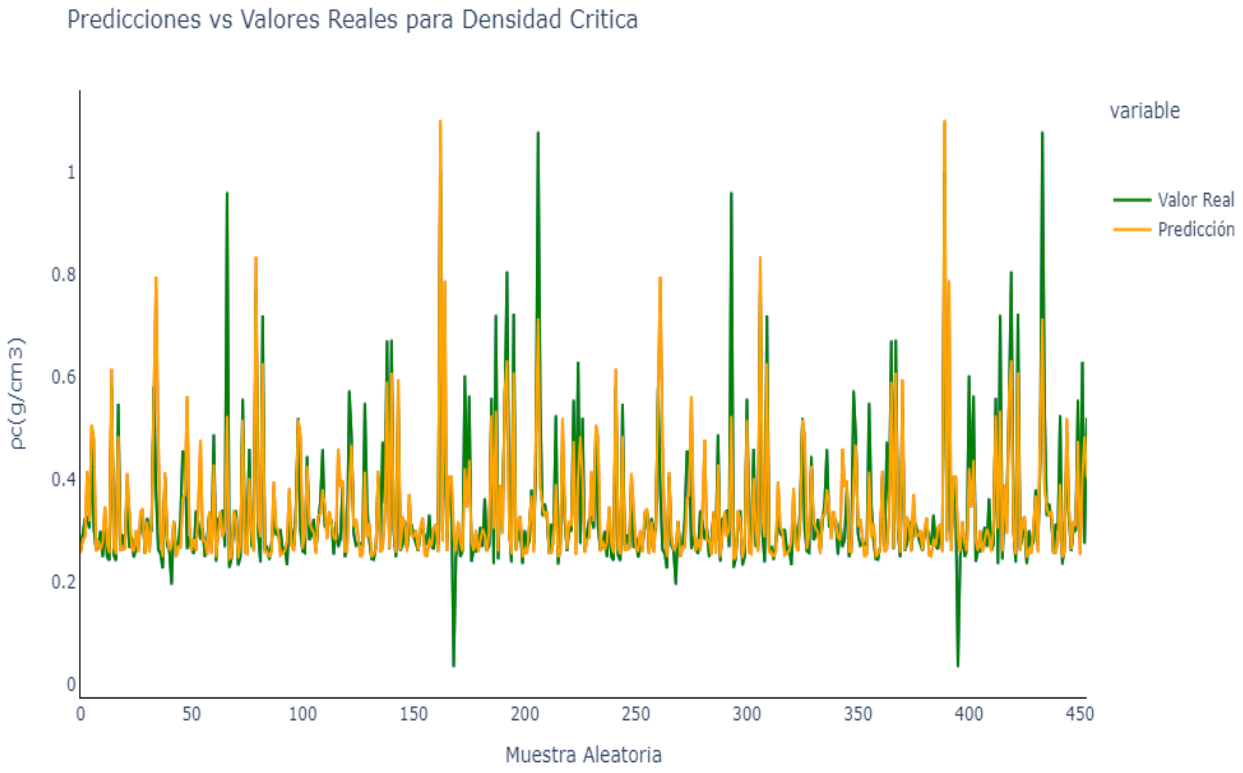

Figura 5.22: Grafica Valores Reales y Valores Predecidos en una muestra Aleatoria del conjunto de datos para Prueba de Eficiencia del Modelo RED-Dc

Tabla 5.9: Resultados de la predicción y la incertidumbre en el conjunto de prueba para RED-Dc.

Sustancia	Valor Real (g/cm3)	Predicción (g/cm3)	Diferencia	Incertidumbre (%)
BUTILO-TRIDECILO-SULFURO	0.2617	0.25658938	0.005110622	1.95 %
M-TERFENILO	0.3	0.28184032	0.018159688	6.05 %
CLORURO DE ETILO	0.3226	0.2874779	0.035122097	10.89 %
METIL CLOROSILANO	0.3276	0.41622794	0.088627934	27.05 %
METACRILATO DE ETILO	0.3044	0.31784147	0.013441473	4.42 %
1,1,1-TRICLOROETANO	0.4747	0.5066446	0.031944603	6.73 %
BENZOATO DE BENCILO	0.3058	0.47893763	0.17313763	56.62 %
SULFURO DE BUTILO-DODECILO	0.2623	0.2596258	0.002674222	1.02 %
CETONA DE METIL-ETILO	0.2701	0.2807472	0.010647208	3.94 %
1,2-DIMETILNAFTALENO	0.2996	0.2625393	0.037060708	12.37 %
ETILENIMINA	0.2489	0.2769149	0.028014898	11.26 %
FORMIATO DE ETILO	0.3235	0.3465283	0.023028284	7.12 %
1-PENTADECINO	0.2488	0.27342355	0.024623558	9.90 %
DIISOPROPILAMINA	0.2421	0.25600174	0.01390174	5.74 %
2,3-DIBROMOBUTANO	0.5812	0.6165458	0.035345793	6.08 %

La incertidumbre del modelo RED-Dc en el conjunto de prueba es del 15.39 %, lo que lo coloca como el tercero con mayor incertidumbre entre los seis modelos evaluados. Se observa que el modelo tiene un intervalo de adaptación coherente con los valores de densidad critica reales.

Modelo RED-Pc: Propiedad Presión Crítica

El modelo RED-Pc es una red neuronal desarrollada para predecir la propiedad termodinámica crítica "Pc"(Presión crítica) de compuestos químicos. Hay que mencionar que para RED-Tc y RED- Pc son las únicas 2 redes en este proyecto que se añadió una variable extra aparte de los descriptores de reactividad química en el caso de RED-Pc se agrego la variable Tc(K) Temperatura crítica debido a su estrecha relación por la técnica de matriz de correlación, la red consta de una iteración de 1500 épocas y una tasa de aprendizaje de 0.001

- **Arquitectura del Modelo**(Figura 5.23) : La arquitectura de la red neuronal La arquitectura del Modelo RED-Pc consta de tres capas, diseñadas para capturar relaciones complejas entre los descriptores químicos y la presión crítica. Consta de una capa de entrada con 10 neuronas, seguida de dos capas ocultas con 15 y 5 neuronas respectivamente, y una capa de salida con 1 neurona para predicciones continuas. Las funciones de activación ReLU en las capas ocultas permiten al modelo aprender eficazmente relaciones no lineales en los datos. Se ha buscado mantener una arquitectura simple para mejorar la eficiencia computacional y evitar el sobreajuste en el conjunto de datos limitado, favoreciendo la generalización a nuevos datos o pruebas.

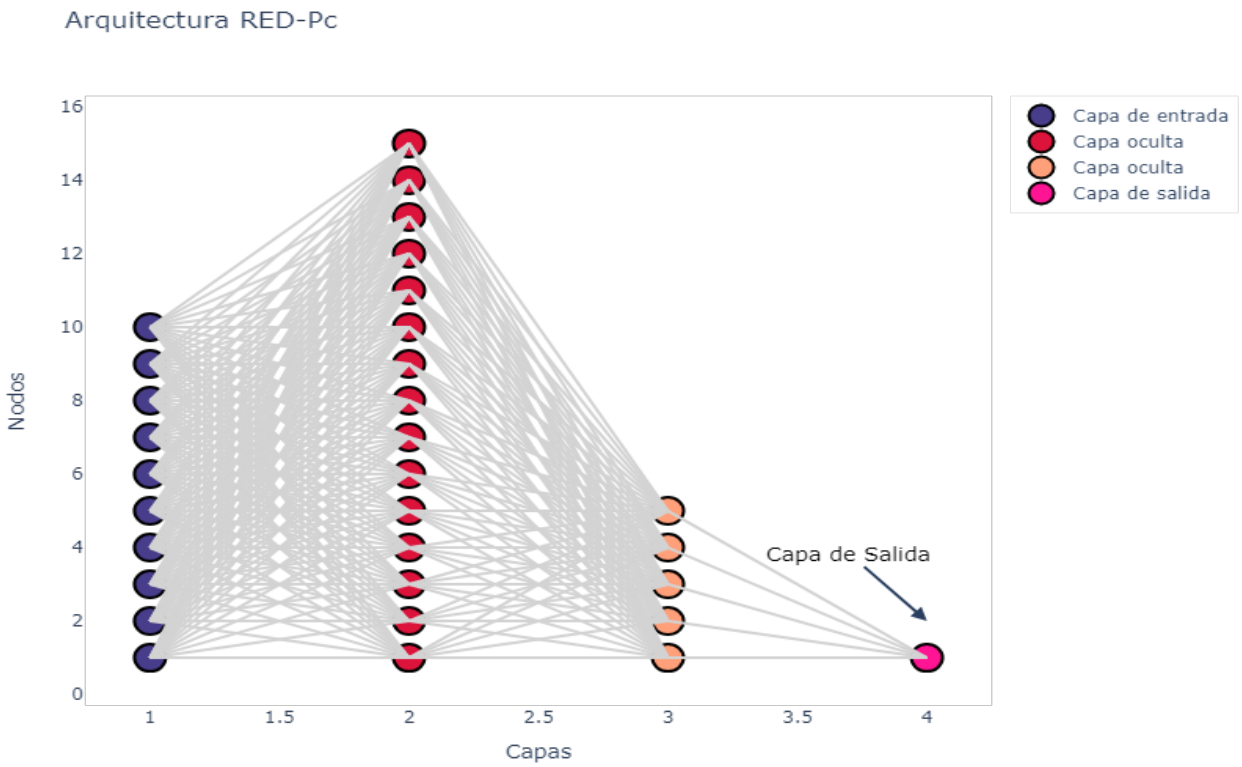

Figura 5.23: Arquitectura RED-Pc

- **Resultado Metricas**

Tabla 5.10: Métricas de Rendimiento en el Conjunto de Prueba

MSE (Error Cuadrático Medio)	0.2772
RMSE (Raíz del Error Cuadrático Medio)	0.5265
MAE (Error Absoluto Medio)	0.2434
R^2 (Coeficiente de Determinación)	0.7362

Estos valores de métricas proporcionan una idea del rendimiento del modelo RED-Pc. Un MSE bajo, RMSE bajo y MAE más bajo indican que las predicciones del modelo son más cercanas a los valores reales en el conjunto de prueba. El coeficiente de determinación R^2 cercano a 1 sugiere que el modelo explica una gran parte de la varianza total en los datos, esto sgnifica que las predicciones del modelo están muy cerca de los valores reales, y que el modelo es capaz de capturar y entender una proporción sustancial de las fluctuaciones o cambios en los datos, lo que es un buen indicativo de un buen ajuste del modelo a los datos.

En los valores de las métricas muestran que el modelo RED-Pc tiene un rendimiento aceptable para hacer predicciones de la propiedad presión critica.

- **Curvas de Pérdida en Entrenamiento, Validación y Prueba** (Figura 5.24 Para la RED-Pc se observa que la pérdida del conjunto de entrenamiento es menor que la pérdida en los conjuntos de validación y prueba, indica que el modelo está teniendo un ajuste a los datos de entrenamiento. Es decir, el modelo ha aprendido bien los patrones y relaciones presentes en el conjunto de entrenamiento y puede hacer predicciones precisas en esos datos específicos.

  Esto es una señal positiva de que el modelo está aprendiendo y generalizando adecuadamente los datos de entrenamiento, especialmente si la pérdida en los conjuntos de validación y prueba también está disminuyendo o se mantienen estables a medida que el entrenamiento avanza.

- **Gráfica de Calibración** 5.25

Al interpretar el gráfico, se tienen las siguientes observaciones:

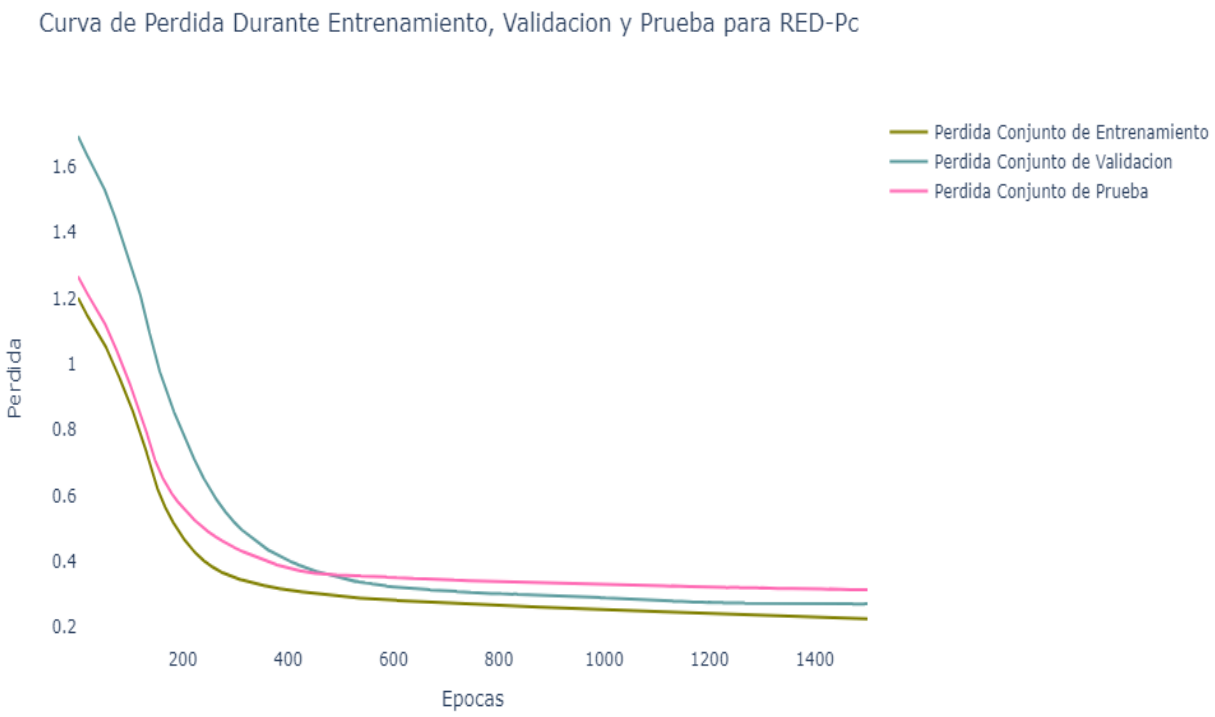

Figura 5.24: Gráfica de Perdida en Entrenamiento, Validación, Prueba para la RED-Pc

Los puntos en el gráfico de calibración están concentrados en la parte inferior de la línea de idealidad, significa que las predicciones del modelo tienden a estar sistemáticamente por debajo de los valores reales. Esto sugiere que el modelo está subestimando los resultados y que existe una tendencia hacia una mejor precisión en las predicciones en la región de valores más bajos.

Existen 3 valores grandes de presión que podrían afectar el comportamiento de las predicciones esperadas.

Esta subestimación sistemática tiene diversas implicaciones:

Sesgo en las predicciones: Él modelo tiene una tendencia a predecir valores más bajos de lo que realmente son. Por ejemplo, si se predicen presiones críticas (Pc) para diferentes compuestos químicos, el modelo podría estar subestimando las Pc reales, lo que llevaría a una menor estimación de la presión crítica en comparación con el valor real.

Falta de capacidad de generalización: Sí las predicciones están sesgadas hacia valores más bajos en el conjunto de prueba, esto indica que el modelo no ha aprendido correctamente las relaciones entre los descriptores químicos y la propiedad objetivo (en este caso, la presión crítica).

Errores en los descriptores químicos: La subestimación también se debe a errores en los descriptores químicos utilizados como entrada para el modelo. Si los descriptores no reflejan adecuadamente las propiedades reales de las moléculas, el modelo puede producir predicciones inexactas.

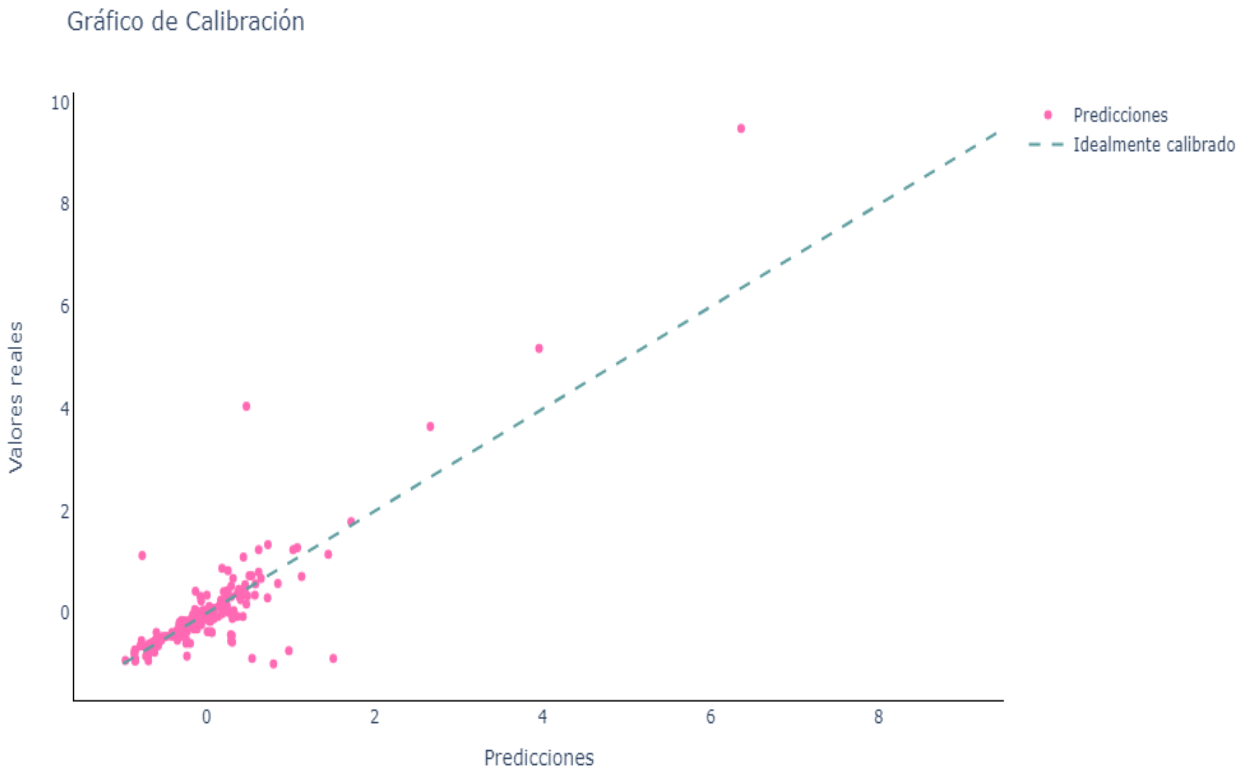

Figura 5.25: Grafica de calibración RED-Pc

- **Grafica de Residuales Analisis** 5.26

  Se observa que existen 5 puntos muy por encima de la línea cero: Los puntos que se encuentran por encima de la línea cero indican que el modelo ha subestimado la propiedad Pc para esos compuestos. Es decir, las predicciones del modelo son mayores que los valores reales de Pc. Asimismo hay 4 puntos por debajo de la línea cero esto indica que el modelo ha subestimado la propiedad Pc para esos compuestos. Es decir, las predicciones del modelo son menores que los valores reales de Pc.

  Respecto a la dispersión de los puntos en el modelo RED-Pc la mayoría se concentran alrededor de la línea cero esto nos da una idea de la precisión del modelo en sus predicciones. Si los puntos están más concentrados cerca de la línea cero, significa que el modelo tiene una buena precisión; de las 225 muestras de conjunto de prueba vemos que solo unas 8 o 9 datos se salen de un comportamiento adecuado.

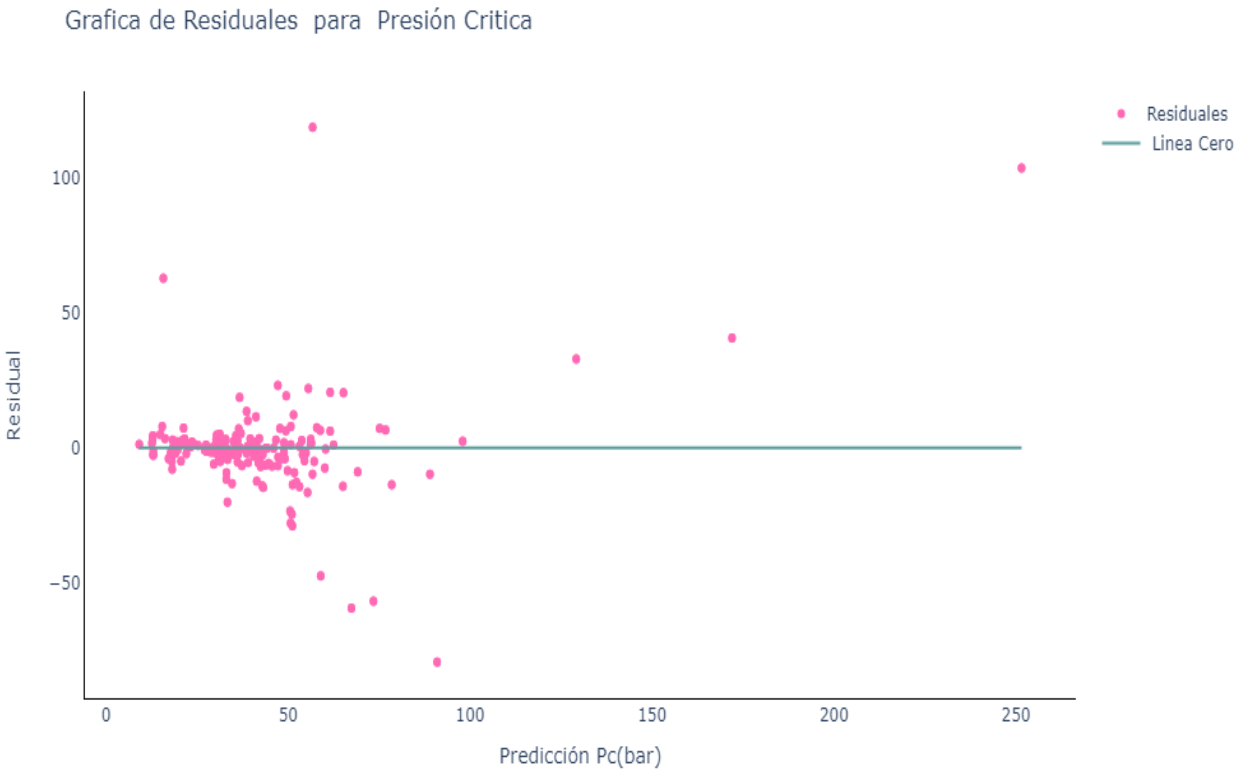

Figura 5.26: Grafica de Residuales RED-Pc

- **Grafica Valores Reales y Valores Predecidos en una muestra Aleatoria del conjunto de datos para Prueba de Eficiencia del Modelo RED-Pc**

  Él gráfico 5.27 ayuda a visualizar el comportamiento de las predicciones el modelo se acopla bien pero esto puede ser engañoso hasta calcular la incertidumbre total del modelo y su incertidumbre por muestra.

Tabla 5.11: Resultados de la predicción y la incertidumbre en el conjunto de prueba RED-Pc. (bar)

Sustancia Química	Valor Real	Predicción	Diferencia	Incertidumbre (%)
HIDRÓGENO SELENIUDO	83.44	76.79731	6.6426926	8 %
2,6-DIMETILNAFTALENO	31.7	32.81618	1.1161804	4 %
SILANO DE DIMETILO	35.6	42.45293	6.852932	19 %
2,2,4,4-TETRAMETILPENTANO	23.61	23.217752	0.39225006	2 %
P-AMINOAZOBENCENO	29	41.383213	12.383213	43 %
FLÚOR	52.15	57.17714	5.0271378	10 %
CICLOBUTENO	52.66	41.181576	11.478424	22 %
2-ETIL-1-HEXENO	30.7	37.321236	6.621235	22 %
N-BUTILCICLOHEXANO	25.7	23.595945	2.1040554	8 %
SULFURO DE BUTIL-HEXILO	21.41	33.023724	11.613722	54 %
FORMIATO DE N-PROPILO	40.63	43.05818	2.4281807	6 %
ÉTER METÍLICO DEL N-BUTILO SEC	34.1	35.55666	1.4566612	4 %
SULFURO DE BUTIL-HEPTILO	19.63	16.18402	3.445982	18 %
SULFURO DE BUTIL-HEXADECIL	10.27	12.972426	2.702427	26 %
ISOBUTIRATO DE ETILO	30.4	32.52802	2.1280193	7 %
2-METIL-1,3-BUTADIENO (ISOPRENO)	37.4	40.620243	3.2202415	9 %
1-CLORO-3-METILBUTANO	33.63	39.021828	5.3918266	16 %
CIANOACETATO DE ETILO	33.4	35.137836	1.7378349	5 %
CIANOACETATO DE METILO	38.1	40.9266	2.826603	7 %
SULFURO DE BUTIL-HEXILO	21.41	34.608353	13.198351	62 %

La incertidumbre general del modelo RED-Pc es de 26 %‘esto explica bien el comportamiento de de la gráfica de calibración 5.25 y la gráfica de residuales 5.26, contrastando con la curva de perdida 5.24 el modelo está entrenado por lo que se infiere que la falla en predicción es debido al conjunto de datos de entrada puede que los descriptores químicos de reactividad no interpretan de manera adecuada la presión critica esto nos lleva a un modelo con subestimación sistémica.

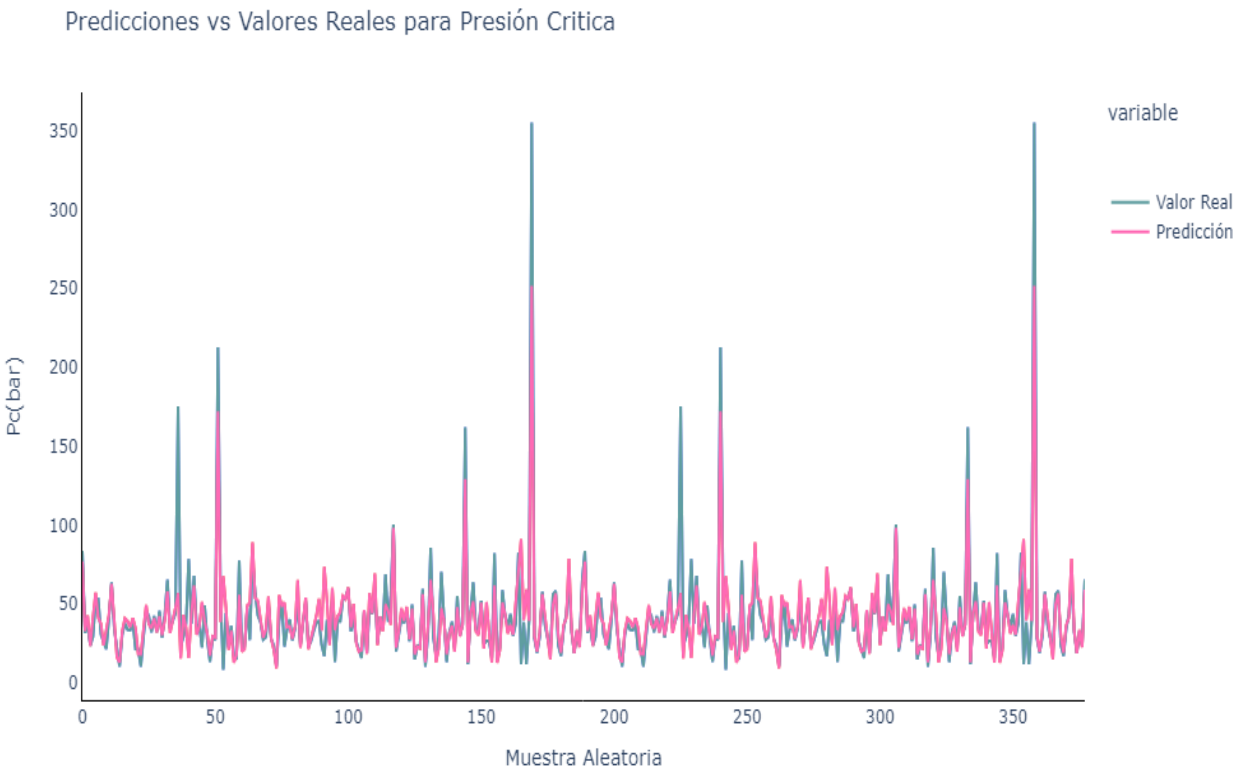

Figura 5.27: Prueba de modelo para RED-Pc

Modelo RED-OMEGA: Propiedad factor acentrico

La RED-OMEGA es una red neuronal de regresión que utiliza la función de activación ReLU en las capas ocultas para introducir no linealidades y mejorar el aprendizaje de relaciones complejas entre descriptores químicos y la propiedad factor acéntrico. El optimizador Adam y la función de pérdida MSE se utilizan durante el entrenamiento de 1000 épocas. Los datos fueron escalados para tener media cero y varianza unitaria, y el modelo se evaluó en conjuntos de validación y prueba para verificar su rendimiento.

- **Arquitectura del Modelo** (Figura 5.28: La arquitectura de la red neuronal La RED-OMEGA cuenta con 9 neuronas de entrada, seguida de dos capas ocultas con 9 neuronas cada una, y una capa de salida con 1 neurona.

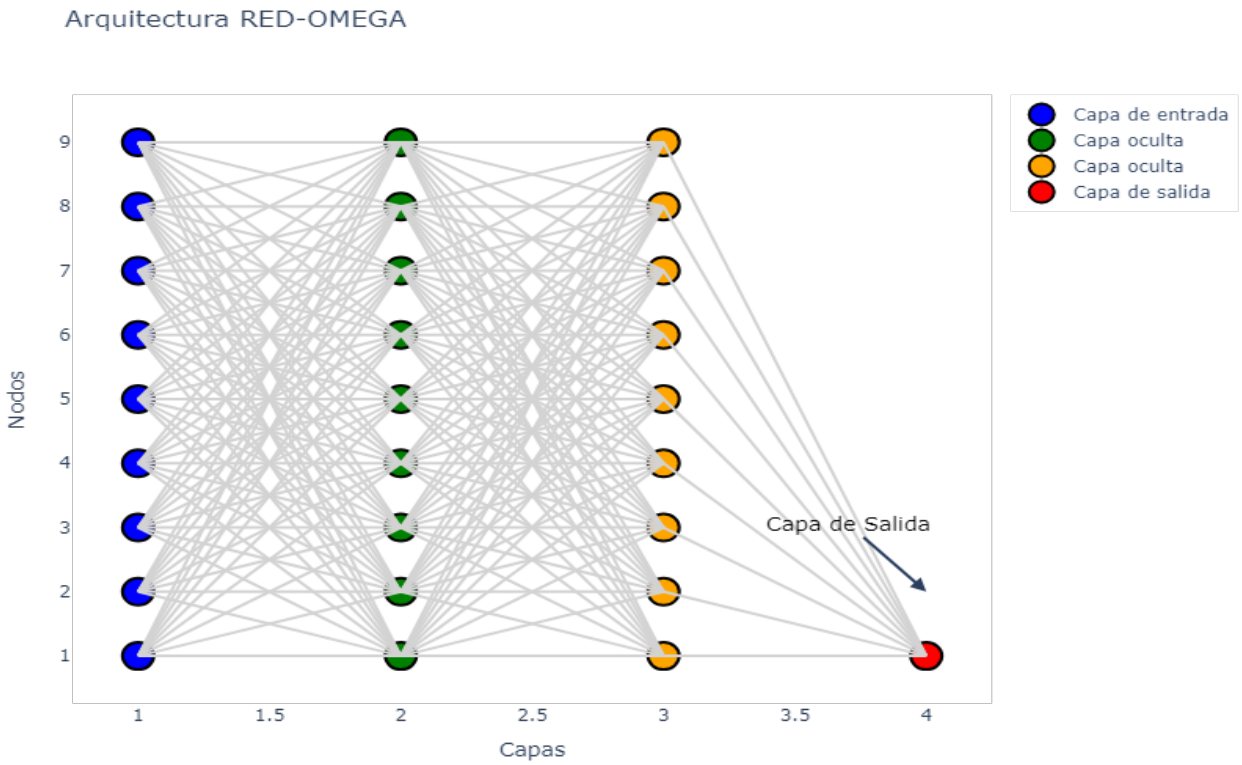

Figura 5.28: Arquitectura RED-OMEGA

- **Métricas de Rendimiento**: La Tabla 5.12 muestra las métricas de rendimiento del modelo RED-OMEGA en el conjunto de prueba.

Tabla 5.12: Métricas de Rendimiento en el Conjunto de Prueba para RED-OMEGA

MSE (Error Cuadrático Medio)	0.0731
RMSE (Raíz del Error Cuadrático Medio)	0.2703
MAE (Error Absoluto Medio)	0.1704
R^2 (Coeficiente de Determinación)	0.6479

- **Curvas de Pérdida en Entrenamiento, Validación y Prueba**:

  El gráfico de pérdida en la Figura 5.29 muestra la función de pérdida en función de las épocas de entrenamiento. A medida que aumentan las épocas, la pérdida en los conjuntos de entrenamiento y validación disminuye, este gráfico en particular muestra un comportamiento correcto lo que indica un entrenamiento adecuado.

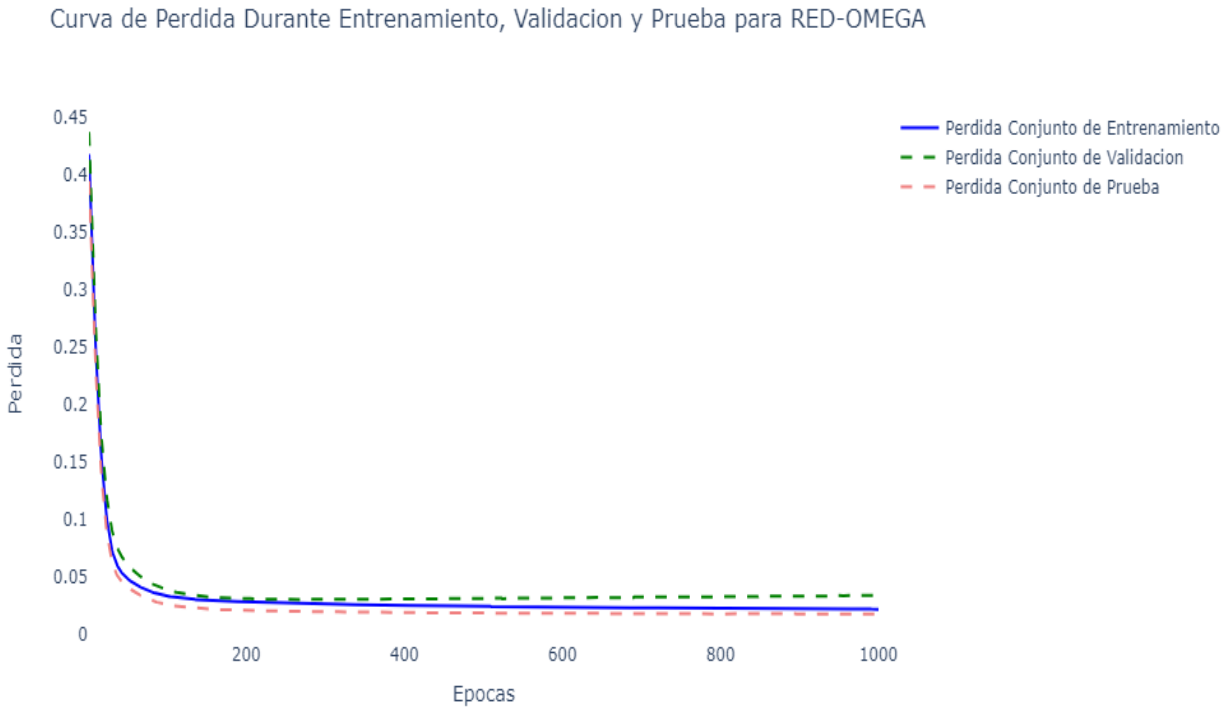

Figura 5.29: Funcion de Perdida durante Entrenamiento, Validación y Prueba RED-OMEGA

- **Grafica de Calibracion** La gráfica de calibración en la Figura 5.30 muestra que las predicciones del modelo RED-OMEGA están cerca de la línea de idealidad, lo que indica que el modelo produce estimaciones calibradas de la propiedad OMEGA.

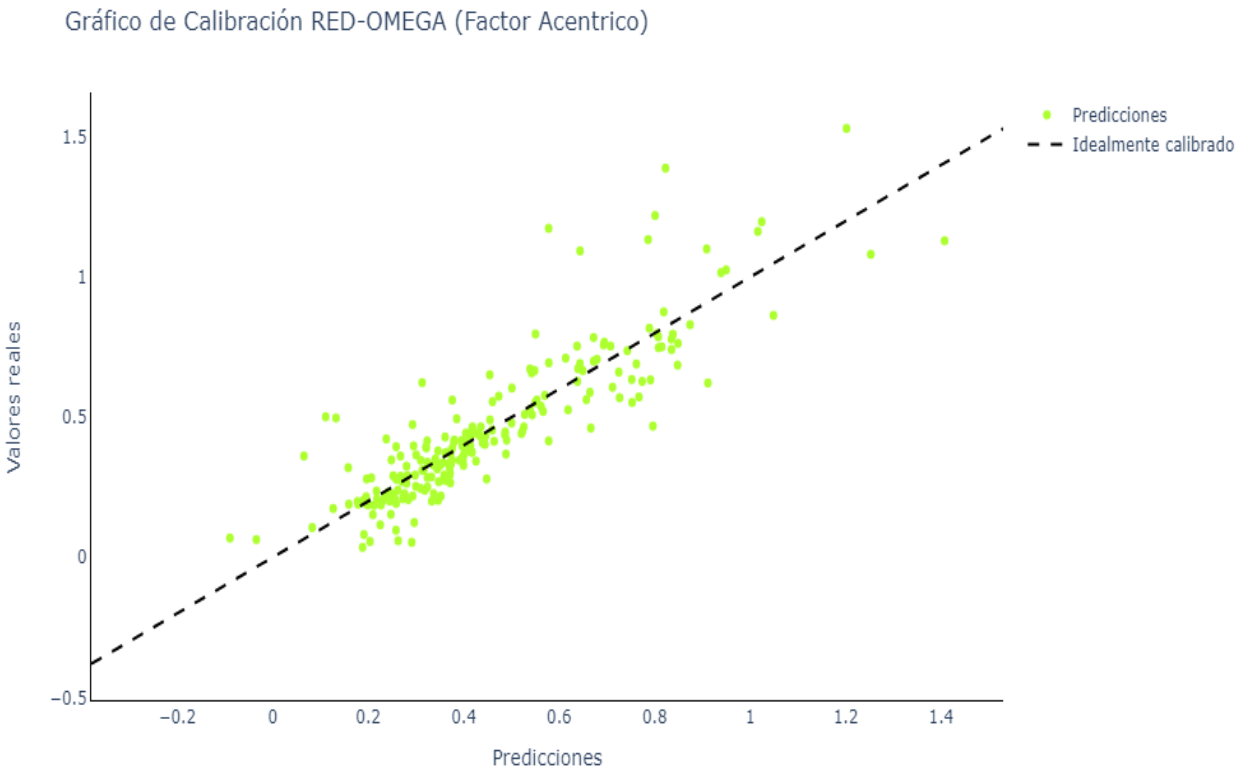

Figura 5.30: Grafica de Calibracion RED-OMEGA

- **Gráfica de Residuales Análisis**: En la Figura 5.31, la gráfica de residuales distribuye de manera uniforme los puntos de diferencia entre los valores reales y las predicciones alrededor de la línea cero, es decir, el modelo tiene **Variabilidad inherente** ya que incluso cuando la dispersión de los residuales es homogénea, podría haber variabilidad natural en los datos. Algunas predicciones tienen un error más alto simplemente debido a la complejidad intrínseca de los datos en cuestión.

Figura 5.31: Grafica de Residuales RED-OMEGA

- **Gráfica Valores Reales y Valores Predecidos en una muestra Aleatoria del conjunto de datos para Prueba de Eficiencia del Modelo RED-OMEGA** :

  El grafico 5.32 muestra los valores reales y los valores predecidos por el modelo RED-OMEGA en una muestra aleatoria del conjunto de prueba lo que indica una concordancia general entre las predicciones y los valores reales.

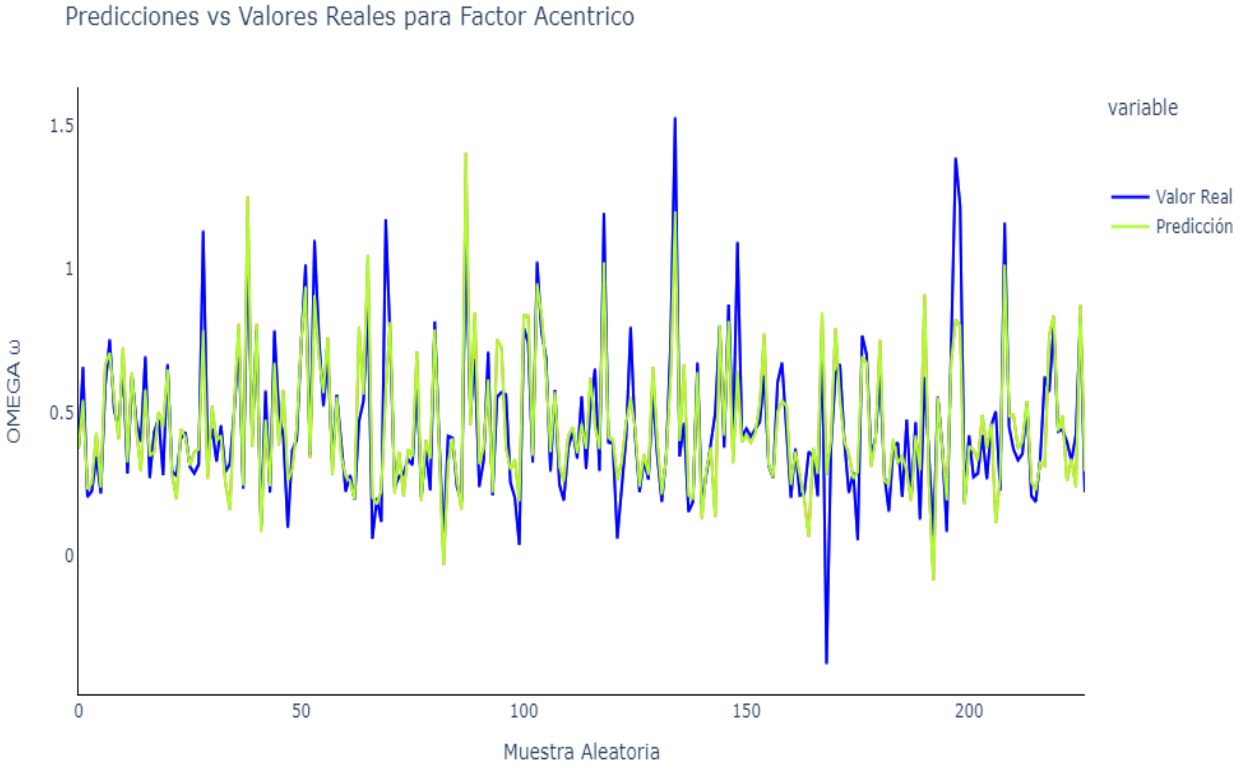

Figura 5.32: Valores Reales vs Predecidos RED-OMEGA

La tabla 5.13 apunta que los resultados de las predicciones del modelo RED-OMEGA en el conjunto de prueba. Cada fila muestra el valor real de la propiedad OMEGA, la predicción realizada por el modelo, la diferencia entre ambos valores y la incertidumbre en porcentaje. Esta tabla permite evaluar la precisión y confiabilidad de las predicciones del modelo.

La incertidumbre total del modelo RED-OMEGA para el conjunto de prueba es de %26.17 se inferiere que esto se debe a la calidad y la cantidad de datos de entrenamiento y prueba ya que estos son fundamentales para la generalización y la reducción de la in-

certidumbre. Si el conjunto de datos es limitado en términos de tamaño o diversidad puede sesgar el modelo,

Tabla 5.13: Resultados de la predicción y la incertidumbre en el conjunto de prueba para RED-OMEGA.

Sustancia	Valor Real (ω)	Predicción (ω)	Diferencia	Incertidumbre (%)
DECAFLUOROBUTANO	0.372	0.37325835	0.0013	0.34 %
M-TERFENILO	0.658	0.5423733	0.1156	17.57 %
CLORURO DE ETILO	0.204	0.23083302	0.0268	13.15 %
METIL CLOROSILANO	0.225	0.25162065	0.0266	11.83 %
2,2,4,4-TETRAMETILHEXANO	0.344	0.42597413	0.0820	23.83 %
BORURO DE TRIBROMO	0.216	0.23562229	0.0196	9.08 %
1-DODECANAL	0.754	0.70757234	0.0464	6.16 %
SULFURO DE BUTILO-PENTILO	0.508	0.5434532	0.0355	6.98 %
4-ETILOCTANO	0.443	0.40451878	0.0385	8.69 %
1-HEXADECINO	0.661	0.7248824	0.0639	9.66 %
ETILAMINA	0.285	0.3239854	0.0390	13.68 %
1-PENTADECINO	0.628	0.63820547	0.0102	1.63 %
1-DECENO	0.465	0.43595588	0.0290	6.25 %
2,3,5-TRIMETILHEPTANO	0.397	0.29504836	0.1020	25.68 %

5.2 Discusión

En general, cuatro de los seis modelos de red neuronal presentaron un rendimiento satisfactorio para predecir las propiedades termodinámicas críticas. Los resultados muestran que las métricas de error cuadrático medio (MSE), raíz del error cuadrático medio (RMSE) y error absoluto medio (MAE) son consistentes con el rendimiento de cada modelo.

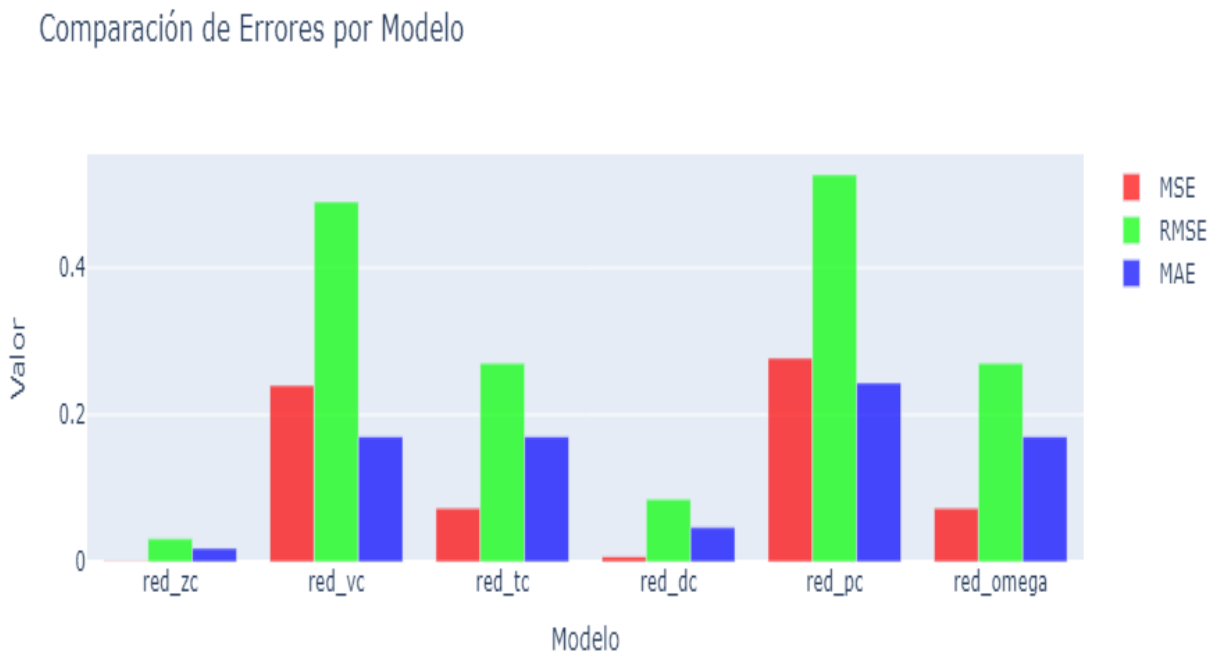

Figura 5.33: Comparacion de Errores por Modelo

En la figura 5.33 se detalla las metricas de error para cada modelo, en ciencia de datos y aprendizaje autoamtico no hay un estandar ideal respecto al valor de error esperado, entre el error sea menor se espera que la eficiencia sea optima. Se observa un comportamiento consistente en cuanto a la convergencia de los modelos durante el proceso de entrenamiento y validación. Las curvas de pérdida mostraron una disminución progresiva, lo que indica que los modelos aprendieron a partir de los datos de entrenamiento y generalizaron en los conjuntos de validación y prueba.

Es importante señalar que, aunque los modelos presentaron rendimiento general, existen variaciones en el desempeño entre los diferentes modelos y propiedades termodinámicas. Estas variaciones podrían estar relacionadas con la complejidad intrínseca de cada propiedad y la disponibilidad de datos de entrenamiento. Por lo tanto,es recomendable realizar análisis adicionales y considerar el uso de técnicas de mejora del rendimiento, como el incremento en el volumen de los datos y la optimización de hiperparámetros, para lograr un mejor rendimiento en casos específicos.

Tabla 5.14: Incertidumbre por modelo.

Modelo	**Incertidumbre (%)**
RED-Zc	7.92 %
RED-Vc	9.55 %
RED-Tc	12.82 %
RED-Dc	15.39 %
RED-OMEGA	26.19 %
RED-Pc	27.45 %

La Tabla 5.14 proporciona una perspectiva adicional sobre la incertidumbre asociada a cada modelo. Se observa que la RED-Pc y la RED-OMEGA presentan niveles de incertidumbre pora arriba del 20 % que como lo reporta la literatura en esos casos los modelos no son confiables en comparación con los otros modelos por debajo del 20 % . No obstante sus curvas de regresión mostraron un comportamiento cercano a la linealidad esto debe atribuirse a las particularidades de las propiedades en cuestión y a las complejidades inherentes en su relación con los descriptores químicos.

La implementación exitosa de modelos de redes neuronales para predecir propiedades termodinámicas críticas resalta el papel crucial de la química computacional en la aceleración de la predicción de propiedades químicas. Estos modelos demuestran el potencial de las técnicas de aprendizaje automático para abordar desafíos en la predicción de propiedades físicas y químicas. También hay que destacar que la elección de los descriptores químicos es algo crucial en la construcción de modelos predictivos precisos.5.1

Algunos descriptores son más relevantes para ciertos modelos en comparación con otros. Esto se debe a la naturaleza intrínseca de las propiedades termodinámicas y cómo se rela-

cionan con ciertos atributos moleculares. Por ejemplo, **Capacidad Calorifica Electronica** parece ser un descriptor importante en el modelo RED-Vc, mientras que **Orbital Donating Donor en HOMO** es relevante en RED-Pc y RED-Zc. La diversidad de descriptores utilizados en cada modelo sugiere que las propiedades termodinámicas dependen de diferentes características moleculares. Esto resalta la importancia de adaptar a los descriptores a las propiedades que se están prediciendo.

5.2.1. Limitaciones y Áreas de Mejora

Es fundamental considerar las limitaciones que este estudio presenta al interpretar sus resultados. Una de las principales es la **disponibilidad limitada de datos**. Las propiedades termodinámicas críticas, en muchos casos, son costosos y/o dificiles de obtener experimentalmente. Esta carencia de datos impacta la capacidad de los modelos para capturar relaciones complejas y específicas de cada propiedad. En consecuencia, podría haber casos en los que los modelos no lograron predecir con precisión debido a la falta de ejemplos de entrenamiento representativos.

Además, las **características intrínsecas de las propiedades termodinámicas** presentan desafíos. Algunas propiedades pueden dependen de factores intrínsecos de las moléculas que son difíciles de representar de manera efectiva a través de descriptores moleculares convencionales. Esto resulta en dificultades para el modelado preciso, incluso con enfoques de aprendizaje automático avanzados.

Para abordar estas limitaciones y mejorar el rendimiento de los modelos, se podrían considerar varias **áreas de mejora**. En primer lugar, **aumentar la cantidad de datos disponibles** es fundamental. Esto se lograría mediante la recopilación de datos experimentales adicionales o mediante técnicas de generación de datos sintéticos que mantengan las características esenciales de las propiedades termodinámicas.

Explorar **otras arquitecturas de redes neuronales** también es crucial. Si bien las arquitecturas utilizadas en este estudio han demostrado ser efectivas, existen variantes y enfoques respecto a lsa arquitectura que se adaptarían mejor a las propiedades específicas y sus rela-

ciones con los descriptores.

La **incorporación de información adicional** es beneficiosa. Esto podría incluir datos de otras áreas de la química, información estructural o propiedades físicas adicionales de las moléculas. Integrar estas fuentes de datos proporcionaría una comprensión más completa y holística de las relaciones entre descriptores y propiedades.

Este estudio reconoce las limitaciones inherentes y propone una mejora mediante estrategias como el aumento de datos, la exploración de arquitecturas alternativas y la incorporación de información adicional. Estas mejoras ayudarían a superar los desafíos y permitir que los modelos de redes neuronales alcancen su máximo potencial en la predicción de propiedades termodinámicas críticas.

5.2.2. Perspectivas Futuras

La investigación en el campo de la predicción de propiedades termodinámicas utilizando modelos de redes neuronales abre un emocionante camino hacia el futuro. Las perspectivas futuras en este ámbito sugieren una serie de direcciones prometedoras que enriquecen la capacidad predictiva y la aplicabilidad de estos modelos. Las perspectivas futuras en la predicción de propiedades termodinámicas utilizando modelos de redes neuronales se enfocan en mejorar la arquitectura de los modelos, integrar datos experimentales, aplicar técnicas avanzadas de aprendizaje automático y explorar nuevas propiedades. Estas direcciones ofrecen potencial para el avance de la investigación en este campo y su aplicación en la industria y la academia.

Capítulo 6

Conclusiones

En conclusión, los resultados obtenidos demuestran la viabilidad de utilizar modelos de red neuronal para predecir propiedades termodinámicas críticas. Estos modelos pueden ser una herramienta útil para acelerar el proceso de cálculo y proporcionar estimaciones rápidas y precisas de propiedades de interés en la química y la ingeniería.

El estudio de la materia en el estado crítico contribu ayel avance de la ciencia y la tecnología en múltiples áreas, por lo que este estudio podría fungir como punto de partida tanto para la predicción de propiedades como para el diseño de procesos.

Capítulo 7

Apéndices

A Apéndice A: Detalles Adicionales

A.1. Funcional WB97XD

La química computacional desafía continuamente los límites de la precisión y eficiencia en el análisis de sistemas moleculares complejos. En este contexto, el funcional de densidad WB97XD emerge como una herramienta excepcional que fusiona el poder del funcional híbrido de densidad funcional (DFT) B97 con una corrección por dispersión (D) llamada Exchange-Hole Dipole Moment (XDM). Esta combinación da lugar a un método de cálculo que se destaca por su capacidad para describir con precisión las interacciones intermoleculares, en particular las fuerzas de dispersión.

El corazón del funcional WB97XD reside en la colaboración entre el funcional de intercambio y correlación B97 y la corrección por dispersión. El funcional B97, desarrollado por Becke en 1997, es una mezcla ponderada de los funcionales B88 y B97, lo que permite tratar las interacciones electrónicas a distintas distancias de manera más precisa. Esta característica esencial se combina magistralmente con la corrección por dispersión basada en el método XDM, una creación de Khaliullin y colaboradores. Esta corrección se enfoca en el momen-

to dipolar inducido por la polarizabilidad de los electrones en la región de intercambio del funcional de Becke, logrando una descripción meticulosa de las interacciones dispersivas.

El funcional WB97XD se ha consolidado como una herramienta altamente precisa y versátil en la química computacional. Destaca en diversas aplicaciones, desde el estudio de interacciones de van der Waals hasta enlaces de hidrógeno y estados de transición, abriendo oportunidades para comprender sistemas químicos complejos con un grado excepcional de detalle. Su habilidad para capturar enlaces débiles y fuerzas de dispersión cruciales convierte a este funcional en una elección sobresaliente para el análisis de moléculas y sistemas de gran relevancia en química.

No obstante, es relevante mencionar que esta precisión viene acompañada de un mayor costo computacional. Si bien el funcional WB97XD proporciona resultados más precisos que los métodos DFT convencionales, la inclusión de la corrección por dispersión implica un aumento en la demanda de recursos computacionales. La elección de este funcional debe, por lo tanto, equilibrar la precisión buscada con la disponibilidad de recursos para obtener resultados eficientes.

En síntesis, el funcional WB97XD representa un paso significativo en la química computacional al combinar el funcional B97 con una corrección por dispersión mediante el método XDM. Su capacidad para describir con precisión interacciones dispersivas y enlaces débiles lo convierte en una herramienta esencial para el análisis de sistemas moleculares complejos. No obstante, la comunidad científica debe considerar cuidadosamente el costo computacional asociado con este enfoque y evaluar su pertinencia para problemas de investigación específicos.

B Apéndice B: Código Fuente

B.1. Modelo Red-Zc: Propiedad factor de comprensibilidad crítico

RED_ZC_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > RED_ZC_VALIDATION.py > ...

```python
# -*- coding: utf-8 -*-
"""
Created on Fri Jul 28 03:04:01 2023

@author: dano
"""
import torch
import torch.nn as nn
import torch.optim as optim
import pandas as pd

import matplotlib.pyplot as plt
import numpy as np

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from torch.utils.data import DataLoader, TensorDataset
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
# Load data from csv file
df = pd.read_csv('C:\\Users\\nama_\\DownLoads\\wetransfer_tesis_2023-07-25_0023\\TESIS\\dataset_final_sorted_2.4.3_CSV_prueba_red.csv')

# Split data into training and testing sets
#X = df[['Intensive_Entropy_Max','Theta_Inverse_Max', 'ODD_at_HOMO_Max',
#       'ODD_at_LUMO_Max', 'Theta_Inverse_Min',
 #      'thermophilicity_intensive_MAX',
  #     'electrophilicity_intensive_MAX', 'Dipole Moment', 'RHOC(g/cm3)']]
X = df[['ODD_at_HOMO_Max','ODD_at_LUMO_Max', 'Theta_Inverse_Max',
        'thermophilicity_intensive_MAX','Intensive_Entropy_Max','Intensive_Atomic_softness_Max']].values

y = df['Zc'].values
# Split the data into training, validation, and test sets
X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.30, random_state=34)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.50, random_state=34)
```

Figura B.1: Carga de datos, importación de librerías, Creación de conjuntos Entrenamiento,Validación y Prueba

RED_ZC_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > RED_ZC_VALIDATION.py > ...

```
34
35    # Scale the data to have zero mean and unit variance
36    sc = StandardScaler()
37    X_train = sc.fit_transform(X_train)
38    X_val = sc.transform(X_val)
39    X_test = sc.transform(X_test)
40
41    # Convert data to PyTorch tensors
42    X_train = torch.Tensor(X_train)
43    y_train = torch.Tensor(y_train)
44    X_val = torch.Tensor(X_val)
45    y_val = torch.Tensor(y_val)
46    X_test = torch.Tensor(X_test)
47    y_test = torch.Tensor(y_test)
48
49    # Unsqueeze the target tensors to add the third axis (batch size of 1)
50    y_train = y_train.unsqueeze(1)
51    y_val = y_val.unsqueeze(1)
52    y_test = y_test.unsqueeze(1)
53
54    # Create DataLoader objects for training, validation, and test data
55    train_data = TensorDataset(X_train, y_train)
56    val_data = TensorDataset(X_val, y_val)
57    test_data = TensorDataset(X_test, y_test)
58
59    batch_size = 32*2
60
61    train_loader = DataLoader(train_data, batch_size=batch_size, shuffle=True)
62    val_loader = DataLoader(val_data, batch_size=batch_size, shuffle=False)
63    test_loader = DataLoader(test_data, batch_size=batch_size, shuffle=False)
64
```

Figura B.2: Función Scaler,conversión de datos a tensores de PyTorch, ajustar tamaño de tensores para match en operaciones, Creación de DataLoaders para cada conjunto, Entrenamiento, Validación y Prueba

RED_ZC_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > RED_ZC_VALIDATION.py > ...

```
65  # Define the neural network architecture
66  class Net(nn.Module):
67      def __init__(self):
68          super(Net, self).__init__()
69          self.fc1 = nn.Linear(6, 9)
70          self.fc2 = nn.Linear(9, 6)
71          self.fc3 = nn.Linear(6, 1)
72          self.relu = nn.ReLU()
73      def forward(self, x):
74          x = self.relu(self.fc1(x))
75          x = self.relu(self.fc2(x))
76          x = self.fc3(x)
77          return x
78  net = Net()
79  # Define the loss function and optimizer
80  criterion = nn.MSELoss()
81  optimizer = optim.Adam(net.parameters(), lr=0.001)
82  # Training loop
83  train_losses = []
84  val_losses = []
85  test_losses = []
86  for epoch in range(500):
87      net.train()
88      train_loss = 0.0
89      for X_batch, y_batch in train_loader:
90          optimizer.zero_grad()
91          outputs = net(X_batch)
92          loss = criterion(outputs, y_batch)
93          loss.backward()
94          optimizer.step()
95          train_loss += loss.item() * X_batch.size(0)
96      train_loss /= len(train_loader.dataset)
97      train_losses.append(train_loss)
98
99      # Validation loop
100     with torch.no_grad():
```

Figura B.3: Definir Arquitectura, Definir función de perdida, función de optimización , junto con rango de aprendizaje, iteraciones del conjunto de datos de entrenamiento

RED_ZC_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > RED_ZC_VALIDATION.py > ...

```
    # Validation loop
    with torch.no_grad():
        net.eval()
        val_loss = 0.0
        for X_val_batch, y_val_batch in val_loader:
            outputs_val = net(X_val_batch)
            loss_val = criterion(outputs_val, y_val_batch)
            val_loss += loss_val.item() * X_val_batch.size(0)

        val_loss /= len(val_loader.dataset)
        val_losses.append(val_loss)
    # Test loop
    with torch.no_grad():
        net.eval()
        test_loss = 0.0
        for X_test_batch, y_test_batch in test_loader:
            outputs_test = net(X_test_batch)
            loss_test = criterion(outputs_test, y_test_batch)
            test_loss += loss_test.item() * X_test_batch.size(0)

        test_loss /= len(test_loader.dataset)
        test_losses.append(test_loss)

    if epoch % 1 == 0:
        print('Epoch [{}/{}], Train Loss: {:.4f}, Val Loss: {:.4f}, Test Loss: {:.4f}'.format(epoch + 1, 1000, train_loss, val_loss, test_loss))
# Plot the losses for the training set, validation set, and test set
import matplotlib.pyplot as plt

plt.plot(train_losses, label='Training Loss',  )
plt.plot(val_losses, label='Validation Loss',linestyle = "--")
plt.plot(test_losses, label='Test Loss',  linestyle = "--")  # Add this line to plot the test loss
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.title('Training, Validation, and Test Losses Zc')
plt.legend()
plt.show()
```

Figura B.4: Iteraciones del conjunto de datos de Validación y Prueba, imprimir en terminal épocas y valores de función de perdida para cada conjunto, gráfico de Perdida

RED_ZC_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > RED_ZC_VALIDATION.py >

```
predicted = outputs_test
# set the model to evaluation mode
net.eval()
with torch.no_grad():
    predicted = net(X_test)
    test_loss = criterion(predicted, y_test)
# create empty arrays to store predicted and actual values
predicted = np.array(predicted)

actual2 = np.array(y_test)
# loop through the test set and generate predictions
with torch.no_grad():
    for x, y in test_loader:
        x = x.to('cpu')
        y = y.to('cpu')
        y_pred = net(x)
        predicted = np.append(predicted, y_pred.cpu().detach().numpy())

        actual2 = np.append(actual2, y.cpu().detach().numpy())
# calculate the mean squared error for each output
mse2 = np.mean(np.square(predicted - actual2))
print("Mean Squared Error for output 2: {:.4f}".format(mse2))
# plot the predicted and actual values for each output separately
plt.figure(figsize=(15,6))
plt.subplot(2, 1, 1)
plt.plot(actual2, label="Actual")
plt.plot(predicted, label="Predicted")
plt.legend()
plt.title("Predicted vs Actual for Acentric Factor ")
plt.xlabel("Sample")
plt.ylabel("Value")


```

Figura B.5: Evaluación de Modelo, creación de matrices vacías, llenado por iteración de las matrices vacías con los resultados de predicción, gráfico de valores reales vs predicciones.

RED_ZC_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > RED_ZC_VALIDATION.py > ...

```
# plot the residuals for each output
residuals = actual2 - predicted
plt.figure(figsize=(15, 6))
plt.subplot(2, 1, 1)
plt.scatter(predicted, residuals)
plt.title("Residual Plot for Acentric Factor")
plt.xlabel("Predicted")
plt.ylabel("Residual")
plt.axhline(y=0, color='r', linestyle='-')
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
# Calculate MSE, RMSE, and MAE
mse = mean_squared_error(actual2, predicted)
rmse = np.sqrt(mse)
mae = mean_absolute_error(actual2, predicted)
# Calculate R-squared
r2 = r2_score(actual2, predicted)
print(f"MSE: {mse:.4f}")
print(f"RMSE: {rmse:.4f}")
print(f"MAE: {mae:.4f}")
print(f"R2: {r2:.4f}")
# Obtener predicciones del modelo para el conjunto de prueba
with torch.no_grad():
    net.eval()
    predictions_zc = net(X_test)
# Convertir las predicciones y las etiquetas a arrays de NumPy
predictions_zc = predictions_zc.cpu().numpy().flatten()
y_test4 = y_test.cpu().numpy().flatten()
# Guardar las variables en un archivo
np.savez('variables_predicciones4.npz', predictions_zc=predictions_zc,y_test4=y_test4)
# Graficar el gráfico de calibración
plt.figure(figsize=(8, 6))
plt.plot(predictions_zc, y_test, 'o', label='Predicciones')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', label='Idealmente calibrado')
plt.xlabel('Predicciones')
plt.ylabel('Valores reales')
plt.title('Gráfico de Calibración')
```

Figura B.6: Gráfico de Calibración, Metricas de Rendimiento, Grafico de Residuales

B.2. Modelo RED-Vc: Propiedad Volumen Critica

Red_Volumen_critica_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > Red_Volumen_critica_VALIDATION.py > ...

```python
# -*- coding: utf-8 -*-
"""
Created on Fri Jul 28 01:45:02 2023

@author: dano_
"""

# Import the required libraries
import torch
import torch.nn as nn
import torch.optim as optim
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from torch.utils.data import DataLoader, TensorDataset

# Load data from csv file
df = pd.read_csv('C:\\Users\\nama_\\DownLoads\\Wetransfer_tesis_2023-07-25_0023\\TESIS\\dataset_final_sorted_2.4.3_CSV_prueba_red.csv')

X = df[['molecular volume', 'Intensive_Heat_Capacity_Max',
        'electrophilicity_intensive_MAX', 'Intensive_Atomic_Hypersoftness_Max',
        'Optimum_Chemical_Potential_Min', 'Optimum_Chemical_Potential_Max',
        'OAD_at_LUMO_Max', 'OAD_at_HOMO_Max']].values

y = df['Vc(cm3/mol)'].values

# Split the data into training, validation, and test sets
X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.30, random_state=34)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.50, random_state=34)

# Scale the data to have zero mean and unit variance for input features (X)
sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_val = sc_X.transform(X_val)
X_test = sc_X.transform(X_test)
```

Figura B.7: Carga de datos, importación de librerías, Creación de conjuntos Entrenamiento,Validación y Prueba, Función Scaler aplicada a valores de entrada

Red_Volumen_critica_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > Red_Volumen_critica_VALIDATION.py >

```
# Scale the data to have zero mean and unit variance for the target variable (y)
sc_y = StandardScaler()
y_train = sc_y.fit_transform(y_train.reshape(-1, 1))
y_val = sc_y.transform(y_val.reshape(-1, 1))
y_test = sc_y.transform(y_test.reshape(-1, 1))

# Convert data to PyTorch tensors
X_train = torch.Tensor(X_train)
y_train = torch.Tensor(y_train)
X_val = torch.Tensor(X_val)
y_val = torch.Tensor(y_val)
X_test = torch.Tensor(X_test)
y_test = torch.Tensor(y_test)


# Create DataLoader objects for training, validation, and test data
train_data = TensorDataset(X_train, y_train)
val_data = TensorDataset(X_val, y_val)  # Create a DataLoader object for validation data
test_data = TensorDataset(X_test, y_test)

batch_size = 32

# Create DataLoader objects for training, validation, and test data
train_loader = DataLoader(train_data, batch_size=batch_size, shuffle=True,)
val_loader = DataLoader(val_data, batch_size=batch_size, shuffle=False, )
test_loader = DataLoader(test_data, batch_size=batch_size, shuffle=False, )

```

Figura B.8: Función Scaler para variables objetivo (predicciones), conversión de datos a tensores de PyTorch, Creación de DataLoaders para cada conjunto, Entrenamiento, Validación y Prueba

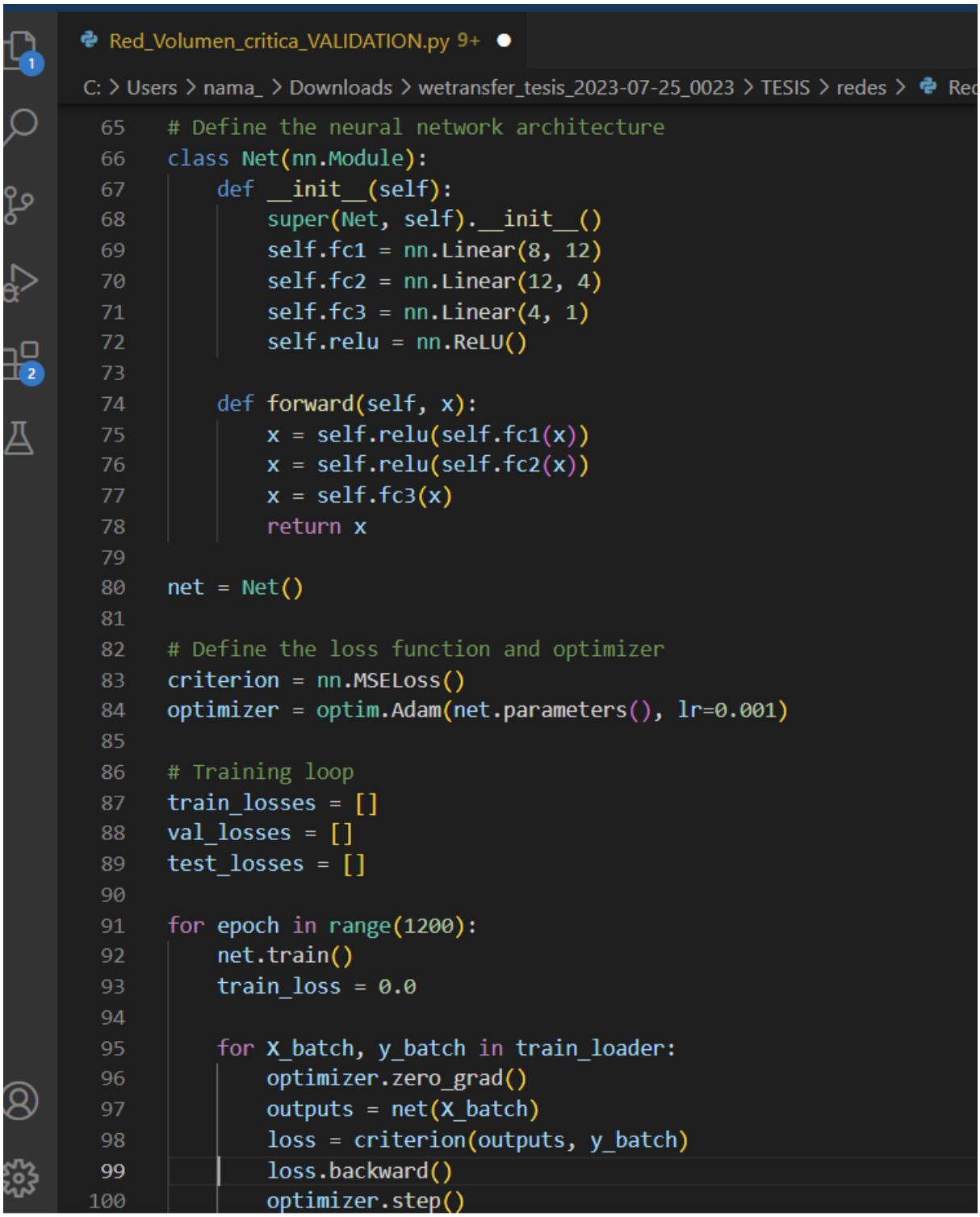

Figura B.9: Definir Arquitectura, Definir función de perdida, función de optimización , tasa de aprendizaje, iteraciones del conjunto de datos de entrenamiento

Red_Volumen_critica_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > Red_Volumen_critica_VALIDATION.py > ...

```
    train_loss /= len(train_loader.dataset)
    train_losses.append(train_loss)

# Validation loop
    with torch.no_grad():
        net.eval()
        val_loss = 0.0
        for X_val_batch, y_val_batch in val_loader:
            outputs_val = net(X_val_batch)
            loss_val = criterion(outputs_val, y_val_batch)
            val_loss += loss_val.item() * X_val_batch.size(0)

    val_loss /= len(val_loader.dataset)
    val_losses.append(val_loss)

# Test loop
    with torch.no_grad():
        net.eval()
        test_loss = 0.0
        for X_test_batch, y_test_batch in test_loader:
            outputs_test = net(X_test_batch)
            loss_test = criterion(outputs_test, y_test_batch)
            test_loss += loss_test.item() * X_test_batch.size(0)

    test_loss /= len(test_loader.dataset)
    test_losses.append(test_loss)


    if epoch % 1 == 0:
        print('Epoch [{}/{}], Train Loss: {:.4f}, Val Loss: {:.4f}, Test Loss: {:.4f}'.format(epoch + 1, 1000, train_loss, val_loss, test_loss))

```

Figura B.10: Iteraciones del conjunto de datos de Validación y Prueba, imprimir en terminal épocas y valores de función de perdida para cada conjunto.

Red_Volumen_critica_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > Red_Volumen_critica_VALIDATION.py > ...

```
135  # Plot the losses for the training set, validation set, and test set
136  import matplotlib.pyplot as plt
137
138  plt.plot(train_losses, label='Training Loss',  )
139  plt.plot(val_losses, label='Validation Loss',linestyle = "--")
140  plt.plot(test_losses, label='Test Loss',  linestyle = "--")  # Add this line to plot the test loss
141
142  plt.xlabel('Epoch')
143  plt.ylabel('Loss')
144  plt.title('Training, Validation, and Test Losses Vc')
145  plt.legend()
146  plt.show()
147
148  # Obtain predictions for each output separately
149  predicted = outputs_test
150  # set the model to evaluation mode
151  net.eval()
152  with torch.no_grad():
153      predicted = net(X_test)
154      test_loss = criterion(predicted, y_test)
155
156  # create empty arrays to store predicted and actual values
157  predicted = np.array(predicted)
158
159  actual2 = np.array(y_test)
160
161
162  # loop through the test set and generate predictions
163  with torch.no_grad():
164      for x, y in test_loader:
165          x = x.to('cpu')
166          y = y.to('cpu')
167          y_pred = net(x)
168          predicted = np.append(predicted, y_pred.cpu().detach().numpy())
169
170          actual2 = np.append(actual2, y.cpu().detach().numpy())
```

Figura B.11: Grafico de perdida,Evaluación de Modelo, creación de matrices vacías, llenado por iteración de las matrices vacías con los resultados de predicción, gráfico de valores reales vs predicciones.

Red_Volumen_critica_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > Red_Volumen_critica_VALIDATION

```
178   # plot the predicted and actual values for each output separately
179   plt.figure(figsize=(15,6))
180   plt.subplot(2, 1, 1)
181   plt.plot(actual2, label="Actual")
182   plt.plot(predicted, label="Predicted")
183   plt.legend()
184   plt.title("Predicted vs Actual for Acentric Factor ")
185   plt.xlabel("Sample")
186   plt.ylabel("Value")
187
188
189   # plot the residuals for each output
190   residuals = actual2 - predicted
191
192   plt.figure(figsize=(15, 6))
193   plt.subplot(2, 1, 1)
194   plt.scatter(predicted, residuals)
195   plt.title("Residual Plot for Acentric Factor")
196   plt.xlabel("Predicted")
197   plt.ylabel("Residual")
198   plt.axhline(y=0, color='r', linestyle='-')
199   from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
200
201   # Calculate MSE, RMSE, and MAE
202   mse = mean_squared_error(actual2, predicted)
203   rmse = np.sqrt(mse)
204   mae = mean_absolute_error(actual2, predicted)
205
206   # Calculate R-squared
207   r2 = r2_score(actual2, predicted)
208
209   print(f"MSE: {mse:.4f}")
210   print(f"RMSE: {rmse:.4f}")
211   print(f"MAE: {mae:.4f}")
212   print(f"R2: {r2:.4f}")
213
```

Figura B.12: Gráfico de Prediccion vs valores reales, Métricas de Rendimiento, Gráfico de Residuales.

Red_Volumen_critica_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > Red_Volumen_critica_VALIDATION.py > ...

```
214  # Obtener predicciones del modelo para el conjunto de prueba
215  with torch.no_grad():
216      net.eval()
217      predictions_vc = net(X_test)
218
219  # Convertir las predicciones y las etiquetas a arrays de NumPy
220  predictions_vc = predictions_vc.cpu().numpy().flatten()
221  y_test2 = y_test.cpu().numpy().flatten()
222
223  # Guardar las variables en un archivo
224  np.savez('variables_predicciones2.npz', predictions_vc=predictions_vc,y_test2=y_test2)
225  # Graficar el gráfico de calibración
226  plt.figure(figsize=(8, 6))
227  plt.plot(predictions_vc, y_test, 'o', label='Predicciones')
228  plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', label='Idealmente calibrado')
229  plt.xlabel('Predicciones')
230  plt.ylabel('Valores reales')
231  plt.title('Gráfico de Calibración')
232  plt.legend()
233  plt.show()
234
```

Figura B.13: Gráfico de Calibración, guardado de datos, y prueba de modelo.

B.3. Modelo RED-Tc: Propiedad Temperatura Critica

red_tc_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > red_tc_VALIDATION.py > ...

```python
import torch
import torch.nn as nn
import torch.optim as optim
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import os
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from torch.utils.data import DataLoader, TensorDataset
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
# Load data from csv file
df = pd.read_csv('C:\\Users\\nama_\\DownLoads\\wetransfer_tesis_2023-07-25_0023\\TESIS\\dataset_final_sorted_2.4.3_CSV_prueba_red.csv')

# Split data into training and testing sets
X = df[['Intensive_Entropy_Max','Theta_Inverse_Min', 'Theta_Inverse_Max', 'ODD_at_HOMO_Max',
        'ODD_at_LUMO_Max','Intensive_Atomic_Hypersoftness_Max','molecular volume',
        'thermophilicity_intensive_MAX', 'electrophilicity_intensive_MAX',
        'Dipole Moment', 'Intensive_Atomic_softness_Max','Pc(bar)']].values

y = df['Tc(K)'].values
# Split the data into training, validation, and test sets
X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.30, random_state=34)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.50, random_state=34)

# Scale the data to have zero mean and unit variance for input features (X)
sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_val = sc_X.transform(X_val)
X_test = sc_X.transform(X_test)
# Scale the data to have zero mean and unit variance for the target variable (y)
sc_y = StandardScaler()
y_train = sc_y.fit_transform(y_train.reshape(-1, 1))
y_val = sc_y.transform(y_val.reshape(-1, 1))
y_test = sc_y.transform(y_test.reshape(-1, 1))
```

Figura B.14: Carga de datos, selección variables de entrada y variable objetivo, importación de librerías, Creación de conjuntos Entrenamiento,Validación y Prueba, Función Scaler aplicada a valores de entrada y variable objetivo.

red_tc_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > red_tc_VALIDATION.py > ...

```
44  # Convert data to PyTorch tensors
45  X_train = torch.Tensor(X_train)
46  y_train = torch.Tensor(y_train)
47  X_val = torch.Tensor(X_val)
48  y_val = torch.Tensor(y_val)
49  X_test = torch.Tensor(X_test)
50  y_test = torch.Tensor(y_test)
51
52
53  # Create DataLoader objects for training, validation, and test data
54  train_data = TensorDataset(X_train, y_train)
55  val_data = TensorDataset(X_val, y_val)  # Create a DataLoader object for validation data
56  test_data = TensorDataset(X_test, y_test)
57
58  batch_size = 64*4
59  # Create DataLoader objects for training, validation, and test data
60  train_loader = DataLoader(train_data, batch_size=batch_size, shuffle=True,)
61  val_loader = DataLoader(val_data, batch_size=batch_size, shuffle=False, )
62  test_loader = DataLoader(test_data, batch_size=batch_size, shuffle=False, )
63
64  class Net(nn.Module):
65      def __init__(self):
66          super(Net, self).__init__()
67          self.fc1 = nn.Linear(12, 6)
68          self.fc2 = nn.Linear(6, 3)
69          self.fc3 = nn.Linear(3, 1)
70          self.relu = nn.ReLU()
71
72      def forward(self, x):
73          x = self.relu(self.fc1(x))
74          x = self.relu(self.fc2(x))
75          x = self.fc3(x)
76          return x
77
78  net = Net()
```

Figura B.15: Conversión de datos a tensores de PyTorch, Creación de DataLoaders para cada conjunto, Entrenamiento, Validación y Prueba, Arquitectura de Red

red_tc_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > red_tc_VALIDATION.py > ...

```
79
80  # Define the loss function and optimizer
81  criterion = nn.MSELoss()
82  optimizer = optim.Adam(net.parameters(), lr=0.001)
83
84  # Training loop
85  train_losses = []
86  val_losses = []
87  test_losses = []
88
89  for epoch in range(1500):
90      net.train()
91      train_loss = 0.0
92
93      for X_batch, y_batch in train_loader:
94          optimizer.zero_grad()
95          outputs = net(X_batch)
96          loss = criterion(outputs, y_batch)
97          loss.backward()
98          optimizer.step()
99          train_loss += loss.item() * X_batch.size(0)
100
101     train_loss /= len(train_loader.dataset)
102     train_losses.append(train_loss)
103
104 # Validation loop
105     with torch.no_grad():
106         net.eval()
107         val_loss = 0.0
108         for X_val_batch, y_val_batch in val_loader:
109             outputs_val = net(X_val_batch)
110             loss_val = criterion(outputs_val, y_val_batch)
111             val_loss += loss_val.item() * X_val_batch.size(0)
112
113     val_loss /= len(val_loader.dataset)
114     val_losses.append(val_loss)
```

Figura B.16: Definir función de perdida, función de optimización , tasa de aprendizaje, iteraciones del conjunto de datos de entrenamiento y validación.

red_tc_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > red_tc_VALIDATION.py >

```
115
116 # Test loop
117     with torch.no_grad():
118         net.eval()
119         test_loss = 0.0
120         for X_test_batch, y_test_batch in test_loader:
121             outputs_test = net(X_test_batch)
122             loss_test = criterion(outputs_test, y_test_batch)
123             test_loss += loss_test.item() * X_test_batch.size(0)
124
125     test_loss /= len(test_loader.dataset)
126     test_losses.append(test_loss)
127
128
129     if epoch % 10 == 0:
130         print('Epoch [{}/{}], Train Loss: {:.4f}, Val Loss: {:.4f}, Test Loss: {:.
131
132
133 # Plot the losses for the training set, validation set, and test set
134 plt.plot(train_losses, label='Training Loss',  )
135 plt.plot(val_losses, label='Validation Loss',linestyle = "--")
136 plt.plot(test_losses, label='Test Loss',  linestyle = "--")  # Add this line to pl
137 plt.xlabel('Epoch')
138 plt.ylabel('Loss')
139 plt.title('Training, Validation, and Test Losses Tc')
140 plt.legend()
141 plt.show()
142 # Obtain predictions for each output separately
143 predicted = outputs_test
144 # set the model to evaluation mode
145 net.eval()
146 with torch.no_grad():
147     predicted = net(X_test)
148     test_loss = criterion(predicted, y_test)
149
```

Figura B.17: Iteraciones conjunto de prueba, gráfico de función de perdida, ajustar modelo para evaluación.

red_tc_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > red_tc_VALIDATION.py > ...

```
150  # create empty arrays to store predicted and actual values
151  predicted = np.array(predicted)
152
153  actual2 = np.array(y_test)
154
155
156  # loop through the test set and generate predictions
157  with torch.no_grad():
158      for x, y in test_loader:
159          x = x.to('cpu')
160          y = y.to('cpu')
161          y_pred = net(x)
162          predicted = np.append(predicted, y_pred.cpu().detach().numpy())
163
164          actual2 = np.append(actual2, y.cpu().detach().numpy())
165
166
167  # calculate the mean squared error for each output
168  mse2 = np.mean(np.square(predicted - actual2))
169
170  print("Mean Squared Error for output 2: {:.4f}".format(mse2))
171
172
173
174  from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
175
176
177  # Deshacer la transformación de escalado en las etiquetas y_test
178  actual_unscaled = sc_y.inverse_transform(actual2.reshape(-1, 1))
179
180  # Deshacer la transformación de escalado en las predicciones (outputs_test)
181  predicted_unscaled = sc_y.inverse_transform(predicted.reshape(-1, 1))
```

Figura B.18: Creación de matrices vacías, rellenar matrices vacías con resultados, revertir función Scaler para posterior evaluación de resultados

red_tc_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > red_tc_VA

```
183    # plot the predicted and actual values for each output separately
184    plt.figure(figsize=(15,6))
185    plt.subplot(2, 1, 1)
186    plt.plot(actual_unscaled, label="Actual")
187    plt.plot(predicted_unscaled, label="Predicted")
188    plt.legend()
189    plt.title("Predicted vs Actual for Critical Temperature")
190    plt.xlabel("Sample")
191    plt.ylabel("Tc(K)")
192
193    # plot the residuals for each output
194    residuals = actual_unscaled - predicted_unscaled
195
196    plt.figure(figsize=(15, 6))
197    plt.subplot(2, 1, 1)
198    plt.scatter(predicted_unscaled, residuals)
199    plt.title("Residual Plot for Critical Temperature ")
200    plt.xlabel("Predicted")
201    plt.ylabel("Residual")
202    plt.axhline(y=0, color='r', linestyle='-')
203
204    # Resto del codigo...
205
206    # Calculate MSE, RMSE, and MAE
207    mse = mean_squared_error(actual2, predicted)
208    rmse = np.sqrt(mse)
209    mae = mean_absolute_error(actual2, predicted)
210
211    # Calculate R-squared
212    r2 = r2_score(actual2, predicted)
213
214    print(f"MSE: {mse:.4f}")
215    print(f"RMSE: {rmse:.4f}")
216    print(f"MAE: {mae:.4f}")
217    print(f"R2: {r2:.4f}")
```

Figura B.19: Gráfico valores reales vs predicciones, métricas de rendimiento, gráfica de residuales.

red_tc_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > red_tc_VALIDATION.py > ...

```
206  # Calculate MSE, RMSE, and MAE
207  mse = mean_squared_error(actual2, predicted)
208  rmse = np.sqrt(mse)
209  mae = mean_absolute_error(actual2, predicted)
210
211  # Calculate R-squared
212  r2 = r2_score(actual2, predicted)
213
214  print(f"MSE: {mse:.4f}")
215  print(f"RMSE: {rmse:.4f}")
216  print(f"MAE: {mae:.4f}")
217  print(f"R2: {r2:.4f}")
218  # Obtener predicciones del modelo para el conjunto de prueba
219  with torch.no_grad():
220      net.eval()
221      predictions_tc = net(X_test)
222
223  # Convertir las predicciones y las etiquetas a arrays de NumPy
224  predictions_tc = predictions_tc.cpu().numpy().flatten()
225  y_test6 = y_test.cpu().numpy().flatten()
226
227  # Guardar las variables en un archivo
228  np.savez('variables_predicciones6.npz', predictions_tc=predictions_tc,y_test6=y_test6)
229  # Graficar el gráfico de calibración
230  plt.figure(figsize=(8, 6))
231  plt.plot(predictions_tc, y_test, 'o', label='Predicciones')
232  plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', label='Idealmente calibrado')
233  plt.xlabel('Predicciones')
234  plt.ylabel('Valores reales')
235  plt.title('Gráfico de Calibración')
236  plt.legend()
237  plt.show()
238
```

Figura B.20: Gráfico de calibración. guardado de datos y evaluación del modelo en conjunto de prueba.

B.4. Modelo RED-Dc: Propiedad Densidad Critica

Este es el contenido introductorio de la sección.

Red_Critical_Density_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > Red_Critical_Density_VALIDATION.py > ...

```
# Import the required libraries
import torch
import torch.nn as nn
import torch.optim as optim
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from torch.utils.data import DataLoader, TensorDataset
# Load data from csv file
df = pd.read_csv('C:\\Users\\nama_\\DownLoads\\wetransfer_tesis_2023-07-25_0023\\TESIS\\dataset_final_sorted_2.4.3_CSV_prueba_red.csv')

# Split data into training and testing sets

X = df[['Theta_Inverse_Min', 'ODD_at_HOMO_Max',
        'ODD_at_LUMO_Max', 'Theta_Inverse_Max',
        'thermophilicity_intensive_MAX',
        'electrophilicity_intensive_MAX', 'Intensive_Entropy_Max','Intensive_Atomic_Hypersoftness_Max']].values

y = df['RHOC(g/cm3)'].values
# Split the data into training, validation, and test sets
X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.30, random_state=34)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.50, random_state=34)
# Scale the data to have zero mean and unit variance
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_val = sc.transform(X_val)
X_test = sc.transform(X_test)
# Convert data to PyTorch tensors
X_train = torch.Tensor(X_train)
y_train = torch.Tensor(y_train)
X_val = torch.Tensor(X_val)
y_val = torch.Tensor(y_val)
X_test = torch.Tensor(X_test)
y_test = torch.Tensor(y_test)
```

Figura B.21: Importación de librerías,carga de datos, creación de conjunto entrenamiento, validación y prueba, funcion Scaler en los datos, conversion de los datos a tensores de Pytorch.

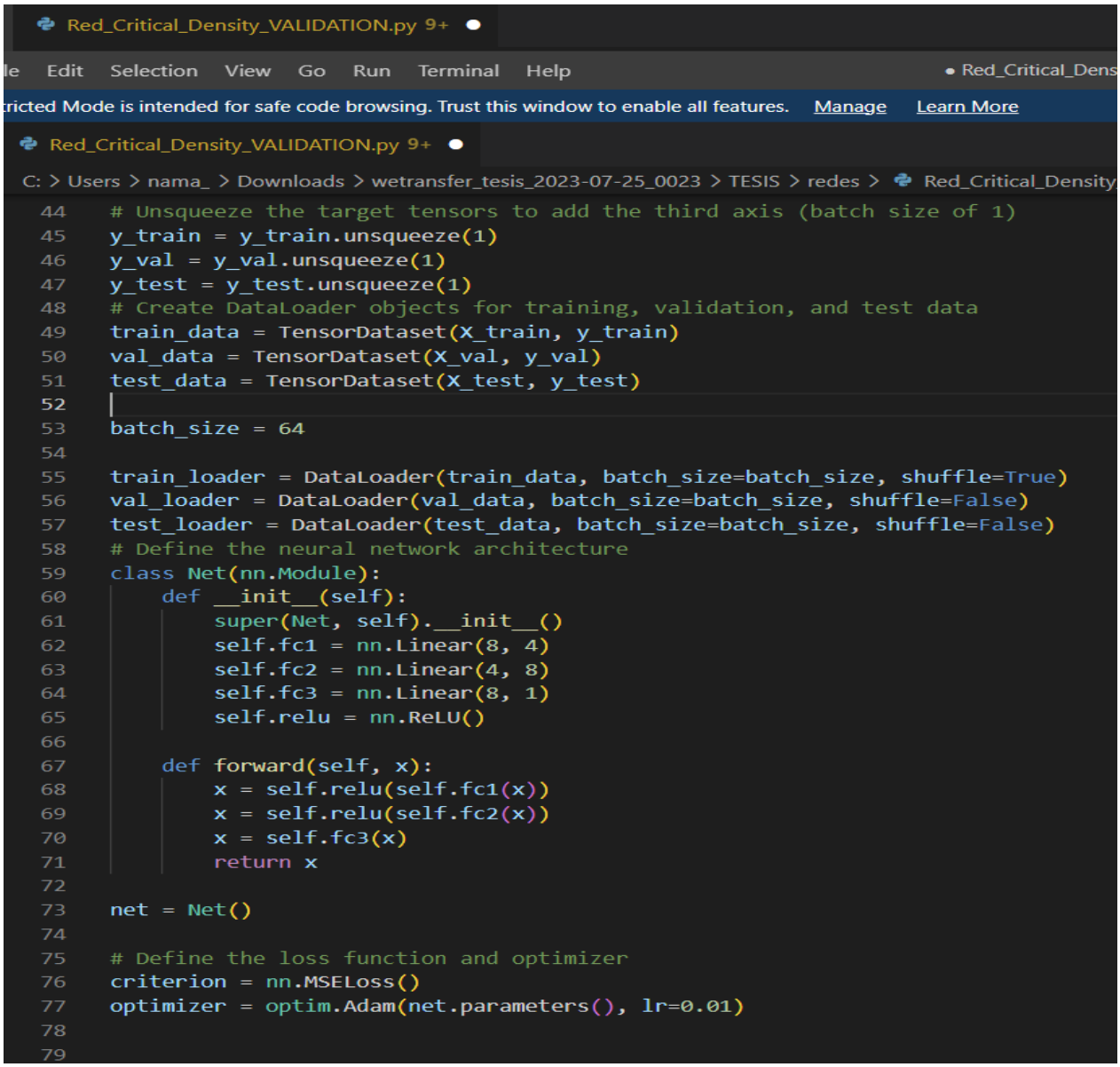

Figura B.22: Ajuste de medidas de tensor variable objetivo, creación de Dataloaders para manejo de datos, arquitectura de red , tasa de aprendizaje, optimizador y funcion de perdida.

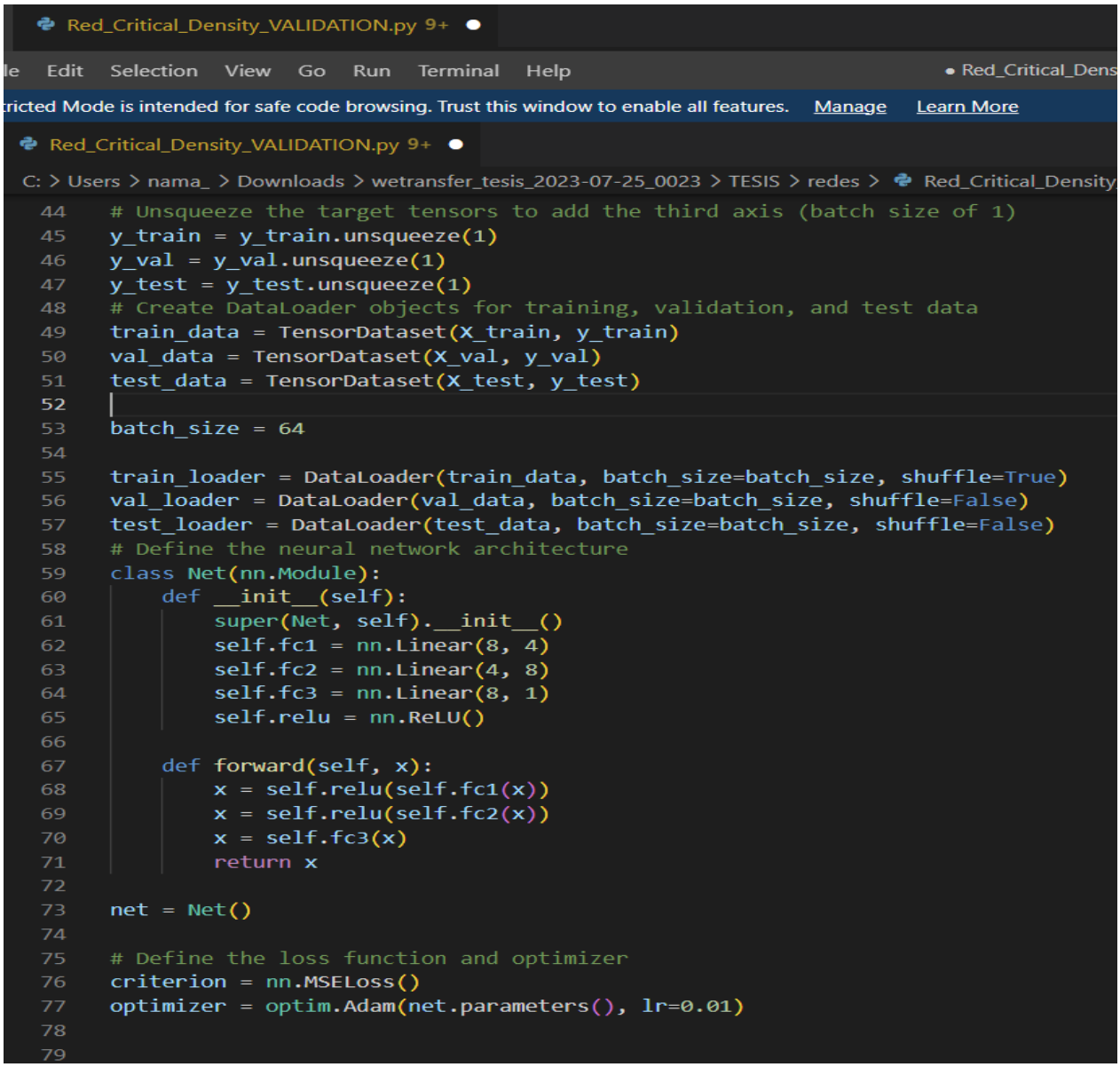

Figura B.23: Ajuste de medidas de tensor variable objetivo, creación de Dataloaders para manejo de datos, arquitectura de red , tasa de aprendizaje, optimizador y función de perdida.

Red_Critical_Density_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > Red_Critical_De

```
78
79
80  # Training loop
81  train_losses = []
82  val_losses = []
83  test_losses = []
84
85  for epoch in range(1000):
86      net.train()
87      train_loss = 0.0
88
89      for X_batch, y_batch in train_loader:
90          optimizer.zero_grad()
91          outputs = net(X_batch)
92          loss = criterion(outputs, y_batch)
93          loss.backward()
94          optimizer.step()
95          train_loss += loss.item() * X_batch.size(0)
96
97      train_loss /= len(train_loader.dataset)
98      train_losses.append(train_loss)
99
100     # Validation loop
101     with torch.no_grad():
102         net.eval()
103         val_loss = 0.0
104         for X_val_batch, y_val_batch in val_loader:
105             outputs_val = net(X_val_batch)
106             loss_val = criterion(outputs_val, y_val_batch)
107             val_loss += loss_val.item() * X_val_batch.size(0)
108
109         val_loss /= len(val_loader.dataset)
110         val_losses.append(val_loss)
111
112
```

Figura B.24: Iteraciones para conjunto de entrenamiento y validación.

Red_Critical_Density_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > Red_Critical_Density_VALIDATION.py > ...

```
    # Test loop
    with torch.no_grad():
        net.eval()
        test_loss = 0.0
        for X_test_batch, y_test_batch in test_loader:
            outputs_test = net(X_test_batch)
            loss_test = criterion(outputs_test, y_test_batch)
            test_loss += loss_test.item() * X_test_batch.size(0)

        test_loss /= len(test_loader.dataset)
        test_losses.append(test_loss)

    if epoch % 1 == 0:
        print('Epoch [{}/{}], Train Loss: {:.4f}, Val Loss: {:.4f}, Test Loss: {:.4f}'.format(epoch + 1, 1000, train_loss, val_loss, test_loss))

# ... (continue with the code)

# Plot the losses for the training set, validation set, and test set
import matplotlib.pyplot as plt

plt.plot(train_losses, label='Training Loss',  )
plt.plot(val_losses, label='Validation Loss',linestyle = "--")
plt.plot(test_losses, label='Test Loss',  linestyle = "--")  # Add this line to plot the test loss
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.title('Training, Validation, and Test Losses RHOc')
plt.legend()
plt.show()
# Obtain predictions for each output separately
predicted = outputs_test

# set the model to evaluation mode
net.eval()
with torch.no_grad():
    predicted = net(X_test)
    test_loss = criterion(predicted, y_test)
```

Figura B.25: iteraciones conjunto de prueba, grafico de perdida, evaluacion del modelo.

Red_Critical_Density_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > Red_Critical_Density_VALIDATIO

```
154   # create empty arrays to store predicted and actual values
155   predicted = np.array(predicted)
156
157   actual2 = np.array(y_test)
158
159
160   # loop through the test set and generate predictions
161   with torch.no_grad():
162       for x, y in test_loader:
163           x = x.to('cpu')
164           y = y.to('cpu')
165           y_pred = net(x)
166           predicted = np.append(predicted, y_pred.cpu().detach().numpy())
167
168           actual2 = np.append(actual2, y.cpu().detach().numpy())
169
170
171   # calculate the mean squared error for each output
172   mse2 = np.mean(np.square(predicted - actual2))
173
174   print("Mean Squared Error for output 2: {:.4f}".format(mse2))
175
176
177   # plot the predicted and actual values for each output separately
178   plt.figure(figsize=(15,6))
179   plt.subplot(2, 1, 1)
180   plt.plot(actual2, label="Actual")
181   plt.plot(predicted, label="Predicted")
182   plt.legend()
183   plt.title("Predicted vs Actual for Critical Density")
184   plt.xlabel("Sample")
185   plt.ylabel("Value")
186
```

Figura B.26: Creación de matrices vacías, llenado de matrices con los resultados, Gráfico valores reales vs predicciones

Red_Critical_Density_VALIDATION.py 9+

C: > Users > nama_ > Downloads > wetransfer_tesis_2023-07-25_0023 > TESIS > redes > Red_Critical_Density_VALIDATION.py > ...

```
187
188  # plot the residuals for each output
189  residuals = actual2 - predicted
190
191  plt.figure(figsize=(15, 6))
192  plt.subplot(2, 1, 1)
193  plt.scatter(predicted, residuals)
194  plt.title("Residual Plot for Critical Density")
195  plt.xlabel("Predicted")
196  plt.ylabel("Residual")
197  plt.axhline(y=0, color='r', linestyle='-')
198  from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
199
200  # Calculate MSE, RMSE, and MAE
201  mse = mean_squared_error(actual2, predicted)
202  rmse = np.sqrt(mse)
203  mae = mean_absolute_error(actual2, predicted)
204
205  # Calculate R-squared
206  r2 = r2_score(actual2, predicted)
207
208  print(f"MSE: {mse:.4f}")
209  print(f"RMSE: {rmse:.4f}")
210  print(f"MAE: {mae:.4f}")
211  print(f"R2: {r2:.4f}")
212  # Obtener predicciones del modelo para el conjunto de prueba
213  with torch.no_grad():
214      net.eval()
215      predictions_dc = net(X_test)
216
217  # Convertir las predicciones y las etiquetas a arrays de NumPy
218  predictions_dc = predictions_dc.cpu().numpy().flatten()
219  y_test1 = y_test.cpu().numpy().flatten()
220
221  # Guardar las variables en un archivo
222  np.savez('variables_predicciones1.npz', predictions_dc=predictions_dc,y_test1=y_test1)
```

Figura B.27: Gráfico de residuales, prueba de modelo con conjunto de prueba, métricas de rendimiento y guardado de resultados

```
# Graficar el gráfico de calibración
plt.figure(figsize=(8, 6))
plt.plot(predictions_dc, y_test, 'o', label='Predicciones')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', label='Idealmente calibrado')
plt.xlabel('Predicciones')
plt.ylabel('Valores reales')
plt.title('Gráfico de Calibración')
plt.legend()
plt.show()
```

Figura B.28: Gráfico de Calibración

B.5. Modelo RED-Pc: Propiedad presión critica

Para consultar el código fuente de este modelo consultar el siguiente enlace: https://github.com/Dan07619/REDES_TESIS_PTC

B.6. Modelo RED-OMEGA: Propiedad factor acentrico

Para consultar el código fuente de este modelo entrar al siguiente enlace: https://github.com/Dan07619/REDES_TESIS_PTC

Bibliografía

1. Span, R. *Multiparameter equations of state: an accurate source of thermodynamic property data* (Springer Science & Business Media, 2000).

2. Wilhelmsen, Ø. *et al.* Thermodynamic modeling with equations of state: present challenges with established methods. *Industrial & Engineering Chemistry Research* **56,** 3503-3515 (2017).

3. Canuto, V. Equation of state at ultrahigh densities. *Annual Review of Astronomy and Astrophysics* **12,** 167-214 (1974).

4. Melosh, H. A hydrocode equation of state for SiO2. *Meteoritics & Planetary Science* **42,** 2079-2098 (2007).

5. Tsonopoulos, C. y Heidman, J. L. From Redlich-Kwong to the present. *Fluid phase equilibria* **24,** 1-23 (1985).

6. Soave, G. Equilibrium constants from a modified Redlich-Kwong equation of state. *Chemical engineering science* **27,** 1197-1203 (1972).

7. Zudkevitch, D. y Joffe, J. Correlation and prediction of vapor-liquid equilibria with the redlich-kwong equation of state. *AIChE Journal* **16,** 112-119 (1970).

8. Ding-Yu, P. y Donald, B. R. A new two-constant equation of state. *Industrial & Engineering Chemistry Fundamentals* **15,** 59-64 (1976).

9. Gurvich, L. V. Reference books and data banks on the thermodynamic properties of individual substances. *Pure and Applied Chemistry* **61,** 1027-1031 (1989).

10. Sengers, J. y Sengers, J. L. Thermodynamic behavior of fluids near the critical point. *Annual Review of Physical Chemistry* **37,** 189-222 (1986).

11. Rogers, G. F. C. y Mayhew, Y. R. *Thermodynamic and transport properties of fluids* (John Wiley & Sons, 1995).

12. Cengel, Y. A., Boles, M. A. y Kanoğlu, M. *Thermodynamics: an engineering approach* (McGraw-hill New York, 2011).

13. Zemansky, M. W. y Dittman, R. H. *Heat and thermodynamics* 1998.

14. Wan, S. The Determination of Critical Temperature from Index of Refraction. *The Journal of Physical Chemistry* **45,** 903-907 (1941).

15. Rossini, F. Pure Compounds from Petroleum. *Analytical Chemistry* **20,** 110-121 (1948).

16. Booth, H. S. y Swinehart, C. F. The Critical Constants and Vapor Pressures at High Pressure of Some Gaseous Fluorides of Group IV1. *Journal of the American Chemical Society* **57,** 1337-1342 (1935).

17. Ashour, I., Al-Rawahi, N., Fatemi, A. y Vakili-Nezhaad, G. Applications of equations of state in the oil and gas industry. *Thermodynamics-kinetics of dynamic systems* **1,** 165-178 (2011).

18. Kontogeorgis, G. M. y Tassios, D. P. Critical constants and acentric factors for long-chain alkanes suitable for corresponding states applications. A critical review. *Chemical Engineering Journal* **66,** 35-49 (1997).

19. Kontogeorgis, G. M. y Folas, G. K. *Thermodynamic models for industrial applications: from classical and advanced mixing rules to association theories* (John Wiley & Sons, 2009).

20. Patzek, T. W. Thermodynamics of the corn-ethanol biofuel cycle. *Critical Reviews in plant sciences* **23,** 519-567 (2004).

21. De Hemptinne, J.-C. *et al.* A view on the future of applied thermodynamics. *Industrial & Engineering Chemistry Research* **61,** 14664-14680 (2022).

22. Smith, J. M., Van Ness, H. C., Abbott, M. M. y García, C. R. *Introducción a la termodinámica en ingeniería química* **660.296 9 S724i 2003.** (McGraw-Hill, 2007).

23. Van Vleck, J. H. The correspondence principle in the statistical interpretation of quantum mechanics. *Proceedings of the National Academy of Sciences* **14,** 178-188 (1928).

24. Aggarwal, C. C. *et al.* Neural networks and deep learning. *Springer* **10,** 3 (2018).

25. Mahesh, B. Machine learning algorithms-a review. *International Journal of Science and Research (IJSR).[Internet]* **9,** 381-386 (2020).

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

27. Goodfellow, I., Bengio, Y. y Courville, A. *Deep learning* (MIT press, 2016).

28. Raschka, S., Liu, Y. H., Mirjalili, V. y Dzhulgakov, D. *Machine Learning with PyTorch and Scikit-Learn: Develop machine learning and deep learning models with Python* (Packt Publishing Ltd, 2022).

29. Jensen, F. *Introduction to computational chemistry* (John wiley & sons, 2017).

30. Szabo, A. y Ostlund, N. S. *Modern quantum chemistry: introduction to advanced electronic structure theory* (Courier Corporation, 2012).

31. Fischer, C. F. Hartree–Fock method for atoms. A numerical approach (1977).

32. Hohenberg, P. y Kohn, W. Inhomogeneous electron gas. *Physical review* **136,** B864 (1964).

33. Kohn, W. y Sham, L. J. Self-consistent equations including exchange and correlation effects. *Physical review* **140,** A1133 (1965).

34. Koch, W. y Holthausen, M. C. *A chemist's guide to density functional theory* (John Wiley & Sons, 2015).

35. Blanchard, P., Brüning, E., Blanchard, P. y Brüning, E. Density functional theory of atoms and molecules. *Mathematical Methods in Physics: Distributions, Hilbert Space Operators, Variational Methods, and Applications in Quantum Physics,* 563-573 (2015).

36. Politzer, P. y Seminario, J. M. *Modern density functional theory: a tool for chemistry* (Elsevier, 1995).

37. Ayers, P. W. y Parr, R. G. Variational principles for describing chemical reactions: the Fukui function and chemical hardness revisited. *Journal of the American Chemical Society* **122,** 2010-2018 (2000).

38. Geerlings, P., De Proft, F. y Langenaeker, W. Conceptual density functional theory. *Chemical reviews* **103,** 1793-1874 (2003).

39. Pearson, R. G. Absolute electronegativity and hardness: application to inorganic chemistry. *Inorganic chemistry* **27,** 734-740 (1988).

40. Liu, S., Zhang, X. *et al.* Chemical concepts from density functional theory. *Acta Phys. Chim. Sin* **34,** 563-566 (2018).

41. Franco-Pérez, M. The electronic temperature and the effective chemical potential parameters of an atom in a molecule. A Fermi–Dirac semi-local variational approach. *Physical Chemistry Chemical Physics* **24,** 807-816 (2022).

42. Franco-Pérez, M., Gázquez, J. L., Ayers, P. W. y Vela, A. Revisiting the definition of the electronic chemical potential, chemical hardness, and softness at finite temperatures. *The Journal of chemical physics* **143** (2015).

43. Goh, G. B., Hodas, N. O. y Vishnu, A. Deep learning for computational chemistry. *Journal of computational chemistry* **38,** 1291-1307 (2017).