JOAQUÍN LUQUE RODRÍGUEZ

SINFONÍA DE LOS DATOS

Una mirada desde la ingeniería

Lección inaugural leída en la solemne apertura
del curso académico 2026-2027
en la Universidad de Sevilla

Sinfonía de los datos

Joaquín Luque Rodríguez

Sinfonía de los datos
Una mirada desde la ingeniería

Lección inaugural leída en la solemne apertura
del curso académico 2026-2027
en la Universidad de Sevilla

Editorial Universidad de Sevilla

Sevilla 2026

Colección: Textos institucionales
Núm.: 134

© Editorial Universidad de Sevilla 2026
Porvenir, 27 - 41013 Sevilla.
Tlfs.: 954 487 447; 954 487 451
Correo electrónico: info-eus@us.es
Web: https://editorial.us.es

© Joaquín Luque Rodríguez 2026

Maquetación y realización electrónica:
Editorial Universidad de Sevilla

A mis padres,
de los que no soy más que un tibio eco.

A mi mujer,
cascabel sonoro de la fantasía.

A mis hijos,
armoniosas melodías propias.

A mis nietos,
notas vibrantes de mi particular *Gaudeamus*.

Contenido

Excma. Sra. rectora magnífica de la Universidad de Sevilla;
Excmas. e Ilmas. autoridades;
Claustro de profesorado,
Personal Técnico, de Gestión y de Administración y Servicios;
Estudiantes;
Familiares, amigas, amigos;
Señoras y señores:

Afinación

Desde la revolución copernicana del siglo XVI y la consolidación del método científico del XVII, el dato se ha constituido como la base del conocimiento y, eventualmente, como el pilar de la ingeniería moderna. Su estatus ontológico ha sido discutido a lo largo de la historia del pensamiento. Por un lado, el empirismo, desde Francis Bacon[1] hasta el positivismo lógico[2], veía el dato como algo «puro e inmaculado», algo «dado» por la naturaleza, la base objetiva sobre la cual, por inducción, construimos leyes. Por otro, la filosofía de la ciencia del siglo XX, con Popper[3], Kuhn[4] y Hanson[5], demostró que el dato puro no existe; que todo dato está «cargado de teoría». No es este el escenario ni yo el actor adecuado para discutir enfoques más recientes como los de Chris Anderson[6], que entroniza al dato

dando por «muerta la teoría», o los de Bruno Latour[7] y Rob Kitchin[8], que ponen de manifiesto que el dato no se encuentra en la naturaleza y nos es «dado», sino que es capturado mediante un complejo entramado socio-técnico: un dato solo existe porque hay una red de presupuestos, instituciones, estándares y decisiones de diseño de ingeniería que lo sostienen.

Si bien desde el siglo XVII la ciencia abraza con entusiasmo el nuevo método y pone al dato como fuente y origen de su comprensión del mundo, la ingeniería del momento seguía siendo un oficio gremial fuertemente basado en reglas empíricas y la intuición acumulada, a menudo desconectada de la ciencia de vanguardia[9]. No es hasta la Revolución Industrial del siglo XIX cuando termina de cuajar la verdadera «ingeniería guiada por datos y modelos matemáticos», la ingeniería moderna, proceso al que contribuyeron de manera decisiva la creación de las Escuelas Politécnicas, un movimiento del que fue pionera la École Polytechnique en Francia en 1794[10]. En España tuvimos que esperar hasta 1850 para la creación de las primeras Escuelas Industriales de Madrid, Barcelona, Guipúzcoa y Sevilla[11], de cuya institución la actual Escuela Politécnica Superior se siente orgullosa heredera[12].

Al final de este proceso de modernización, también la ingeniería acaba teniendo en el dato, como reclamaba Arquímedes[13], ese punto de apoyo en el que hacer palanca

para mover el mundo. Y precisamente de ese humilde dato es de lo que quiero hablarles en esta lección inaugural. Me gustaría invitarles a una breve excursión que nos permita ir descubriendo juntos cómo es posible pasar de un simple dato, o un puñado de ellos, a los sorprendentes resultados de la Inteligencia Artificial[1] (IA), un término, por cierto, que parece moderno pero que fue acuñado hace ahora justo 70 años en la ya famosa Conferencia de Dartmouth de 1956, considerada el origen de la disciplina[14]. Un camino en el que pretendo desvelar el «truco» de magia detrás de estos programas cuyos resultados nos sorprenden a todos, incluso a los supuestos especialistas.

A la Inteligencia Artificial cabe acercarse desde múltiples perspectivas. Podemos hacerlo desde las matemáticas, definiendo los algoritmos que trabajan detrás de estos desarrollos[15]; desde la economía, midiendo su coste, sus retornos, su impacto en la productividad y el empleo[16];

1. La inmensa mayoría de los sistemas actuales de Inteligencia Artificial están basados en un uso masivo de datos que son utilizados para entrenar modelos en un proceso denominado Aprendizaje Automático o, más técnicamente, Aprendizaje Estadístico. Sin embargo, ello no agota el campo de la IA; existen otras ramas (actualmente muy minoritarias) basadas en la lógica y el cumplimiento de reglas (la IA simbólica), que engloban enfoques como los sistemas expertos, las ontologías formales o la programación lógica, y que, por razones de espacio, no tendrán cabida en este documento.

desde el derecho, analizando y desarrollando normativas que regulen, pongan orden y protejan a la sociedad de posibles excesos[17]; desde la filosofía, analizando el estatus metafísico de los asistentes conversacionales[18]; o, incluso, desde la religión, como pone de manifiesto la reciente encíclica de León XIV[19]. En esta lección, por formación y por el Centro al que represento hoy aquí, les voy a proponer una mirada desde la ingeniería. Y esto, en un doble sentido: dando, por un lado, unas pinceladas de cómo funciona «técnicamente»; y analizando, por otro, qué utilidad tiene en la ingeniería, proponiéndoles ejemplos simples en distintas de sus ramas.

En 1950 Alan Turing propuso un test ya clásico, conocido como «el juego de la imitación», para discernir si una máquina «piensa»[20]. Bajo su planteamiento, una máquina superaría ese test cuando sus respuestas fuesen indistinguibles de las de un humano. Si bien la prueba en sí ha sido ampliamente debatida y contestada desde numerosas ópticas, quizás la más potente de ellas formulada por Searle en 1980[21], el test de Turing no ha perdido su vigencia y su atractivo. Pues bien, además de la experiencia personal que todos ustedes, con seguridad, tienen en la interacción con estos nuevos programas, un reciente estudio de 2025, publicado en mayo de este mismo año[22], ha contrastado de forma rigurosa que los Grandes Modelos de Lenguaje actuales han superado ya el «juego de la imitación». ¿Cómo es posible que una máquina, cuanto menos,

parezca «pensar»? ¿Cómo pueden desempeñar funciones intelectuales un puñado de datos y unos algoritmos? A lo largo de esta lección trataré también de arrojar algo de luz sobre esa pregunta.

Hasta ahora hemos hablado del dato, en singular, pero un dato aislado apenas tiene utilidad. Un dato sería como una nota musical: aislada, poco o nada nos dice. Es cuando se combina con otras cuando es capaz de constituirse en melodía. Haciendo paráfrasis del clásico refrán español, «un dato no hace modelo, pero ayuda al compañero».

Y con esta analogía entre dato y nota, inauguro una metáfora musical que pretendo me ayude a exponer en términos comprensibles por un público culto pero heterogéneo los distintos aspectos de esta lección. Una metáfora que pretendo aprovechar en cuatro direcciones:

a) En primer lugar, ilustrando cómo el incremento en el número de datos/notas no es solo cuantitativo, sino que tiene importantes consecuencias cualitativas.
b) Por otro lado, el sugerirles que contemplen los conjuntos de datos como una sinfonía me permitirá dividir la exposición en cuatro movimientos que irán marcando el desarrollo de la obra, en nuestro caso, los distintos niveles de complejidad, desde la humilde recta de regresión a los grandes modelos de lenguaje.
c) Y en el empeño de mirar el tema desde la ingeniería aprovecharé cada uno de los movimientos para usar

un ejemplo de cuatro de los grados en Ingeniería de nuestra Escuela: Diseño, Química, Mecánica y Electricidad, reservándole al quinto grado, el de Ingeniería Electrónica Industrial (el que más cercano me resulta), el papel de *luthier*, de constructor de los instrumentos (sensores, procesadores, algoritmos...) que hacen posible la sinfonía[2].

d) Por último, para tratar de enriquecer aún más la metáfora, les invito a que asimilen el desarrollo de la lección con el despliegue de la Novena Sinfonía de Beethoven, una obra cuyo arco narrativo se resume a la perfección bajo el lema de *per aspera ad astra* («por el sendero áspero, a las estrellas» o «a través del esfuerzo, hacia el éxito»)[23]. Ello nos permitirá descubrir analogías con los tempos y motivos de los sucesivos movimientos de la obra y, finalmente, descubrir resonancias comunes entre música y datos.

Termino ya el tiempo de afinación. La orquesta está preparada y, espero y deseo, el público listo para la audición. Comenzamos.

---

2. La elección del orden de presentación de los distintos grados se debe a razones puramente expositivas: nada tiene que ver con su nivel de complejidad o con las preferencias personales del autor.

Primer movimiento

Allegro ma non troppo, un poco maestoso. El conflicto (las *aspera*)

En el primer movimiento del desarrollo de esta sinfonía de los datos empezamos casi desde la nada, con una sensación de caos al que paulatinamente trataremos de poner orden. La música es tensa, inestable, incluso violenta. No hay una melodía reconfortante: todo parece lucha y búsqueda. Esto encarna esfuerzo, dificultad, incertidumbre, conflicto interior y exterior. No hay todavía «camino», solo resistencia.

Pero pasemos de las musas al teatro. Menos ensayo y más concierto. Y para nuestra mirada desde la ingeniería, contemplemos este primer movimiento, tomando un ejemplo muy simplificado de la Ingeniería en Diseño Industrial y Desarrollo del Producto[3].

3. Todos los ejemplos presentados a lo largo de la lección están basados en problemas de ingeniería reales, aunque, por razones didácticas,

Imaginemos un ingeniero en una empresa dedicada al desarrollo de mobiliario escolar ergonómico (mesas, sillas y estaciones de trabajo ajustables). Sus clientes son colegios y administraciones públicas que necesitan equipar aulas para estudiantes de entre 3 y 18 años.

Uno de los principales problemas que enfrentan es que las aulas suelen equiparse con un número limitado de tallas estándar, lo que provoca posturas inadecuadas en muchos estudiantes con la consiguiente incomodidad, fatiga, y riesgo de problemas musculoesqueléticos a largo plazo. Para mejorar esta situación necesita predecir el crecimiento de los estudiantes desde que comenzaron su escolarización (medido en altura corporal) con el objetivo de diseñar mejor las dimensiones de mesas y sillas, optimizar la proporción de tallas que fabrican, y reducir costes logísticos (menos devoluciones o ajustes). En la mayoría de los casos no se dispone de otro dato para predecir la talla que los años de escolarización del estudiante[4].

---

han sido simplificados en mayor o menor grado. Los datos que se utilizan en los ejemplos han sido generados de forma sintética, habiéndose procurado que mantengan un carácter verosímil.

4. Preferimos hablar de años de escolarización y no de edad, y de crecimiento acumulado (desde el comienzo de la escolarización) y no de altura, para que el primero y más sencillo de los modelos empiece en el origen de coordenadas.

En este caso tan simple (más bien, deliberadamente simplificado) tenemos ya, en germen, todos los elementos que requerirán los más complejos sistemas de Inteligencia Artificial[5]. En primer lugar, dispondremos de un conjunto de ejemplos o muestras (los estudiantes en este caso) que asumimos que son representativos del conjunto de la población[6]. Cada una de estas muestras, cada uno de estos estudiantes, queda descrito por:

— Un valor que se desea predecir, en este caso el crecimiento desde que comenzaron su escolarización, que denominamos el «objetivo»[7].

---

5. Nos centraremos aquí y en el resto del documento en el Aprendizaje Supervisado, el enfoque más extendido dentro del Aprendizaje Automático. Otras variantes abarcan el Aprendizaje No Supervisado (orientado a descubrir estructuras ocultas en datos sin etiquetar), el Aprendizaje por Refuerzo (basado en la optimización de acciones mediante recompensas), y variantes híbridas de gran relevancia industrial como los modelos semisupervisados, autosupervisados o por transferencia.

6. Si los datos de entrenamiento no son estadísticamente representativos de la población real, las predicciones subsiguientes estarán sesgadas y limitadas a las particularidades de dicha muestra. Este es un riesgo crítico en los sistemas de Inteligencia Artificial del que se debe ser consciente y que es preceptivo evitar[52].

7. Desde un punto de vista más general, existen problemas en los que son varios los valores que deben predecirse simultáneamente. Por simplicidad de la exposición, dichos problemas, denominados «multiobjetivo», serán obviados durante la mayor parte de esta lección y solo en el último movimiento haremos mención de ellos.

— Un conjunto de valores, denominados atributos o características, que se utilizarán como pistas para realizar la predicción. En el ejemplo, cada estudiante se caracteriza por un único atributo: el tiempo (los años) de escolarización.

Para obtener un modelo que responda a las necesidades de este caso se utiliza una metodología, que será extensible a los más complejos sistemas de IA. Esta metodología consta de seis etapas:

1. Muestreo

En general es imposible el acceso a los datos del conjunto de la población[8] (el conjunto de todos los estudiantes). Por tanto, nuestro primer esfuerzo debe ir dirigido a obtener unas muestras de unos cuantos estudiantes que sean representativos del total de estudiantes, potenciales clientes de la empresa. Imaginemos que, después de este esfuerzo de recolección, disponemos de una muestra de 100 estudiantes. A continuación, separamos aleatoriamente esta muestra en dos conjuntos diferentes: uno de 80 estudiantes que

8. Además, en caso de disponer de los datos del conjunto de la población, dejaría de tener sentido construir un modelo predictivo de una información que ya poseemos.

utilizaremos para entrenamiento del modelo, y otro de 20 estudiantes para evaluación[9] del modelo resultante.

2. Caracterización de las muestras

Cada una de las muestras, cada estudiante, es ahora descrito por un atributo (los años de escolarización), que denotaremos como x, y por un valor objetivo[10] (el crecimiento acumulado), que denotaremos como y. En la figura 1 representamos los valores del crecimiento frente a años de escolarización para las muestras de entrenamiento.

3. Formulación de hipótesis

A continuación, formulamos una hipótesis que denotaremos como h, sobre la relación que existe entre los atributos y el objetivo, entre la x y la y. En este primer movimiento elegiremos la más simple de las hipótesis: la línea recta. Es decir, asumimos que el crecimiento (el cambio de talla) es

9. Este conjunto suele denominarse conjunto de datos de evaluación o de prueba.

10. En este y ulteriores ejemplos, los valores de atributos y objetivos serán considerados números reales. Es posible, e incluso frecuente, que dichos valores sean números enteros o variables categóricas. Por simplicidad omitiremos tales casos, lo que no resta generalidad a los fundamentos de la explicación.

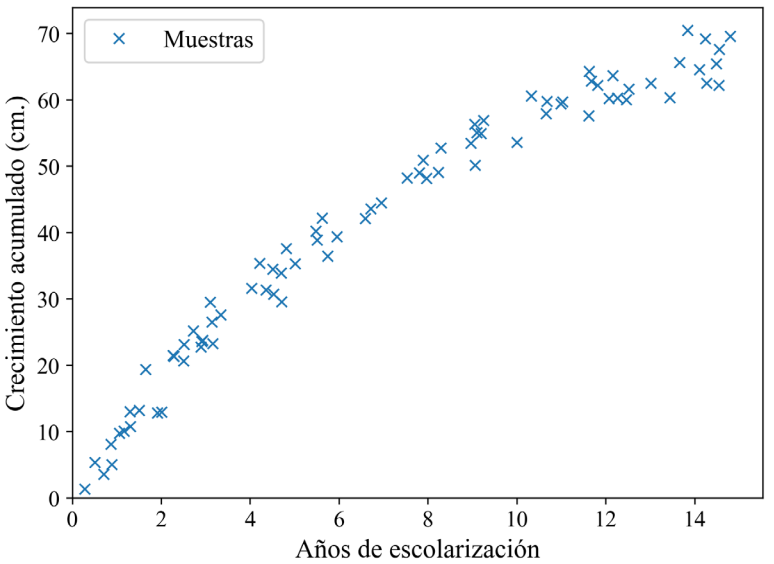

Figura 1. Representación gráfica de las muestras de entrenamiento

directamente proporcional al tiempo de escolarización: a más años, más crecimiento. Si denotamos como w a la constante de proporcionalidad (la pendiente de la recta), la hipótesis queda formulada por la expresión $y = w \cdot x$. Al depender la hipótesis de un parámetro se prefiere denotar como h_w y se formula mediante la expresión

$$y = h_w(x) = w \cdot x \tag{1}$$

En la figura 2 representamos un ejemplo de una hipótesis lineal correspondiente a un valor de la pendiente $w = 6$,

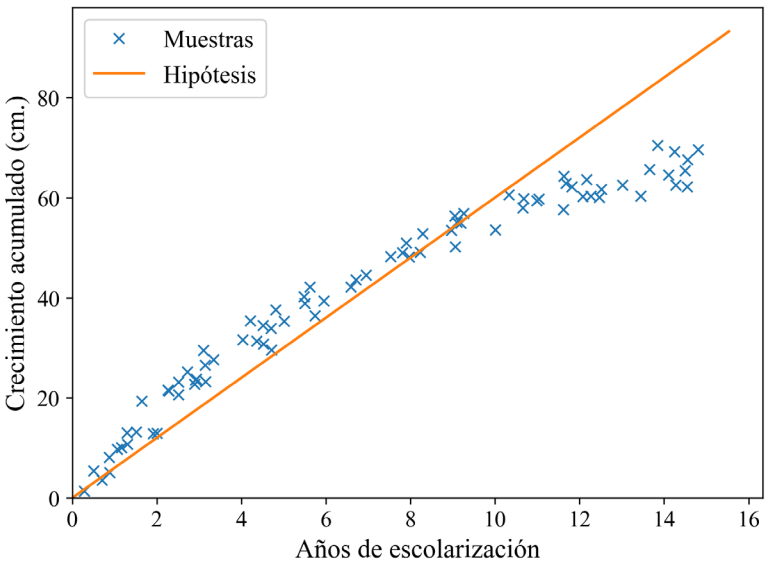

Figura 2. Ejemplo de hipótesis lineal (con $w = 6$)

es decir, nuestra hipótesis inicial es que el estudiante crece 6 cm por curso.

4. Selección de la función de pérdida

En la etapa anterior vimos que el modelo depende de un parámetro: la pendiente de la recta. La adecuada elección del valor de ese parámetro nos dará el mayor o menor éxito en la predicción, en definitiva, el éxito o el fracaso del modelo. ¿Cómo elegimos el «mejor» valor del parámetro? El primer paso para ello consiste en definir cómo medimos la bondad

de un parámetro. Y esta medida de la idoneidad del parámetro es lo que denominamos la función de pérdida.

Para el ejemplo que venimos desarrollando vamos a fijarnos en primer lugar en un estudiante concreto de la muestra, con 10 años de escolarización y un crecimiento acumulado durante ese periodo de 54 cm (figura 3). Definiremos tentativamente la pérdida asociada a ese estudiante (lo mal que funciona en su caso la hipótesis elegida) como la diferencia entre el valor real del crecimiento (marcada con una cruz) y el valor que predice la hipótesis (marcada con un círculo), es decir, como el error de predicción,

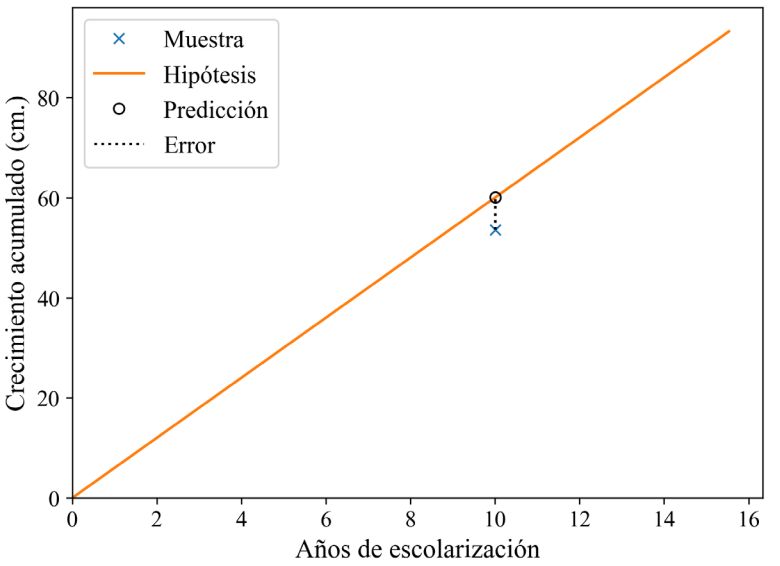

Figura 3. Pérdida asociada a un estudiante

$e(w) = y - h_w(x)$, que puede tomar tanto valores positivos como negativos. Con la hipótesis inicial de que los estudiantes crecen 6 cm por curso, la predicción de crecimiento para ese estudiante es de 60 cm, es decir, 6 cm más del valor real. Este valor es el error de predicción asociado a ese estudiante. Pero, para medir la bondad de una hipótesis, tan malo es pasarse en la predicción como no llegar, por lo que preferimos redefinir la pérdida de la hipótesis en el caso de esa muestra como el cuadrado del error de predicción[11], es decir, $e^2(w) = [y - h_w(x)]^2$. En la muestra elegida el error cuadrático será, por tanto, de 36 cm^2.

Esta pérdida, asociada a un único individuo, la podemos generalizar al caso de todos los estudiantes en el subconjunto de entrenamiento[12] sin más que tomar el valor medio de cada una de las pérdidas individuales, es decir,

$$J(w) = \frac{1}{n_e}\sum_{i=1}^{n_e} e_i^2(w) = \frac{1}{n_e}\sum_{i=1}^{n_e}[y_i - h_w(x_i)]^2 = \frac{1}{n_e}\sum_{i=1}^{n_e}[y_i - w \cdot x_i]^2 \quad (2)$$

11. Se podría haber definido la pérdida como el valor absoluto del error, pero se prefiere usar el cuadrado porque esta función de error es derivable, mientras la primera no lo es en el origen (cuando el error es cero).

12. El cálculo de la función de pérdida y su posterior optimización se realizan tomando exclusivamente las muestras del subconjunto de entrenamiento, es decir, los 80 estudiantes de nuestro ejemplo ($n_e = 80$).

tal como se representa en la figura 4. En nuestro caso esa pérdida es de 86.57 cm^2.

Así definida, la función de pérdida nos indica cuán mala es una hipótesis para el conjunto de datos de la muestra de entrenamiento. Distintas pendientes de la recta, es decir, distintos valores del parámetro w, producirán mejores o peores hipótesis, tal como puede observarse intuitivamente en la figura 5.

Si trazamos el valor de la pérdida de la hipótesis en función de la pendiente, obtenemos un resultado como el mostrado en la figura 6.

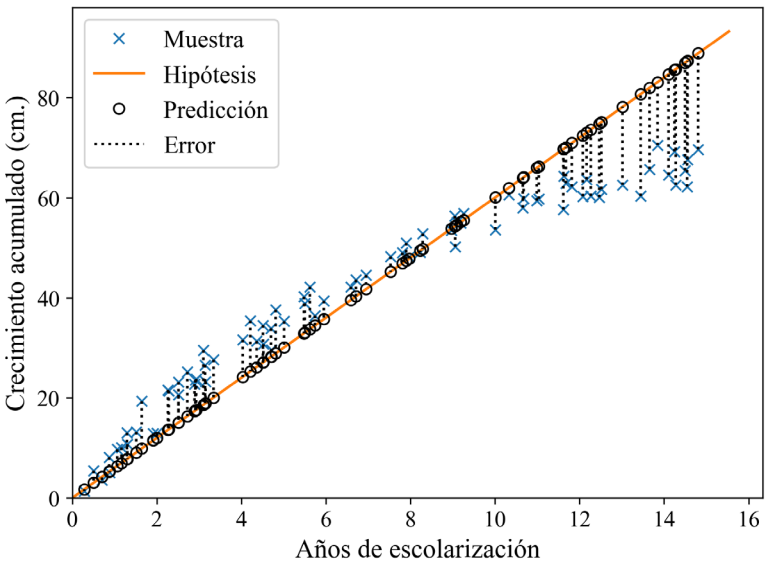

Figura 4. Pérdida asociada a todos los estudiantes del conjunto de datos de entrenamiento

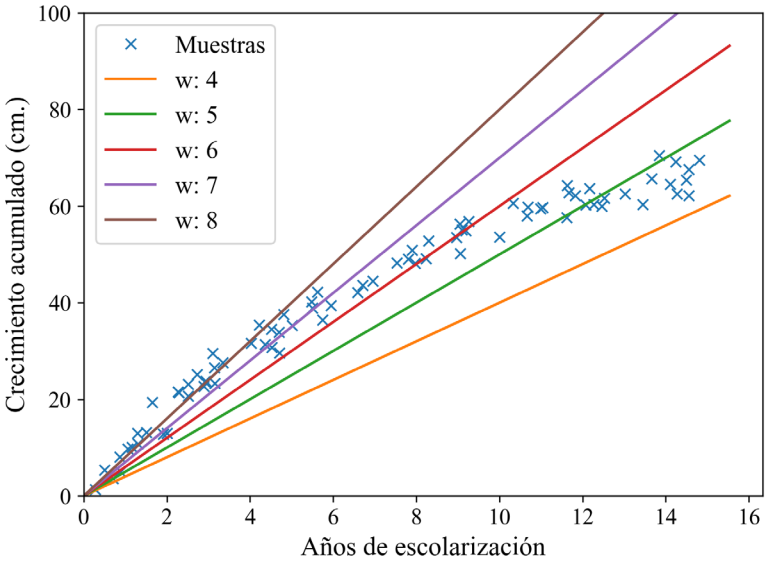

Figura 5. Ejemplo de varias hipótesis lineales usando distintos valores del parámetro (rectas de distinta pendiente)

5. Optimización de la función de pérdida

Resulta evidente de la explicación anterior que la siguiente etapa será encontrar aquel valor del parámetro que produce la mínima pérdida. Es decir, se trata de obtener cuánto vale el parámetro w en la parte más baja de la curva mostrada en la figura 6. En este caso tan sencillo, el valor óptimo de w puede obtenerse de forma analítica mediante el denominado método de los mínimos cuadrados[24]. Sin embargo, en casos más complejos, como aquellos a los que

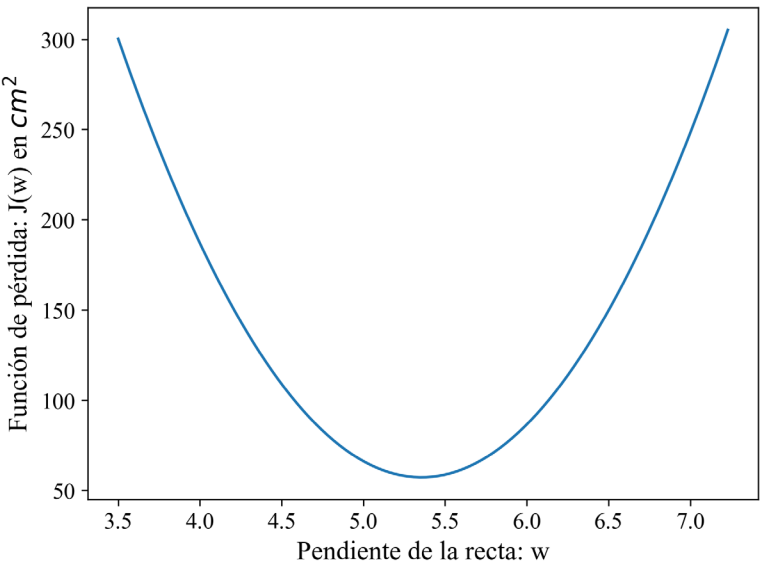

Figura 6. Pérdida de la hipótesis en función del valor del parámetro (de la pendiente de la recta)

nos enfrentaremos en los siguientes movimientos de esta sinfonía, no existe siempre una forma analítica de encontrar el valor óptimo del parámetro. Por ello, se prefiere recurrir a métodos numéricos tales como el método del gradiente descendente o derivados[25].

Pero no se asusten. Tras este nombre tan esotérico, se esconde una idea fácil de entender. Inicialmente se toma un valor de la pendiente de la recta elegida aleatoriamente, digamos que $w = 7$, lo que corresponde al punto A en la figura 7. Como en ese punto la función de pérdida crece

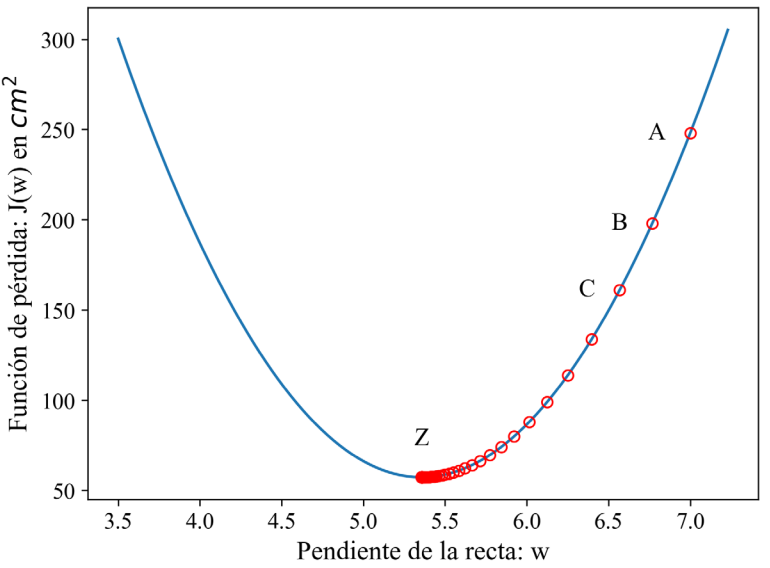

Figura 7. Optimización de la función de pérdida mediante el algoritmo del gradiente descendente

al aumentar w (es decir, tiene pendiente $p_J(w)$ positiva[13]), lo que hacemos es dar un paso en sentido contrario, es decir, hacemos decrecer w para disminuir la pérdida. ¿Cuánto?: una cantidad proporcional a la pendiente de la pérdida, es decir,

13. No confundir la pendiente de la función de pérdida con la pendiente de la hipótesis. La función de pérdida es una parábola, por tanto, con pendiente variable, mientras que la hipótesis es una recta, por tanto, con pendiente constante.

$$\Delta w = -\alpha \cdot p_J(w) = -\alpha \frac{dJ(w)}{dw}, \qquad (3)$$

expresión[14] en la que α es la constante de proporcionalidad, denominada tasa de aprendizaje[15].

Después de dar este primer paso llegamos al punto B de la figura 7, en el que $w = 6.77$. De nuevo miramos la pendiente de la pérdida en ese nuevo punto, y disminuimos el valor de w en una cantidad proporcional a la pendiente de la pérdida en B. Si este proceso se repite iterativamente, acaba alcanzándose un valor óptimo de w (punto Z de la figura 7) para el que la pérdida es la mínima posible. En nuestro ejemplo este mínimo se alcanza para el valor óptimo $w = 5.36$, con una pérdida asociada de $J = 57.28\text{cm}^2$.

6. Evaluación del modelo

Con los pasos anteriores ya hemos obtenido un modelo que resuelve el problema de ingeniería planteado. En el último

14. El signo negativo en la expresión indica que, si la pendiente es positiva, es decir, la pérdida crece cuando w crece, entonces el parámetro debe disminuir, lo que implica que $\Delta w < 0$.

15. La tasa de aprendizaje α constituye un claro ejemplo de lo que técnicamente se conoce como un hiperparámetro del modelo. En siguientes movimientos de la sinfonía aparecerán también otros hiperparámetros a los que nos referiremos en su momento, así como los posibles métodos para su elección.

paso tratamos de evaluar cómo de bueno es el modelo. Es decir, cómo se va a comportar cuando tenga que predecir el crecimiento de nuevos estudiantes, cuyos datos no ha visto antes, es decir, no han sido utilizados para entrenar el modelo. Es ahora cuando cobra sentido la separación que hicimos en la etapa de muestreo. El conjunto de 20 datos de evaluación que entonces apartamos y que en ningún momento hemos utilizado nos servirán ahora para evaluar el resultado.

Esta separación es crucial para mitigar el riesgo de sobreajuste (*overfitting*): si evaluáramos el modelo con los mismos datos con los que fue entrenado, correríamos el riesgo de validar un algoritmo que simplemente ha memorizado las particularidades de su muestra de entrenamiento. Al enfrentarlo a este conjunto «ajeno» y completamente nuevo, ponemos a prueba su verdadera capacidad de generalización.

Podemos asimilar este proceso al del profesor que prepara una colección de 100 problemas resueltos. Publica 80 de ellos, con sus soluciones, para que los estudiantes puedan practicar (se puedan *entrenar*), y reserva otros 20 para evaluar su nivel de aprendizaje. El día del examen plantea los problemas sin resolver, obtiene las respuestas de los estudiantes y compara las respuestas con la solución correcta.

Evaluar un modelo con los mismos datos de entrenamiento sería el equivalente académico a darle a un alumno las preguntas del examen el día antes de clase. El estudiante podría sacar un sobresaliente simplemente memorizando las

respuestas, sin comprender realmente la materia. Al reservar un conjunto de datos de evaluación que el modelo «nunca ha visto», le estamos obligando a enfrentarse a un examen con preguntas nuevas. Solo así sabremos si ha aprendido a resolver el problema subyacente o si solo ha memorizado el camino.

Y en la comparación entre el valor real del crecimiento (solución correcta del problema) y el valor del crecimiento estimado por el modelo (solución suministrada por el estudiante), ¿qué métrica (qué criterios) debemos utilizar? Podríamos pensar en primer lugar en aplicar la propia función de pérdida, el error cuadrático medio[16], a los datos de evaluación. En nuestro ejemplo esto nos da una pérdida de $57.28cm^2$. ¿Es eso un resultado bueno o malo? ¿Estamos ante un buen o un mal modelo? No es fácil decirlo. La función de pérdida está elegida para que el optimizador de los parámetros (el algoritmo del gradiente descendente) funcione cómoda y adecuadamente, no para que sea fácilmente interpretable por un humano. Por ello es habitual recurrir a otras métricas más centradas en las personas. Usando una de estas métricas[17], el coeficiente de determinación R^2, podemos

16. Es más habitual usar la raíz cuadrada de la función de pérdida, es decir, el denominado RMSE (Root Mean Square Error), ya que este último se encuentra en las unidades que queremos predecir, por ejemplo, en centímetros de altura.

17. A lo largo de la presente lección usaremos, para evaluar los modelos, el coeficiente de determinación R^2. Esta métrica, una de las

decir que el modelo ha obtenido una calificación de 8.12 sobre 10, lo que equivaldría a un notable alto, algo que todos podemos identificar como un buen resultado.

Con estas seis etapas hemos construido nuestro primer modelo. Su funcionamiento se esquematiza en la figura 8: los años de escolarización de un estudiante (x) se multiplican por el parámetro w para obtener una predicción de su crecimiento (y).

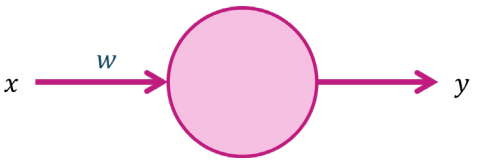

Figura 8. Esquema del modelo lineal

La hipótesis que les he propuesto para este primer modelo, una recta que pasa por el origen, puede llegar a ser en muchos casos excesivamente simplista. Por ello, sin

---

más ampliamente utilizadas en la evaluación de modelos de regresión, mide las prestaciones obtenidas en comparación con el resultado de un modelo trivial: aquel que siempre predice que la salida es igual a la media, independientemente de la entrada. El coeficiente de determinación, por lo general, oscila entre cero (las predicciones tienen el mismo error que si se usara la media) y uno (predicción perfecta). No obstante, es teóricamente posible que tome valores negativos si el modelo produce más errores que dicha media. El modelo del ejemplo ha obtenido un $R^2 = 0.812$, que en el texto traducimos como un 8.12 sobre 10 para asimilarlo a las calificaciones académicas tradicionales.

abandonar la sencillez de la línea recta, podemos adoptar una hipótesis en la que la recta ya no tiene necesariamente que pasar por el origen, tal como se representa en la figura 9.

En este caso, necesitamos dos parámetros para definir la recta: su pendiente y el punto de corte del eje vertical, denominado sesgo. Los denotaremos como w_1 y w_0 respectivamente, por lo que la hipótesis toma la forma

$$y = h_w(x) = w_1 x + w_0 \tag{4}$$

La función de pérdida depende ahora de dos parámetros, es decir,

$$J(w) = \frac{1}{n_e}\sum_{i=1}^{n_e}[y_i - h_w(x_i)]^2 = \frac{1}{n_e}\sum_{i=1}^{n_e}[y_i - (w_1 x_i + w_0)]^2 \tag{5}$$

por lo que su representación gráfica toma la forma de una superficie en tres dimensiones, tal como aparece reflejado en la figura 10.

Para encontrar los valores de los parámetros que hacen mínima la pérdida podemos extender el mismo procedimiento que describimos anteriormente: vamos recorriendo la superficie dando pasos en la dirección de bajada hasta encontrar el valle. Con este procedimiento encontramos que la hipótesis lineal óptima tiene una pendiente $w_1 = 4.85$ y un sesgo $w_0 = 5.03$, lo que produce una pérdida $J = 33.05$.

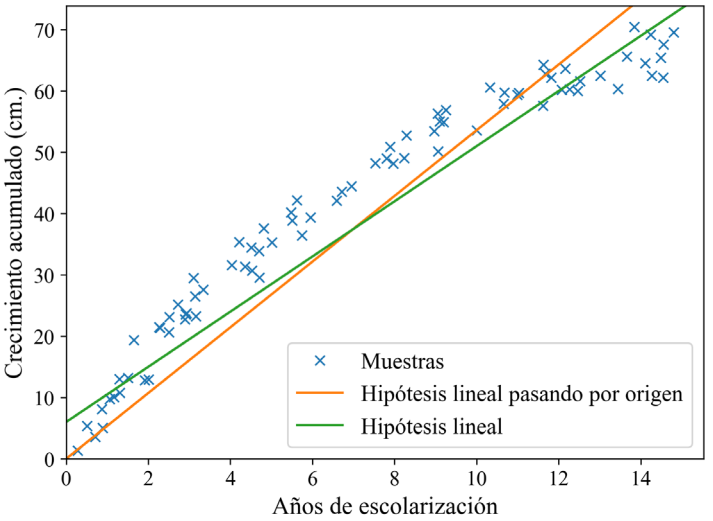

Figura 9. Ejemplo de dos hipótesis lineales (pasando o no por el origen)

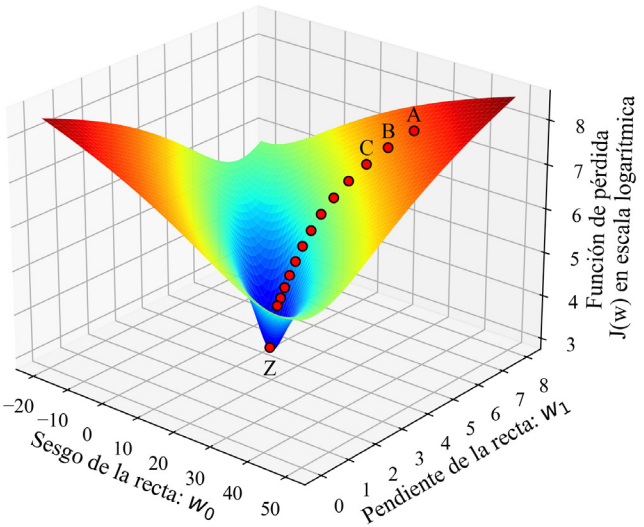

Figura 10. Función de pérdida para una hipótesis con dos parámetros

La posterior evaluación del modelo nos ofrece un resultado de 8.94[18], ligeramente mejor que cuando usamos un único parámetro y ya rozando el sobresaliente.

El funcionamiento de este modelo mejorado se esquematiza en la figura 11: los años de escolarización de un estudiante (x) se multiplican por el parámetro w_1 y al resultado se le suma el parámetro w_0 para obtener una predicción de su crecimiento (y). Este módulo tan elemental, al que denominaremos combinador lineal, será el elemento más simple de lo que luego desarrollaremos en complejas redes neuronales, pero cuya base se encuentra ya en este humilde elemento de computación.

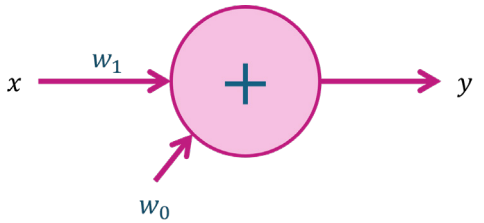

Figura 11. Esquema del modelo lineal mejorado: combinador lineal o recta de regresión

Muchos de ustedes habrán notado que el proceso que acabo de describir en este primer movimiento no es más que

18. Un R^2 de 0.894.

la obtención de la recta de regresión[19], una herramienta estadística ampliamente utilizada no solo en ciencias naturales e ingeniería sino también en diferentes ámbitos de la salud, las ciencias sociales e incluso las humanidades. ¿Es esto Inteligencia Artificial? Aunque nos parezca muy simple, la respuesta es rotundamente afirmativa. Desarrollaremos y justificaremos esta respuesta en los próximos movimientos.

Si volvemos a nuestra metáfora musical, en la melodía que acabamos de componer (el modelo lineal) con unas pocas decenas de notas (datos) encontramos ya la semilla que germinará en las más complejas sinfonías. Es una melodía tipo «Cumpleaños feliz»: compuesta por unas 25 notas, simple, fácil de entender, conocida por todos, pero, aun así, fuertemente evocadora.

Un breve silencio para dejar respirar la música y continuamos.

19. El combinador lineal es la recta de regresión cuando han sido optimizados sus parámetros.

Segundo movimiento

Molto vivace. Presto. La lucha activa

Arranca con un *scherzo* que es puro impulso. Ritmo obsesivo, casi mecánico (los timbales destacan mucho). Sensación de energía combativa, como avanzar a golpes. No hay paz, pero sí dirección. Aquí ya no estamos perdidos: estamos luchando activamente contra las dificultades. Es el momento de enfrentarse al destino.

En el conflicto que tratamos de resolver en el movimiento anterior, es probable que nos quedara una cierta sensación de incompletitud. El modelo lineal es cómodo, pero quizás demasiado simplificador de la realidad. Para encarar esta situación, tomaremos un ejemplo de otro de nuestros grados: la Ingeniería Química Industrial.

Asumamos el rol de una ingeniera en una almazara, donde se produce aceite de oliva virgen extra mediante procesos de extracción exclusivamente mecánicos. Durante

la etapa de batido de la pasta de aceituna, la temperatura influye de forma compleja (no lineal) en fenómenos físicos y bioquímicos como la coalescencia de gotas de aceite, la viscosidad del medio y la actividad enzimática. Asimismo, la velocidad de rotación de los álabes del batidor acelera la coalescencia, pero un exceso de velocidad rompe la emulsión y destruye los componentes volátiles.

En este escenario, necesitamos desarrollar un modelo que permita predecir el tiempo óptimo de batido en función de la temperatura y de la velocidad de giro, con el objetivo de maximizar el rendimiento de extracción sin comprometer la calidad del aceite.

Diferentes procesos extractivos (batidas) realizados en condiciones controladas por operarios expertos nos proporcionan un conjunto de datos en el que cada muestra queda descrita por un par de atributos de entrada: la temperatura a la que se realiza el proceso extractivo, que denotaremos como T, y la velocidad de rotación del batidor, que denotaremos como w; y por un valor objetivo, el tiempo óptimo de batido[20], denotado como t. Al igual que en el caso anterior,

20. El tiempo óptimo asociado a cada temperatura y velocidad de rotación se obtiene a partir de experimentos de laboratorio en los que se realizan procesos extractivos de duración creciente. Para cada temperatura y velocidad, se construye la curva rendimiento-tiempo, definiéndose el tiempo óptimo como el menor tiempo que alcanza un

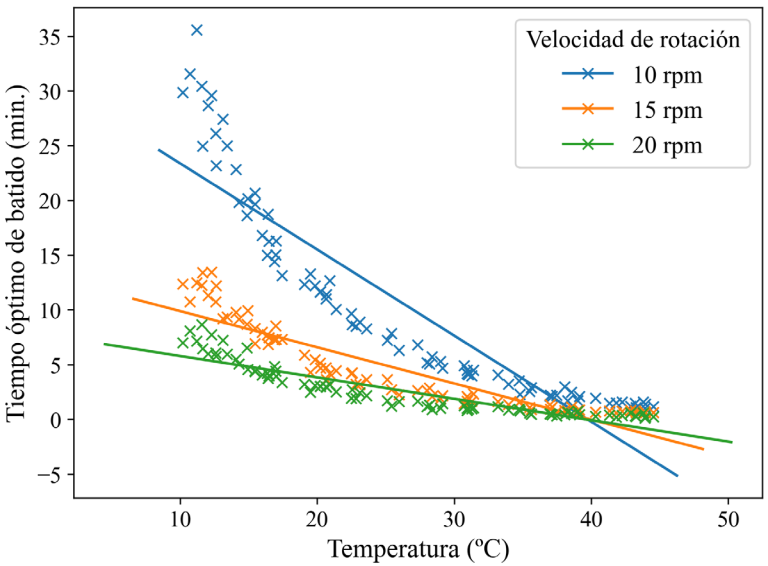

Figura 12. Representación gráfica de las muestras y el modelo lineal resultante

se dispone de 100 muestras, de las cuales se reservan 20 para evaluación del modelo y 80 para entrenamiento.

Repitiendo las seis etapas que describimos en el primer movimiento, encontramos un resultado como el reflejado en la figura 12 donde representamos los valores del tiempo óptimo frente a la temperatura para tres velocidades de rotación diferentes, así como el modelo lineal

---

rendimiento superior a un umbral prefijado, típicamente del 90 % del rendimiento máximo observado.

correspondiente (recta de regresión). Podemos observar en el gráfico que el modelo ofrece un resultado mediocre, lo que se refleja en una calificación[21] de 6.37 sobre 10.

La razón de tan pobres prestaciones radica principalmente en que el modelo asume linealidad entre la temperatura, la velocidad de rotación y el tiempo óptimo, cuando en realidad el proceso fisicoquímico subyacente es fuertemente no lineal.

Para abordar esta cuestión recurriremos a lo que se denomina la ingeniería de características: en vez de usar directamente los valores de temperatura y tiempo óptimo para describir cada muestra, construiremos otros valores diferentes basados en ellos. Y en este segundo movimiento, esa construcción se basará en el conocimiento y experiencia de la ingeniera responsable del modelado. Ella sabe que la relación entre temperatura en grados Kelvin (T), velocidad de rotación (w) y tiempo (t) sigue en primera aproximación una ecuación de tipo Arrhenius[26] de la forma,

$$\frac{1}{t} = A\, T^{B} \omega^{C} e^{-\frac{D}{T}} \qquad (6)$$

en la que los términos A, B, C y D son constantes empíricas características de cada proceso, cuyos valores dependen de las propiedades reológicas de la pasta (variedad, madurez) y de las características de diseño de la batidora.

21. $R^2 = 0.637$.

La ecuación (6) nos da la pista de cómo transformar los valores originales. Para ello tomemos logaritmos naturales en ambos lados de la ecuación, obteniendo

$$-\ln t = \ln A + B \ln T + C \ln \omega - \frac{D}{T}. \qquad (7)$$

A la luz de esta ecuación, la ingeniería de características nos lleva a describir cada muestra por tres atributos (el logaritmo de la temperatura, la inversa de la temperatura y el logaritmo de la velocidad de rotación), que denotaremos respectivamente como x_1, x_2 y x_3, así como por un valor objetivo (el logaritmo del tiempo óptimo), que denotaremos como z. Su definición formal es la siguiente:

$$z = \ln t\,, \qquad x_1 = \ln T\,, \qquad x_2 = \frac{1}{T}, \qquad x_3 = \ln \omega. \qquad (8)$$

La dinámica de la reacción queda así transformada como

$$z = -\ln A - Bx_1 + Dx_2 - Cx_3, \qquad (9)$$

o lo que es lo mismo,

$$z = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_3, \qquad (10)$$

es decir, una ecuación lineal con tres atributos y cuatro parámetros.

Una vez caracterizadas las muestras con los atributos y el objetivo transformados según la ecuación (8), y formulada la hipótesis lineal, el resto de etapas del proceso es idéntico al enumerado en el primer movimiento: selección del error cuadrático medio como función de pérdida, optimización de dicha función mediante el algoritmo del gradiente descendente y evaluación final del modelo sobre el conjunto de datos de evaluación. El resultado de este proceso se refleja en la figura 13 en la que podemos observar una muy buena predicción del modelo, lo que se traduce en una calificación[22] de 9.91 sobre 10: un excelente resultado.

El funcionamiento de este modelo no lineal se esquematiza en la figura 14: la temperatura y la velocidad de rotación de una muestra (T, ω) se transforman mediante ingeniería de características en tres atributos (x_1, x_2, x_3) que se multiplican por los parámetros correspondientes (w_1, w_2, w_3), sumándole posteriormente el valor del parámetro w_0 para obtener una predicción del logaritmo del tiempo óptimo (z). Finalmente, a este valor se le aplica una transformación exponencial, que se denomina función de activación, para obtener la predicción del tiempo óptimo (t).

El gráfico anterior puede esquematizarse aún más. Y en concreto, el combinador lineal y la función de activación pueden agruparse en un único nodo, tal como aparece

22. $R^2 = 0.991$.

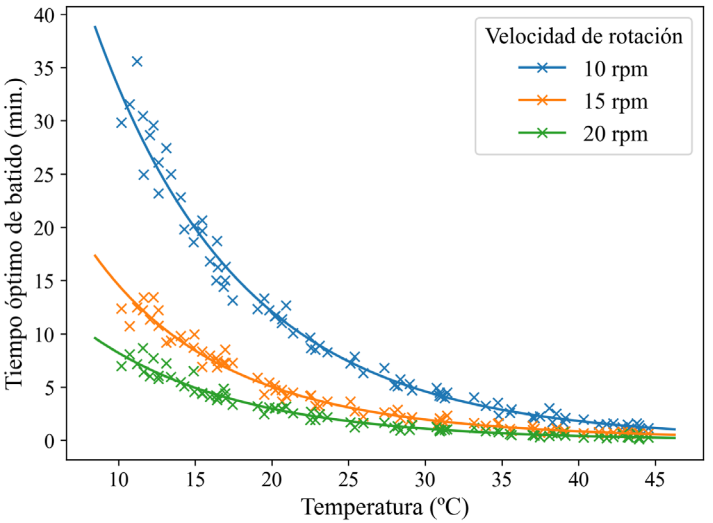

Figura 13. Representación gráfica de las muestras y el modelo resultante aplicando transformación mediante ingeniería de características

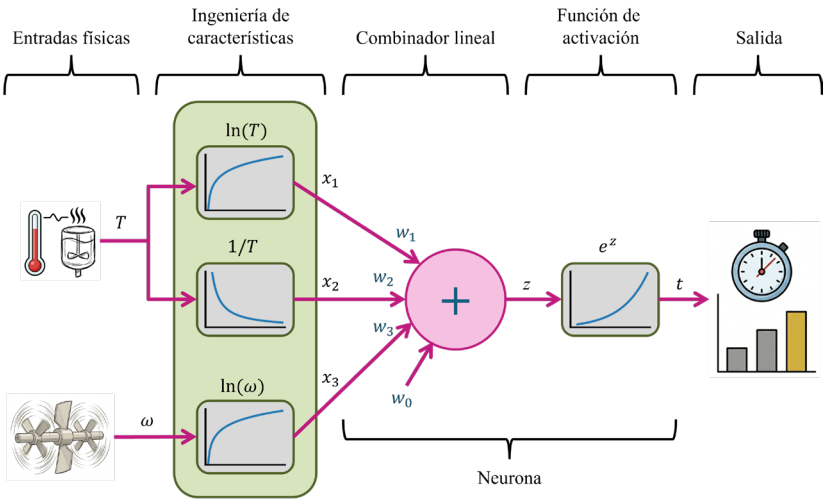

Figura 14. Esquema detallado del modelo no lineal

en la figura 15. Dicho nodo de cálculo constituye lo que se denomina... ¡una neurona!

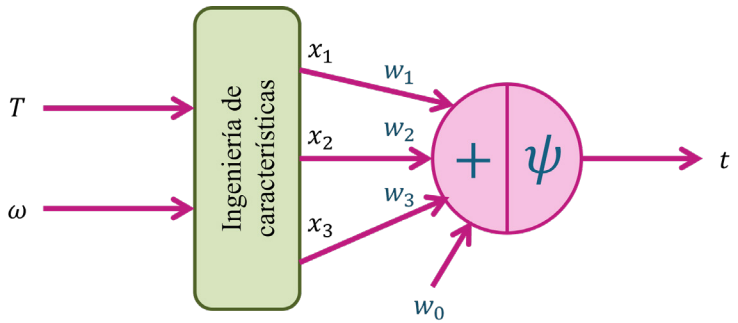

Figura 15. Esquema simplificado del modelo no lineal

Si tuviéramos que asociar este ejemplo a una melodía, ya no podría tener la simplicidad de una cancioncilla. Partimos de dos variables de entrada, que hemos tenido que descomponer en tres para resolver la no linealidad del problema, y al combinador lineal le hemos tenido que añadir una función adicional que hemos bautizado como «función de activación». Permítanme que les proponga asimilarla al *Para Elisa* de Beethoven, una pieza para un único instrumento, pero que ya cuenta con unas 1000 notas y, sin duda, un nivel de complejidad mayor que el *Cumpleaños Feliz*.

Y tras el golpe de efecto del final de este movimiento, presentando nuestra primera neurona, un nuevo respiro y continuamos.

Tercer movimiento

Adagio molto e cantábile.
La contemplación (esperanza)

De repente, todo cambia: música lenta, profundamente lírica y serena. Aparece una belleza casi «fuera del tiempo». Es introspectiva, como mirar hacia dentro. Este es el primer atisbo de «las estrellas»: no hemos llegado aún, pero intuimos que existen. Es la esperanza, la visión de algo mejor.

El movimiento anterior estuvo lleno de frenética actividad. Es lo que ocurre cuando nos enfrentamos a problemas no lineales y tenemos que buscar cómo transformar las características en otras que tengan una relación lineal. La Inteligencia Artificial clásica exige que nuestra ingeniera en la almazara haya tenido que dedicar cerca del 80 % del tiempo a este proceso manual, traduciendo la intuición física y sus conocimientos científico-técnicos en variables sintéticas.

En este tercer movimiento, abordaremos el problema desde una nueva perspectiva, más serena, más calmada. En vez de esforzarnos intentando encontrar los atributos que transformen el problema no lineal original en otro completamente lineal, dejemos que sea la propia máquina la que los encuentre. Ya vimos al final del movimiento anterior, que una neurona, es decir, un combinador lineal seguido de una función de activación era capaz de generar una función no lineal de las entradas.

Pues bien, aprovechemos y extendamos esa idea para la construcción automática de los atributos. Supongamos que queremos obtener el atributo x_1 como una función no lineal (arbitraria) de x, es decir, $x_1 = \varphi_1(x)$. Por ejemplo, el logaritmo de la temperatura del caso anterior. Para ello, en una primera capa, tal como se ilustra en la figura 16, generamos distintas versiones lineales $z_1, z_2, \ldots, z_n$ del atributo original x, mediante el uso de diferentes ponderaciones[23]. Así, la i-ésima combinación se obtiene mediante la expresión

$$z_i = w_{i,0}^{[1]} + w_{i,1}^{[1]} x \qquad (11)$$

23. Usamos la nomenclatura z_i y a_i para referirnos a la salida lineal y no lineal respectivamente de la i-ésima neurona de la primera capa. Usamos el término $w_{i,j}^{[l]}$ para el peso de la j-ésima entrada en la i-ésima neurona de la capa l. Por último, usamos el término $w_{i,0}^{[l]}$ para el valor del sesgo en la i-ésima neurona de la capa l.

A continuación, cada una de esas versiones z_i es transformada en una versión no lineal de la entrada, a_i, usando una función de activación predeterminada, ψ, de acuerdo con la expresión

$$a_i = \psi(z_i) = \psi\left(w_{i,0}^{[1]} + w_{i,1}^{[1]} x\right) \tag{12}$$

Por último, en una segunda capa de una única neurona, realizamos una combinación lineal de las distintas versiones no lineales de la capa anterior, de acuerdo con la expresión

$$x_1 = \varphi_1(x) = w_{1,0}^{[2]} + w_{1,1}^{[2]} a_1 + w_{1,2}^{[2]} a_2 + \cdots + w_{1,n}^{[2]} a_n \tag{13}$$

A una estructura como la que acabamos de describir se la suele denominar Perceptrón Multicapa (o MLP por sus siglas en inglés, *Multi-Layer Perceptron*). Pues bien, se puede demostrar que un MLP es capaz de aproximar cualquier función no lineal[24][27], [28]. Este resultado, denominado Teorema de Aproximación Universal, será la clave de bóveda

24. El Teorema de Aproximación Universal, demostrado originalmente por George Cybenko en 1989 para funciones de activación sigmoideas y generalizado por Kurt Hornik en 1991, establece que una red neuronal artificial de una sola capa oculta con una cantidad finita de neuronas y una función de activación no lineal puede aproximar cualquier función continua en un espacio compacto con el nivel de precisión que se desee.

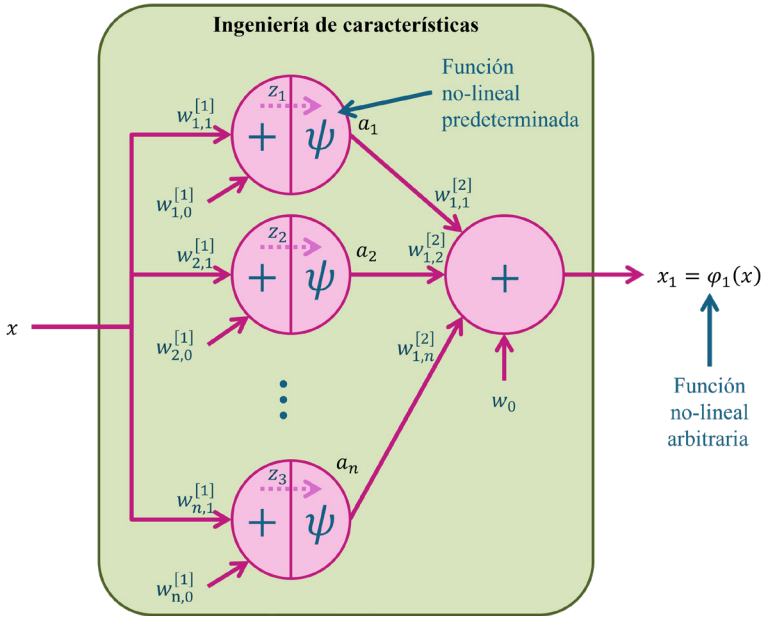

Figura 16. Generación de atributos mediante combinación de neuronas

que nos permitirá modelar mediante redes de neuronas cualquier tipo de problema, con independencia de su complejidad[25]. Y, además, sin necesidad de conocer *a priori* la forma que tenga la relación de no linealidad entre variables.

25. Aunque el Teorema de Aproximación Universal demuestra matemáticamente la capacidad de una red neuronal de aproximar cualquier función, eso no quiere decir que seamos capaces de entrenarla eficientemente con un algoritmo de gradiente descendente y un número necesariamente finito de datos.

Para que la arquitectura necesaria esté completa, nos falta por determinar qué función de activación predeterminada, ψ, debemos elegir. Y la respuesta a este interrogante es que casi cualquier función no lineal resultará válida[26][29]. Históricamente la primera propuesta (en 1943) fue usar una función escalón[30], idea que fue pronto abandonada por su dificultad para obtener los valores óptimos de los coeficientes[27]. Más éxito tuvieron la función sigmoide (en 1986)[31] y la tangente hiperbólica *tanh* (en 1991)[32] que sobrevivieron durante más de un cuarto de siglo, hasta que apareció la unidad de rectificación unitaria ReLU (en 2011)[33]. A pesar de que en las últimas tres décadas se han propuesto cientos de funciones de activación alternativas[34], la práctica industrial actual ha convergido de forma casi absoluta, estimándose en más de un 90 % el uso de arquitecturas basadas en ReLU[28]. Un resumen de las funciones de activación mencionadas puede verse en la figura 17.

---

26. Los únicos requisitos que debe cumplir la función de activación es que sea continua (o continua casi en todas partes) y, fundamentalmente, no polinomial, ya que las combinaciones lineales de polinomios de grado fijo no pueden aproximar funciones arbitrarias.

27. La derivada de la función escalón es cero en casi todas partes e infinita en el origen. Por tanto, es una función muy inadecuada para aplicarle el método del gradiente descendente que está basado en las derivadas.

28. Las modernas redes neuronales utilizan funciones de activación que evitan el cambio brusco de pendiente que presenta la función ReLU en el origen. Entre ellas destacan GELU y SiLU.

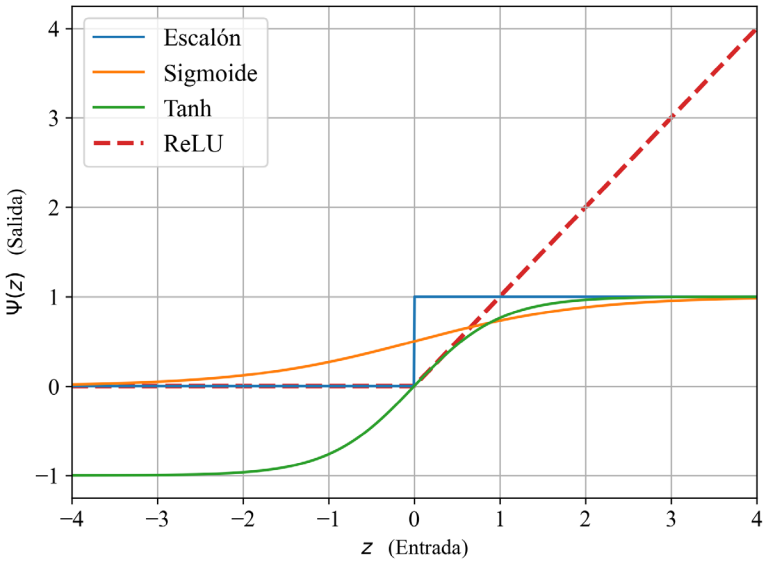

Figura 17. Funciones de activación

Ya tenemos completamente definida la arquitectura necesaria para la construcción automatizada de atributos no lineales. Lo que resta es obtener los valores de los distintos coeficientes $w_{i,j}^{[l]}$. Y, para ello, usamos el mismo procedimiento descrito en los movimientos anteriores: ir ajustando el valor de los parámetros mediante el gradiente descendente de la función de pérdida.

Apliquemos ahora estas ideas a la construcción automática de los atributos de la almazara que tanto esfuerzo le costó a nuestra ingeniera diseñar a mano, donde tuvo que transformar un logaritmo, una función inversa y una

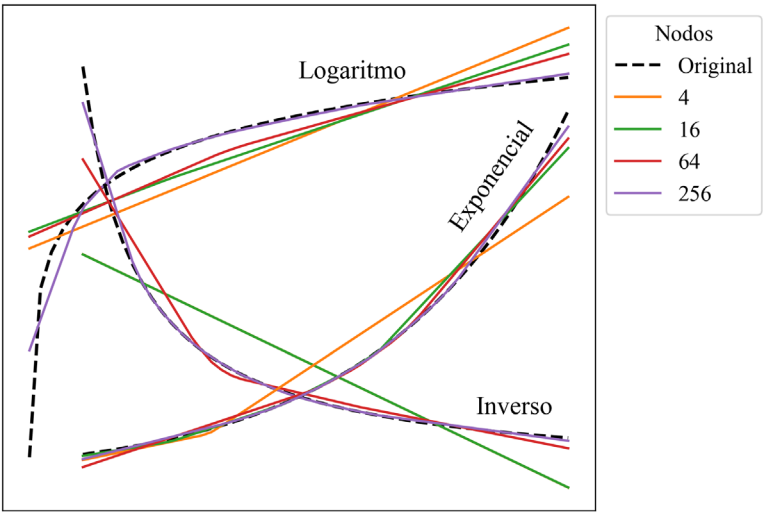

Figura 18. Aproximación de funciones no lineales mediante red neuronal de una capa

exponencial. En la figura 18 se refleja el resultado obtenido al generar estas tres funciones no lineales usando arquitecturas con un número creciente de neuronas.

El buen resultado obtenido en este caso nos invita al optimismo, pero es cierto que la construcción de características requeridas en la almazara es bastante sencilla: tal como mostramos en la ecuación (8), página 45, nos encontramos en presencia de funciones no lineales suaves[29], de variables

29. Funciones infinitamente diferenciables en todo su rango de aplicación.

separables. Adentrémonos en casos más complejos: situaciones donde la función que queremos construir presenta saturaciones[30] y múltiples variables acopladas (no separables). Para profundizar en ello, examinemos un ejemplo del tercero de nuestros grados: la Ingeniería Mecánica.

Nos liberamos pues ya del ritmo obsesivo del *scherzo* que empleamos en el frenético análisis manual de las relaciones entre variables. Contemplemos ahora, desde la serena perspectiva propia del adagio de este tercer movimiento, cómo la red neuronal es capaz de producir el autodescubrimiento de los atributos necesarios.

Asumamos el rol de un ingeniero en una planta de desarrollo de componentes de automoción, donde se diseñan sistemas de suspensión inteligente mediante amortiguadores de características variables. Durante la carrera de trabajo del pistón, las condiciones operativas influyen de forma compleja (no lineal) en fenómenos físicos y dinámicos como la fuerza de resistencia hidráulica, la viscosidad del fluido y los efectos de cavitación o saturación en las válvulas de alivio. Asimismo, la corriente eléctrica inyectada al actuador modifica instantáneamente las propiedades reológicas del fluido para endurecer la respuesta, pero un exceso de

30. La función contiene puntos angulosos en los que no es diferenciable.

velocidad del vástago provoca el truncado físico de la fuerza por límites mecánicos del sistema.

En este escenario necesitamos desarrollar un modelo que permita predecir la fuerza de resistencia del amortiguador en función de la velocidad relativa del pistón, la temperatura del aceite y la intensidad de la corriente en la bobina, con el objetivo de caracterizar con precisión la dinámica del componente para su posterior integración en el sistema de control del vehículo. En la figura 19 se presenta un esquema general del problema planteado.

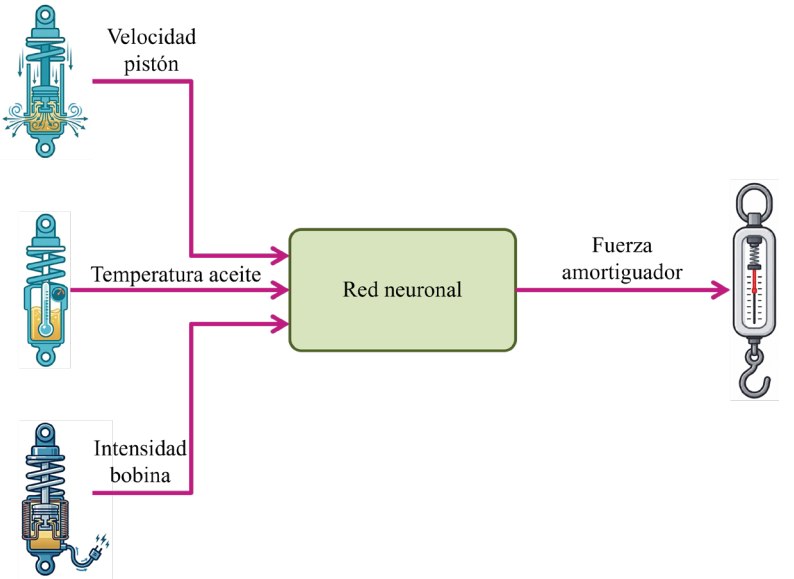

Figura 19. Esquema general del problema de estimación de la fuerza de resistencia de un amortiguador

Diferentes ensayos dinámicos realizados en un banco de pruebas controlado por operarios expertos nos proporcionan un conjunto de datos de 1000 muestras, de las cuales se reserva un subconjunto para la evaluación del modelo y el resto para el entrenamiento. La relación entre los atributos de entrada y la variable de salida resulta ser significativamente compleja. Un ejemplo parcial de dicha relación se muestra en la figura 20, en la que se representa la evolución de la fuerza del amortiguador F en función de la velocidad del pistón v, para una temperatura de =20 °C, y varios

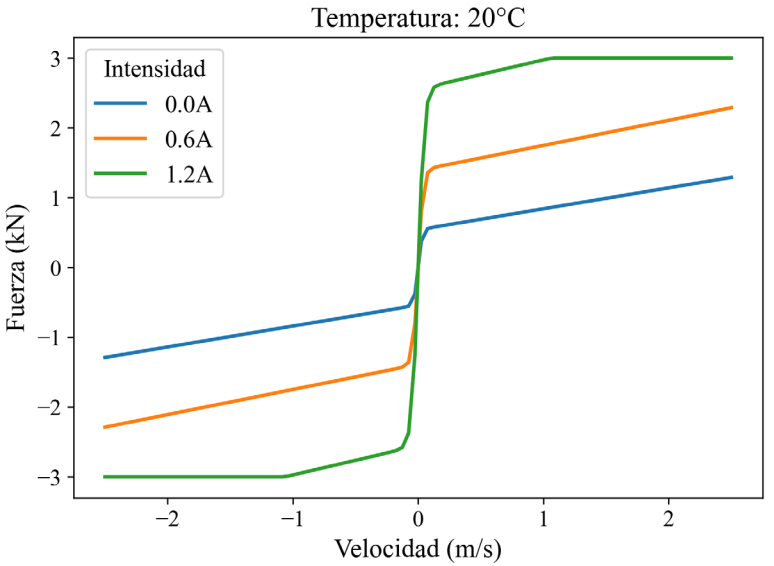

Figura 20. Relación entre la velocidad del pistón y la fuerza del amortiguador para distintas condiciones de operación

valores de la intensidad *I* en la bobina. Puede observarse claramente que el comportamiento es fuertemente no lineal, presentando, además, zonas de saturación de la fuerza.

En una primera aproximación, dejemos que una red neuronal como la de la figura 16, página 52, intente construir los atributos necesarios. Desgraciadamente, por mucho que aumentemos el número de neuronas, el resultado es decepcionante. Una red con una única capa, lo que se denomina una red neuronal superficial, tiene muchas dificultades cuando las funciones a descubrir son muy complejas. Para solventar el problema, en vez de tratar de construir los atributos en un único paso, es decir, usando una única capa, permitamos a la red que los construya en varios pasos: que genere unos atributos provisionales en una primera capa, otros más depurados en la siguiente, y así sucesivamente hasta conseguir alcanzar el resultado deseado. La figura 21 nos muestra un ejemplo de una arquitectura de este tipo con tres variables de entrada y una de salida. La solución se articula con dos capas ocultas, cada una de las cuales va construyendo atributos intermedios cada vez más adecuados. Este tipo de redes, al contar con más capas, hace que se denominen redes neuronales profundas.

Pues bien, aplicando esta idea al problema propuesto, encontramos que con solo dos capas la red es capaz de reconstruir muy aproximadamente la función no lineal de entrada (ver figura 22).

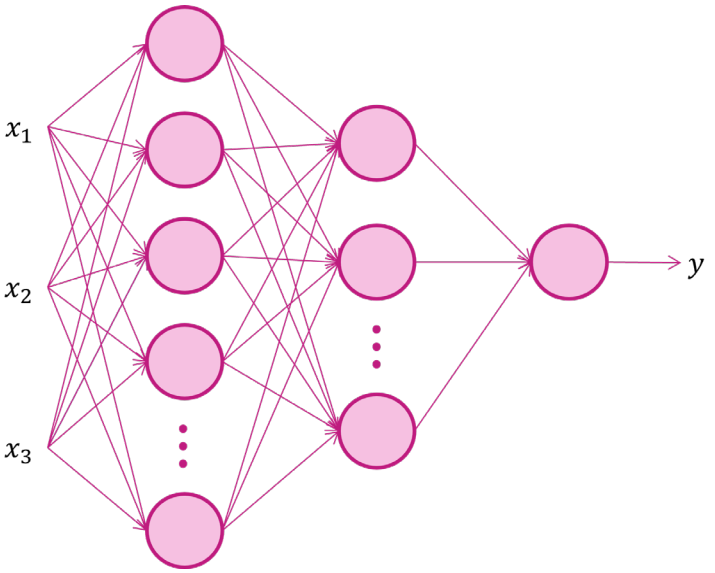

Figura 21. Ejemplo de red neuronal profunda

Tomemos como ejemplo una arquitectura con dos capas ocultas constituidas por 256 y 64 nodos respectivamente. Si miramos con atención en el interior de la red vemos que los atributos construidos en la primera capa tienen una forma similar a la de la función ReLU, pero con distintas pendientes y posiciones del punto de quiebro (figura 23). Se intuye la dificultad de reconstruir con esos elementos tan simples una función tan compleja como la del amortiguador.

La primera capa de la red actúa como un banco de filtros geométricos. El optimizador ha distribuido estratégicamente los pesos y sesgos para especializar a los nodos en familias de

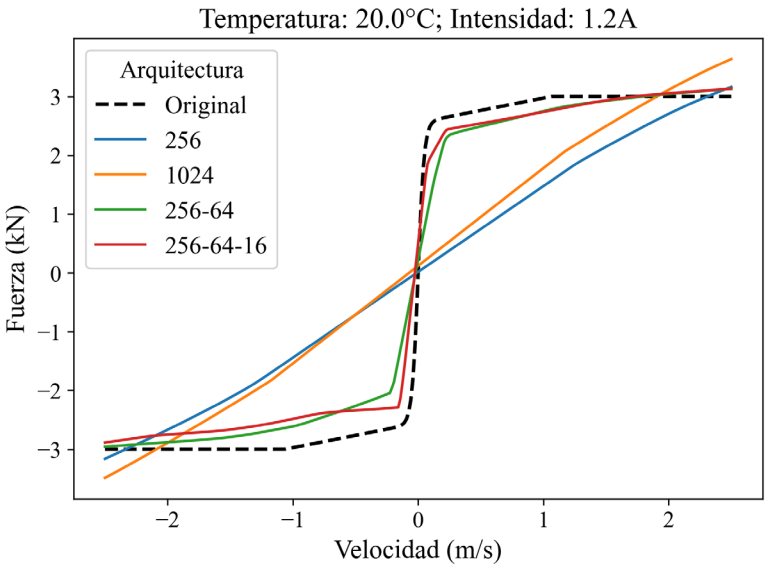

Figura 22. Reconstrucción de la relación entre la velocidad del pistón y la fuerza del amortiguador usando redes neuronales con diferentes arquitecturas

trabajo: un bloque monitoriza exclusivamente la severidad de la extensión (velocidades negativas), otro la de la compresión (velocidades positivas), mientras que un tercer grupo minoritario establece la línea de base asimétrica del componente físico.

Al combinarse linealmente en la segunda capa, estas pendientes individuales y cortes angulares se sumarán y restarán como un puzle, permitiendo suavizar las esquinas y reproducir fielmente las transiciones no lineales y los fenómenos de saturación hidráulica del amortiguador.

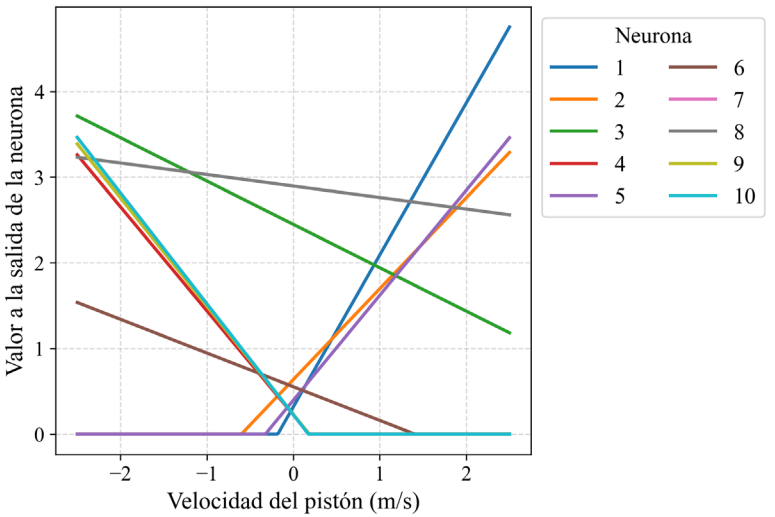

Figura 23. Atributos construidos por las 10 primeras neuronas de la primera capa

Pero si nos fijamos ahora en los atributos de salida de la segunda capa (figura 24) observamos que exhiben una profunda reestructuración y síntesis de la información. El optimizador ha convergido de manera unánime en dos únicos macroscomportamientos simétricos y altamente especializados: un conjunto de nodos dedicado exclusivamente a la fase de extensión (velocidades negativas) y otro a la fase de compresión (velocidades positivas).

Visualmente resalta la aparición de un 'codo' dinámico en la zona de bajas velocidades (±0.2 m/s), donde el modelo genera de forma autónoma una aproximación bilineal por

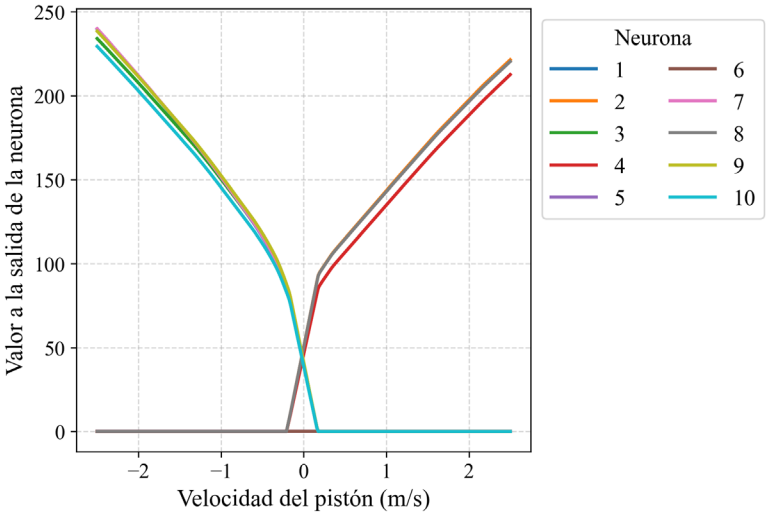

Figura 24. Atributos construidos por las 10 primeras neuronas de la segunda capa

tramos que emula fielmente la transición de rigidez hidráulica del fluido del amortiguador.

Todos estos atributos de la segunda capa se combinarán en la neurona de la capa de salida para obtener los resultados que ya mostramos en la figura 22 (línea verde, correspondiente a una arquitectura de 2 capas).

Los resultados anteriores han sido obtenidos para una arquitectura en dos capas tomada de ejemplo. Pero, en nuestro caso, ¿cuál es la mejor combinación posible de número de capas ocultas y cantidad de neuronas en cada una de ellas? Los modelos basados en redes neuronales plantean pues el

desafío de elegir la mejor arquitectura posible. Simplificando mucho se puede decir que esta arquitectura se elige sencillamente probando: se toman diversas combinaciones posibles de número de capas y número de nodos en cada capa, seleccionando la combinación que obtenga la mejor evaluación.

Este proceso se denomina optimización o búsqueda de hiperparámetros. Aunque en esencia consiste en evaluar diferentes configuraciones, no se realiza al azar; se emplean estrategias sistemáticas como la búsqueda en rejilla, la búsqueda aleatoria, o la optimización bayesiana, la cual aprende de los intentos previos para sugerir la siguiente combinación. Asimismo, la selección final se rige por el principio de parsimonia (Navaja de Ockham[31]): a igualdad o similitud de rendimiento, siempre se prefiere la arquitectura más simple (menos capas y neuronas), ya que reduce el coste computacional y mejora la capacidad de generalización del modelo.

Es crucial que, para evitar el sobreajuste (el fenómeno por el cual el modelo memoriza los datos de entrenamiento perdiendo capacidad de generalizar ante datos nuevos), esta evaluación se realice sobre un conjunto de datos independiente. Por ello, como vimos, el conjunto de datos original se subdivide en uno de desarrollo y otro de evaluación (usado para la calificación final del modelo). Ahora, a su vez, el de

31. *Entia non sunt multiplicanda praeter necessitatem*: No deben multiplicarse [aumentarse] las entidades más de lo necesario.

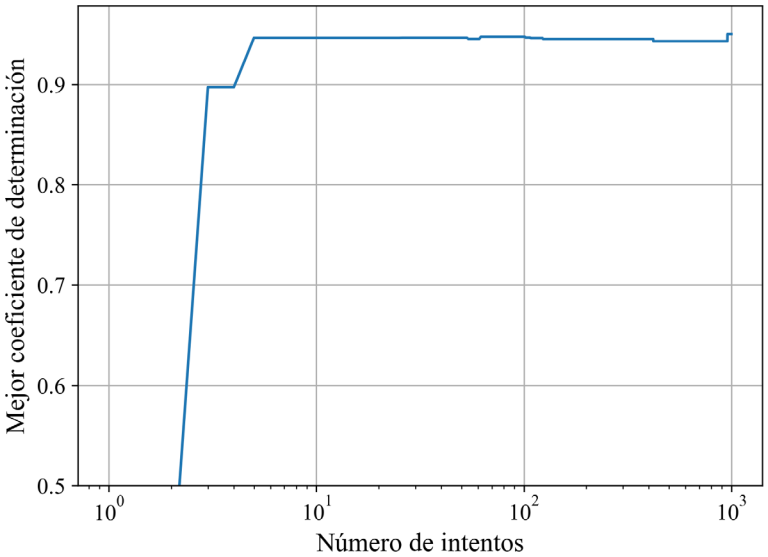

Figura 25. Evolución de las prestaciones de la red neuronal al probar distintas arquitecturas

desarrollo se vuelve dividir en dos subconjuntos: el de entrenamiento (para optimización de los parámetros) y el de validación (para selección de hiperparámetros).

En la figura 25 se muestra el progreso de los resultados obtenidos por la red neuronal durante el proceso de búsqueda bayesiana de la mejor arquitectura posible[32]. Esta

32. En esta búsqueda se incluyen también otros hiperparámetros como la tasa de aprendizaje usado en el algoritmo de gradiente descendente o el número de pasadas (épocas) que realiza dicho algoritmo.

ha resultado ser una red de 3 capas ocultas con 753, 21 y 553 neuronas respectivamente[33]. La evaluación final obtenida por dicho modelo es de un 9.49 sobre 10.

Un aspecto relevante que considerar en el uso de estas redes neuronales es el número de muestras necesarias para entrenarlas. Y la conclusión es que, cuanto más simple sea el problema, menos datos se requieren para el entrenamiento. Aplicando la misma red neuronal que acabamos de obtener a los tres problemas analizados hasta ahora, podemos trazar una curva que mida las prestaciones obtenidas en función del número de muestras necesarias. El resultado se muestra en la figura 26 junto con la indicación del total de muestras necesarias[34].

Se ha incluido también una cierta penalización (hasta de un 10 %) por complejidad del modelo (medida en número de parámetros de la red).

33. Este resultado se ha obtenido con un algoritmo de gradiente descendente entrenado durante 86 épocas con una tasa de aprendizaje de 0.01. En este contexto, una época equivale a un ciclo completo en el que el modelo procesa y analiza todos los datos del conjunto de entrenamiento.

34. Definimos el número de muestras necesarias como aquellas que permiten alcanzar o superar por primera vez una evaluación del 99 % sobre el valor máximo. Se fija este umbral por el mismo principio de parsimonia que enunciamos anteriormente: evitar un exceso de datos de entrenamiento que solo obtienen mejoras marginales en el rendimiento.

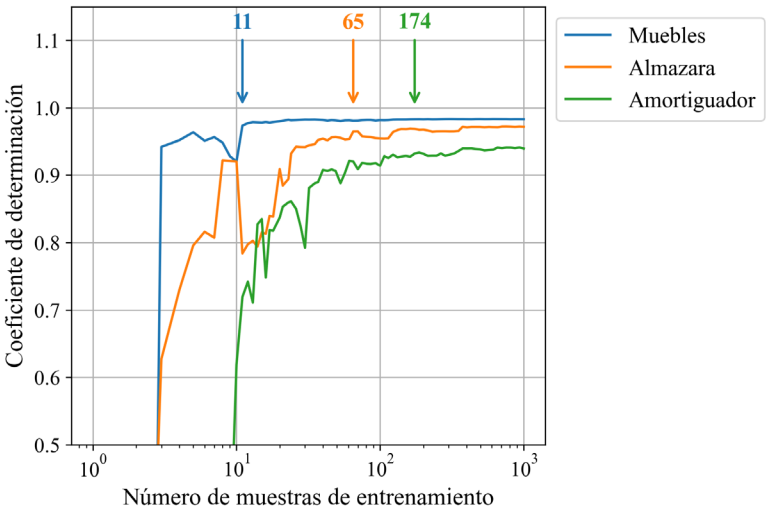

Figura 26. Evolución de las prestaciones en función del número de muestras usadas en el entrenamiento

Podemos comprobar que el más sencillo de los problemas, el diseño de muebles, requiere tan solo 11 muestras para ser convenientemente modelado, mientras que la generación de las no linealidades presentes en el problema de la almazara (ecuación de Arrhenius) necesita 65 muestras. Por último, son necesarias 174 muestras para que la red neuronal autodescubra las funciones subyacentes en el caso del amortiguador. Este fenómeno se puede generalizar: a mayor complejidad del problema, mayor tamaño (y/o complejidad) de la red neuronal que pueda modelarlo, y mayor número de datos necesario para entrenarla.

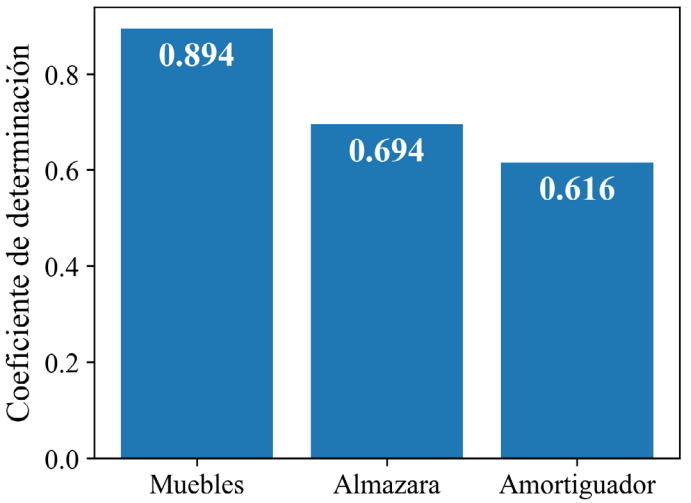

Figura 27. Prestaciones obtenidas por un modelo lineal en los distintos problemas planteados

A la vista de las evaluaciones obtenidas por nuestros modelos en los tres problemas de ingeniería planteados, todas ellas en torno al sobresaliente, cabría pensar que predecir datos es una asignatura bien sencilla. La realidad es bien distinta: el éxito obtenido hasta ahora se ha debido a que, a medida que el problema se hacía más difícil (más variables, mayor interdependencia entre ellas y, sobre todo, mayor no linealidad) hemos ido buscando modelos cada vez más «sofisticados». Esta afirmación es fácil de comprobar: si tratamos de resolver el problema de la almazara o el del amortiguador usando una simple regresión lineal, como

hicimos en el caso del diseño de muebles, observamos que las evaluaciones decaen sensiblemente, tal como se muestra en la figura 27.

Para buscar una melodía a la que asociar el problema del amortiguador tendríamos que recurrir a obras musicales de mayor envergadura. No en vano, nos enfrentamos a un problema con tres variables de entrada, acopladas de forma fuertemente no lineal, para cuyo modelado hemos tenido que emplear una red neuronal profunda (tres capas ocultas) con más de mil neuronas. Les propongo asimilar este caso a un conocido vals: *El Danubio azul* de Johann Strauss hijo, una obra para orquesta que se despliega en unas 10.000 notas y que presenta una complejidad y riqueza musical significativamente mayores que las dos piezas anteriores.

Una última pausa antes de proseguir, para que se acomoden el coro y los solistas.

Cuarto movimiento

Finale Presto. La llegada (*ad astra*)

El final rompe todo lo anterior: Primero rechaza musicalmente los temas previos (como diciendo: «eso no era suficiente»). Luego aparece el tema de la Oda a la Alegría de Friedrich Schiller. Entra el coro: algo revolucionario en una sinfonía. Aquí sí: hemos llegado «a las estrellas». Mensaje de unidad, fraternidad y alegría universal. La lucha previa cobra sentido. Lo individual se convierte en algo colectivo y trascendente.

Recorramos este cuarto movimiento, al igual que hace la Novena, en dos partes: una solo instrumental, y otra vocal en la que aparecerá finalmente el texto, la palabra. Y para la parte instrumental nos basaremos en un ejemplo tomado del cuarto de nuestros grados: la ingeniería eléctrica.

Pongámonos en la piel de una ingeniera en una subestación eléctrica, concretamente la de El Alamillo, encargada de suministrar energía eléctrica a la nueva sede de nuestra Escuela

Politécnica Superior y a la parte norte del Parque Tecnológico de Cartuja. Durante la operación diaria de la red, en un entorno crítico de alta variabilidad, la demanda de potencia activa y reactiva influye de forma compleja (no lineal) en fenómenos físicos y de gestión como la saturación de los transformadores, las caídas de tensión en las líneas de distribución y la estabilidad del factor de potencia. Así mismo, la inercia térmica de los edificios y los hábitos de consumo de la comunidad universitaria, investigadora y empresarial del Parque marcan una fuerte dependencia temporal y cíclica (secuencial), pero factores exógenos repentinos o rupturas estructurales en el calendario pueden desestabilizar la predicción.

En este escenario necesitamos desarrollar un modelo predictivo secuencial que permita anticipar el consumo de potencia activa y reactiva en las próximas cuatro horas en función del consumo del día anterior ($24 \times 4 = 96$ valores), con el objetivo de optimizar el control predictivo de las baterías de condensadores, anticipar el estrés térmico de las máquinas y mejorar la gestión del despacho de carga sin comprometer la seguridad del suministro eléctrico[35]. Y para

35. El ejemplo de este cuarto movimiento es, sin duda, el más cercano a las líneas de trabajo del autor que, en los últimos años, han estado centradas principalmente en la gestión de la demanda de electricidad. Esto incluye, entre otras actividades, el manejo e integración de las distintas fuentes de datos[53], la predicción de la demanda a largo plazo[54], la caracterización espectral del consumo[55], la vinculación

obtener este modelo vamos a recuperar la metodología en seis etapas que describimos en el primer movimiento[36].

1. Muestreo

El registro cuartohorario[37] del contador de facturación de la subestación, monitorizado en condiciones reales de funcionamiento, nos proporciona un conjunto de datos estructurado en series temporales. En este *dataset*, cada muestra queda descrita por un vector de atributos de entrada a lo largo de la ventana de contexto (correspondiente a $24 \times 4 = 96$ muestras), compuesto por la potencia activa media, que denotaremos como P, y la potencia reactiva media, denotada como Q. Disponemos de los datos de consumo de los dos últimos meses. En la figura 28 se muestra un ejemplo de la evolución de ambas variables a lo largo de un periodo de tres días laborables.

---

de los perfiles de demanda con la actividad económica y la ubicación [56], [57], [58], la predicción de la vida útil de baterías eléctricas[59], o la identificación de cargas eléctricas[60].

36. Esta metodología se utilizó también en el segundo y tercer movimiento, aunque no se declaró de manera explícita por mejorar la fluidez del discurso.

37. Los contadores de facturación actuales suelen registrar los consumos en periodos de quince minutos por lo que se denominan quinceminutales o cuartohorarios.

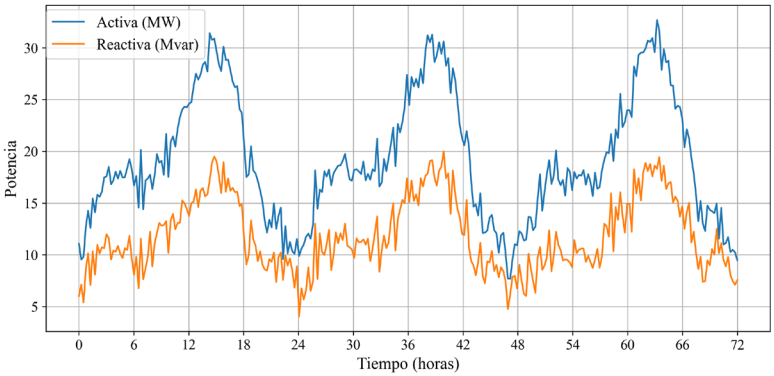

Figura 28. Ejemplo de la evolución de las potencias activa y reactiva durante un periodo de tres días laborables

En total se dispone de 5856 muestras correspondientes a un total de 61 días, a razón de 24 horas cada día, y 4 medidas por hora. Este conjunto de datos se divide en dos secciones: una para entrenamiento (80 %) y otra para evaluación (20 %). Como en el caso de las series temporales el orden de los datos sí es relevante, esta división se hace respetando el orden cronológico.

2. Caracterización de las muestras

Abordar el problema de predicción de los siguientes valores de una serie temporal, como la del ejemplo, mediante el uso de herramientas como las que hemos venido describiendo hasta ahora, requiere un paso previo: la adecuada

identificación y caracterización de las muestras. Y por simplicidad, comenzaremos explicando cómo se hace tomando como ejemplo la secuencia de datos correspondiente a una única magnitud: la potencia activa.

Comenzamos utilizando una ventana que se va deslizando sobre la serie original[38] y que, en cada desplazamiento, nos ofrece un dato de muestra y un valor objetivo, tal como se muestra en la figura 29. Como el problema es la predicción de la demanda en base a la del día anterior, la ventana contendrá 96 elementos: los 96 valores anteriores a los que tenemos que predecir. Al deslizar la ventana sobre la serie temporal, en cada instante[39] tenemos una muestra de los 96 últimos valores de la potencia activa y también sabemos cuál será el valor de potencia activa en el instante siguiente. Así, la primera muestra contiene los valores de las potencias en los instantes del 1 (P_1) al 96 (P_{96}) y el valor que tiene aprender a predecir es el correspondiente al instante 97 (P_{97})[40].

---

38. Esta técnica se denomina de ventana deslizante y es la más extendida para caracterizar series temporales.

39. Los instantes de muestreo en nuestro ejemplo ocurren cada 15 minutos, como es habitual en los modernos contadores de medidas eléctricas. Cada uno de ellos se corresponde con una muestra.

40. En el manejo de series temporales mediante ventana deslizante se manejan hasta 4 índices o variables temporales. Antes de aplicar la ventana deslizante manejamos dos: a) el tiempo absoluto R que indica la fecha y la hora; y b) el índice cronológico k que va numerando cada instante de muestreo, y que va entre 1 y n. Después de aplicar la

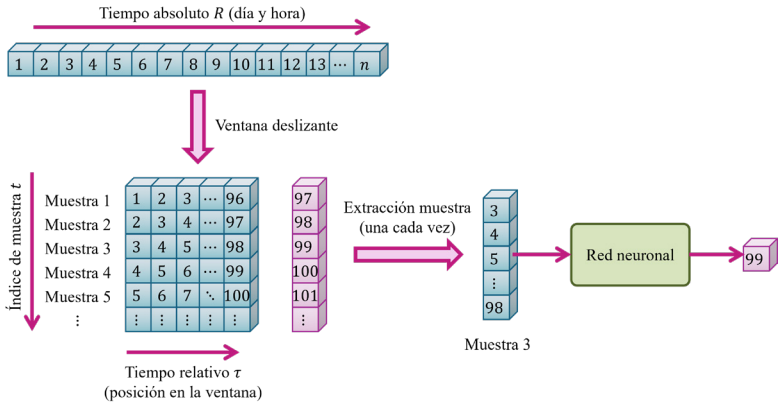

Figura 29. Creación del conjunto de datos a partir de la serie temporal mediante ventana deslizante

Con este procedimiento hemos transformado la naturaleza del problema[41]: ya no nos enfrentamos a una secuencia de valores (una serie temporal), sino que ahora se trata de predecir la variable «demanda de potencia» P_{t+1} tomando como datos 96 variables: los valores de potencia en los 96 instantes anteriores, de P_t a P_{t-95}. Y formalmente

---

ventana deslizante manejamos otros dos: c) el índice de la muestra t, que va entre 1 y $n - 96 - 1$ (para ventana de tamaño 96 y horizonte de predicción $h = 1$); y d) el tiempo relativo dentro de la muestra τ, que va entre 1 y 96.

41. La validez de este procedimiento de ventana deslizante viene sustentada por el teorema de Takens que afirma que, si se cumplen ciertas condiciones, es posible reconstruir una serie a partir de un conjunto de valores anteriores de la misma[61].

estas variables juegan el mismo papel que la temperatura y la velocidad en el problema de la almazara o que la temperatura, velocidad e intensidad eléctrica en el problema del amortiguador.

3. Formulación de hipótesis

La ingeniera en la subestación asume que existe una relación no lineal que liga los 96 valores anteriores con la demanda actual. Pero esta relación intuye que debe ser compleja y, en vez de esforzarse tratando de descubrirla, decide usar una red neuronal que autodescubra estas funciones. Su arquitectura será pues similar a la de los ejercicios anteriores, tal como se muestra en la figura 30. En ella aparecen varias capas que construyen atributos, es decir, relaciones no lineales entre los valores anteriores de potencia, que nos permitirán predecir el próximo valor. La arquitectura específica de la red se elige inicialmente de forma arbitraria[42].

42. Una experiencia previa en el uso de estas técnicas hace desarrollar a la ingeniera una cierta intuición que le facilita elegir una arquitectura inicial «razonable». En cualquier caso, el método descrito no requiere de esa intuición y una arquitectura elegida completamente al azar es también válida en este momento inicial.

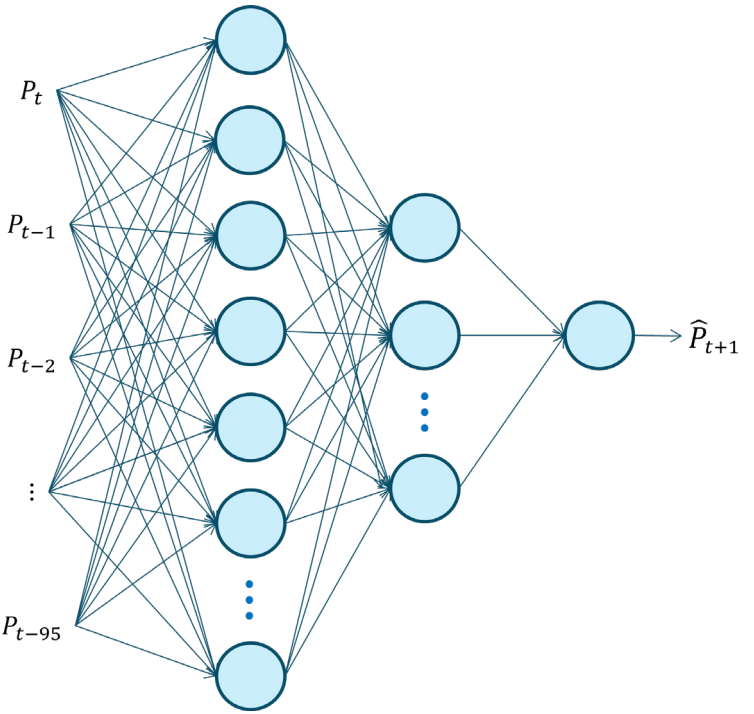

Figura 30. Ejemplo de red neuronal aplicada a la predicción del siguiente valor de potencia activa

Al finalizar la etapa de evaluación volveremos iterativamente a este paso 3 eligiendo nuevas arquitecturas que mejoren la predicción[43].

43. Como ya comentamos, estas nuevas arquitecturas pueden tomarse mediante una búsqueda sistemática en rejilla, una búsqueda aleatoria o una búsqueda bayesiana basada en lo aprendido sobre la evaluación de arquitecturas previamente ensayadas.

4. Selección de la función de pérdida

Al tratarse de un problema de regresión[44], es decir, la predicción de un valor numérico continuo, usaremos como función de pérdida el error cuadrático medio que ya definimos en la ecuación (2), página 27.

5. Optimización de la función de pérdida

Como podemos constatar, el valor de la función de pérdida depende de los datos de entrenamiento (x_i, y_i), que son fijos, y de los parámetros w de la hipótesis (h_w) que sí podemos variar. Por tanto, usando el algoritmo de gradiente descendente o alguna de sus variantes, ajustamos el valor de los parámetros w que minimicen la función de pérdida.

6. Evaluación del modelo

La última etapa consiste en evaluar las prestaciones del modelo usando el 20 % de la serie que habíamos reservado para este menester. Y en la primera iteración (con una

44. La otra gran familia de problemas predictivos en inteligencia artificial la constituye la clasificación, cuyo objetivo es predecir una categoría. En estos escenarios se suelen adoptar funciones de pérdida específicas, como la logística, la bisagra (*hinge loss*), la entropía cruzada categórica, u otras.

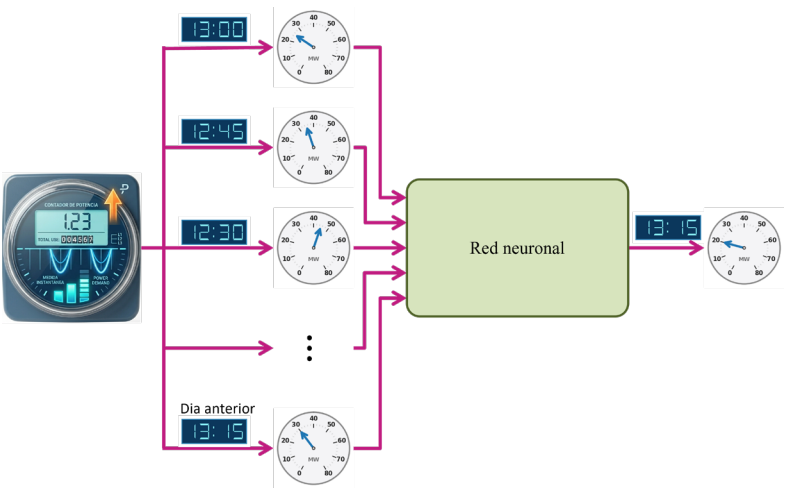

Figura 31. Esquema general del problema de predicción de la demanda de potencia activa. En cada lectura se indica la hora a la que fue tomada (por ejemplo 13:00) y el valor de la potencia en ese instante (la flecha azul dentro del dial)

arquitectura de la red elegida de forma arbitraria) obtenemos un suspenso: un 4.03 sobre 10. Volvemos pues a la etapa 3 de selección de la arquitectura y vamos probando iterativamente nuevas configuraciones de la red[45]. Después de menos de 100 iteraciones obtenemos una evaluación final de 9.27 sobre 10.

45. Esta búsqueda iterativa se hace utilizando una búsqueda bayesiana que no es puramente aleatoria, sino que tiene en cuenta los resultados obtenidos en intentos anteriores.

Una vez cubiertas las seis etapas podemos trazar el esquema conceptual de la predicción del siguiente valor de demanda de potencia, según se detalla en la figura 31. En el ejemplo mostrado, se toman las medidas de potencia activa desde las 13:15 del día anterior hasta las 13:00 del día actual, con periodos de 15 minutos. Estos 96 valores se inyectan en una red neuronal que, previamente entrenada, es capaz de predecir la demanda de potencia a las 13:15 de hoy.

La estructura propuesta es capaz de estimar con un notable acierto la demanda del instante de muestreo siguiente. Pero en muchos casos, eso no es suficiente: la ingeniera de nuestra subestación necesita conocer la estimación de demanda, no de los siguientes 15 minutos, sino de las próximas 4 horas. Es decir, el modelo no debe predecir un único valor sino 16, cuatro por cada hora.

Este problema de predicción con horizontes temporales superiores a un instante puede (y suele) resolverse mediante la aplicación de una estrategia recursiva, donde las predicciones de horizontes anteriores se reincorporan sucesivamente como variables de entrada para los horizontes subsiguientes. En la figura 32 se ilustra este proceso recursivo.

En un primer momento, llamémosle instante o medida t, estimamos el siguiente valor de potencia $\hat{P}_{t+1}$ (la demanda dentro de 15 minutos) en base a las 96 medidas anteriores disponibles, es decir, usando los valores P_t a P_{t-95}.

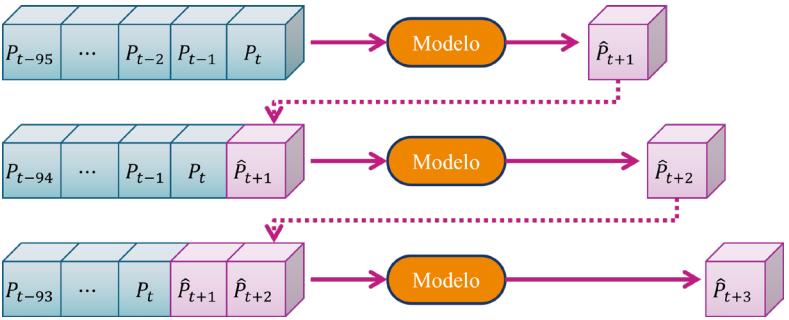

Figura 32. Predicción multihorizonte mediante la aplicación recursiva de la red neuronal

En un segundo paso, siempre en el instante t, intentamos predecir cuál será la potencia dentro de dos instantes de medida, $\hat{P}_{t+2}$, es decir, dentro de 30 minutos. Para ello, necesitaríamos las demandas de los 96 instantes anteriores, es decir, de P_{t+1} a P_{t-94}. En el instante t, disponemos de las medidas P_t a P_{t-94}, pero todavía no hemos medido P_{t+1}. Por ello, en su lugar, usamos la estimación obtenida en el paso anterior, es decir, $\hat{P}_{t+1}$. Este proceso se repite tantas veces como sea necesario hasta obtener todas las estimaciones dentro del horizonte de predicción.

Pero la ingeniera en la subestación no necesita solo predecir el valor de la potencia activa, sino también de la reactiva. Para quienes no estén muy familiarizados con el término, la potencia reactiva no produce ningún trabajo efectivo (como movimiento o calor), pero su presencia es imprescindible para el funcionamiento de numerosos dispositivos eléctricos. No

obstante, es crucial predecir su valor porque, simplificando, su transporte «ocupa sitio» en los cables.

Si nos fijamos en la serie temporal cuyos valores queremos predecir, nos encontramos que ahora es una serie multivariable, es decir, en cada instante de tiempo tenemos dos valores: las potencias activa y reactiva. Si le aplicamos la técnica de la ventana deslizante que ya comentamos anteriormente, el resultado es un bloque de datos de 3 dimensiones. Ahora no sólo tenemos ancho (tamaño de la ventana) y alto (número de muestras), sino que también tenemos profundidad (número de variables: activa y reactiva). Una estructura de este tipo se denomina un tensor. La figura 33 ilustra el concepto de serie multivariable y el tensor construido a partir de ella.

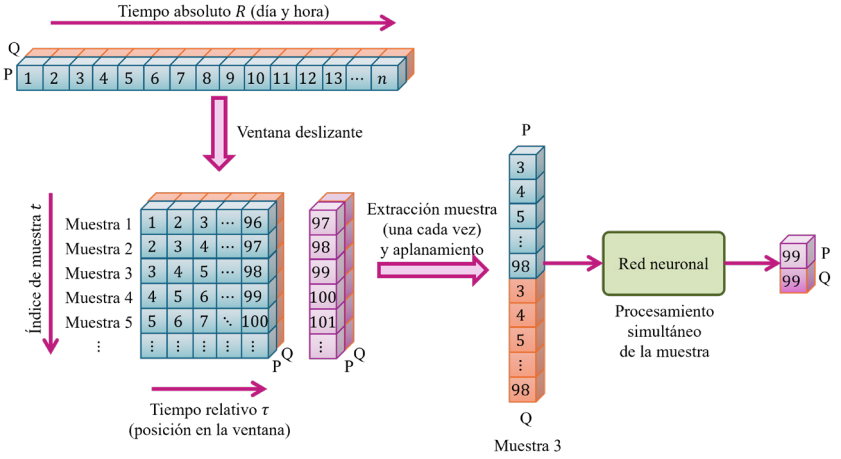

Figura 33. Serie temporal multivariable (activa, reactiva) y su conversión en un tensor de datos mediante ventana deslizante

Desgraciadamente, las redes neuronales que estamos manejando hasta ahora, conocidas como Perceptrón Multicapa (MLP), no son capaces de manejar estructuras tridimensionales (tensores). Por ello, en nuestro caso, tenemos que «aplanar» el tensor tal como aparece en la figura 33. Se trata de la yuxtaposición de los planos correspondientes a las potencias activa y reactiva, cada uno de los cuales constituye una matriz (una tabla de valores).

Para abordar la presencia de dos variables, se plantea, como primera opción, duplicar el esquema de la potencia activa. Es decir, habría dos redes neuronales independientes trabajando en paralelo: una para predecir la potencia activa y otra para la reactiva, tal como se ilustra en la figura 34.

Eligiendo las arquitecturas óptimas de cada red[46] y entrenándolas adecuadamente, podemos predecir de manera recursiva la demanda de ambas potencias para las próximas cuatro horas. En general, será más fácil predecir los valores más cercanos en el tiempo (el próximo cuarto de hora) y más difícil los más alejados (la demanda dentro de cuatro horas). Por ello, la evaluación de la predicción variará con el horizonte de predicción. En nuestro caso, el resultado recibe una evaluación de 8.94 sobre 10 para la predicción de la potencia activa del próximo cuarto de hora, bajando hasta el 8.06 para

46. Al ser redes independientes, sus arquitecturas pueden ser diferentes.

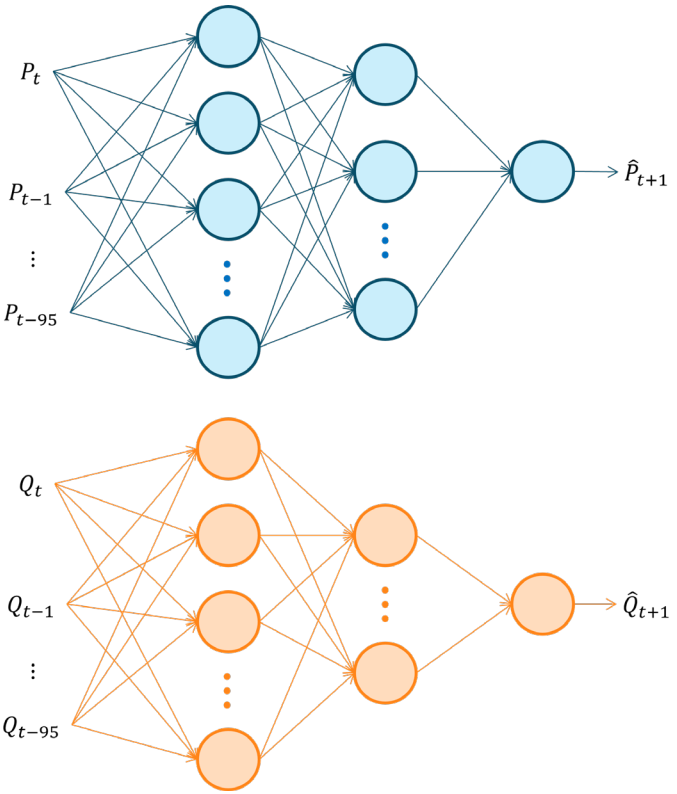

Figura 34. Predicción de las potencias activa y reactiva mediante redes neuronales independientes

la predicción dentro de 4 horas. En el caso de la potencia reactiva los resultados son similares, aunque algo más bajos, oscilando entre un 8.12 y un 7.35, respectivamente[47].

47. El menor rendimiento en la predicción de la potencia reactiva respecto a la activa es un fenómeno habitual. Responde a la menor

Pero abordar la predicción de las dos potencias mediante redes separadas, como acabamos de hacer, resta capacidades a la predicción. En efecto, cada una de las redes, por ejemplo, la de la potencia activa, es capaz de descubrir las relaciones que ligan el valor siguiente con sus valores anteriores, pero no contempla posibles influencias cruzadas entre ambas potencias. Para tratar de descubrir posibles interrelaciones es mejor fusionar las dos redes de forma que los nodos intermedios puedan ahora crear funciones no lineales de ambas potencias. Un ejemplo de la red resultante se muestra en la figura 35.

Un ejemplo del resultado de la predicción de la potencia activa a cuatro horas vista usando tanto las dos redes simples por separado, como una única red conjunta puede verse en la figura 36.

Finalmente, podemos resumir la evaluación de la predicción multihorizonte para las dos estrategias (redes independientes o fusionadas) tal como se recoge en la figura 37. Puede comprobarse cómo, efectivamente, la consideración conjunta de ambas potencias en la misma red ha permitido descubrir relaciones cruzadas que han mejorado la evaluación de la predicción.

---

relación señal-ruido de esta variable y a las discontinuidades provocadas por las actuaciones de control local en la red (como la compensación del factor de potencia), lo que eleva la volatilidad de la serie y penaliza la métrica de evaluación frente al comportamiento más rutinario y estacional de la potencia activa.

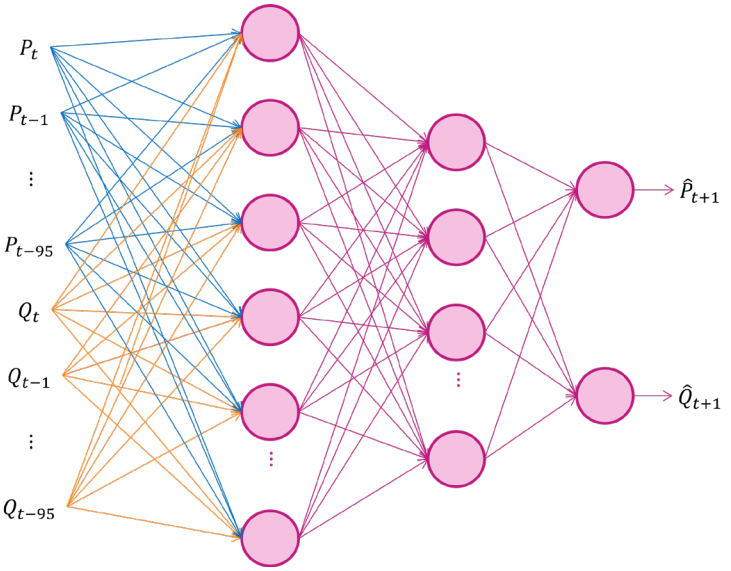

Figura 35. Predicción de las potencias activa y reactiva mediante una única red neuronal

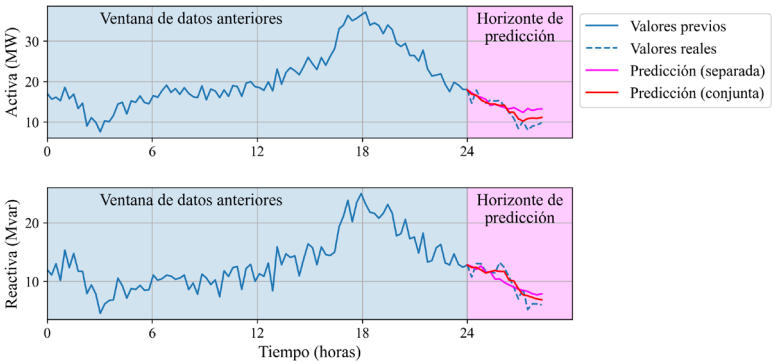

Figura 36. Predicción de las potencias activas y reactiva usando redes neuronales independientes (línea morada) o una única red neuronal conjunta (línea roja)

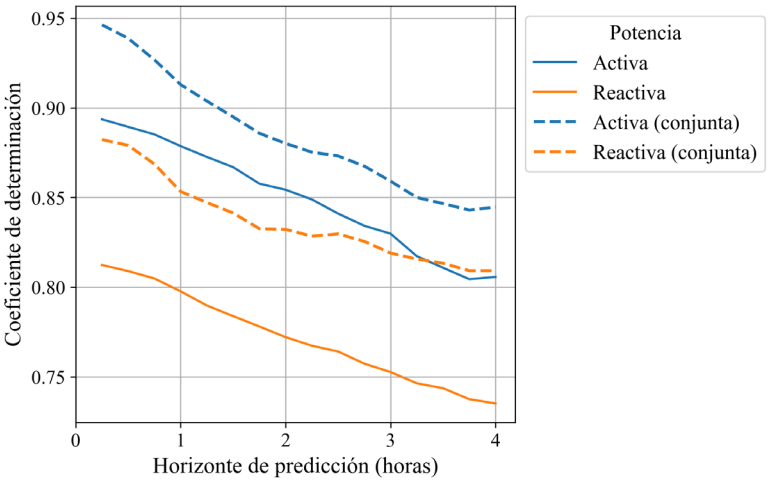

Figura 37. Evaluación de la predicción multihorizonte para las dos potencias (activa y reactiva) usando redes independientes o una única red conjunta

A pesar de su buen rendimiento, este modelo de red neuronal (el Perceptrón Multicapa) sufre de dos «puntos ciegos» importantes cuando trabaja con datos que evolucionan en el tiempo. El primero es que pierde el sentido del orden cronológico. Para la red, la ventana de datos del pasado es como una foto fija o una lista de números en una tabla; no entiende de forma natural qué pasó en el instante 1 y qué pasó en el instante 96, ni que entre ambos existe una secuencia. El segundo inconveniente es que desmonta la identidad del par (P, Q). La potencia activa y la reactiva son las dos caras de una misma moneda en cada instante. Sin embargo,

para poder procesar la información, la red se ve obligada a «triturar» y estirar los datos en una fila única de números independientes. Al hacerlo, disocia ambas potencias, olvidando que nacieron juntas en el mismo momento y en el mismo lugar de la red eléctrica.

Para resolver ambos problemas han surgido dos estrategias: las redes recurrentes y los *Transformers*. Analicemos el enfoque de cada una de ellas. En las redes recurrentes, también usamos la ventana deslizante para construir muestras de, por ejemplo, 96 instantes de tiempo. La diferencia con el enfoque anterior es que cada muestra no se aplana y se procesan todos sus valores simultáneamente, sino que se mantiene su estructura secuencial. La figura 38 refleja los 96 valores de la muestra correspondiente al instante $t = 98$. El tiempo relativo a esa muestra se denota como τ y, en nuestro caso, está en el rango entre 1 y 96.

En un instante de tiempo relativo τ entra a la red neuronal un solo vector (P_τ, Q_τ), no toda la muestra. Ese vector es tratado en una red neuronal generando unos

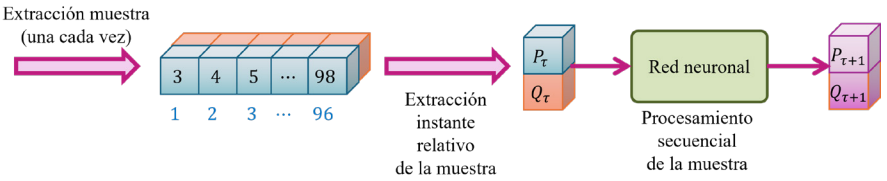

Figura 38. Muestra en un instante genérico preparada para su procesamiento secuencial en una red recurrente

atributos intermedios, digamos por simplicidad que son tres: $H_\tau = (H_{1,\tau}, H_{2,\tau}, H_{3,\tau})$. A continuación, una segunda red toma H_τ y los convierte en las predicciones $(\hat{P}_{\tau+1}, \hat{Q}_{\tau+1})$. El elemento fundamental de las redes recurrentes estriba en que, en el instante siguiente, a las entradas $(P_{\tau+1}, Q_{\tau+1})$ se le añaden como entradas los valores H_τ.

Esta realimentación de H_τ en la entrada es lo que distingue fundamentalmente a estas redes y les permite ir guardando en H_τ una especie de resumen de lo que ha ido ocurriendo en la serie a lo largo del tiempo. H_τ es como la historia resumida de la cual se obtiene la predicción del siguiente valor. La predicción final de la red recurrente para esa muestra será $(\hat{P}_{t+1}, \hat{Q}_{t+1}) = (\hat{P}_{\tau=96}, \hat{Q}_{\tau=96})$, es decir, la predicción del último instante de tiempo relativo. En la figura 39 se refleja un ejemplo de arquitectura de una red neuronal recurrente.

Por su parte, en la figura 40 se recoge un esquema de funcionamiento de dicha red constituida por dos perceptrones multicapas con una realimentación intermedia.

Si bien las redes recurrentes han supuesto un paso importante en el tratamiento de series temporales, presentan algunas dificultades y limitaciones técnicas[48]. Es por ello por lo que

48. Las redes recurrentes presentan dos limitaciones principales. La primera es el llamado «desvanecimiento del gradiente»: a medida que la serie temporal se alarga (por ejemplo, si analizamos muchos

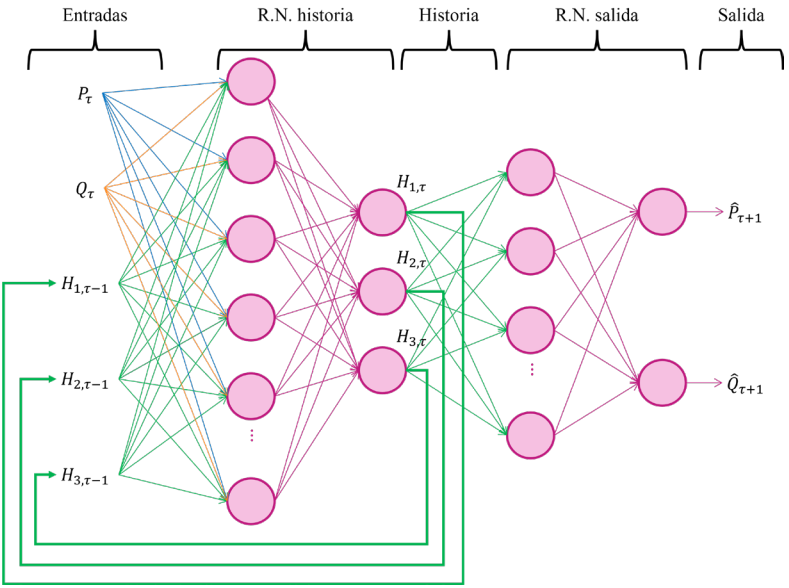

Figura 39. Predicción de las potencias activa y reactiva mediante una red neuronal recurrente

en los últimos años se ha propuesto una arquitectura revolucionaria denominada *Transformers*[35]. Con ella volvemos a

---

cuartos de hora hacia atrás), la información del pasado más remoto tiende a diluirse matemáticamente, haciendo que la red «olvide» los eventos antiguos. La segunda es el cuello de botella secuencial: al estar obligada a procesar los datos paso a paso, uno detrás de otro, la red no puede aprovechar la potencia del cálculo en paralelo de los ordenadores modernos, lo que ralentiza enormemente su entrenamiento con grandes volúmenes de datos.

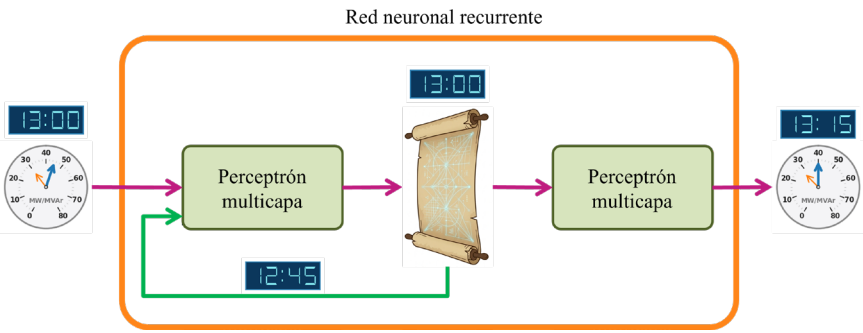

Figura 40. Esquema de bloques de una red neuronal recurrente. Se indica la hora a la que corresponde cada registro (por ejemplo, 13:00) y el valor de las potencias en ese instante (las flechas dentro del dial; azul para la activa, naranja para la reactiva)

recuperar la entrada en paralelo de una ventana completa de datos, pero introduciendo dos modificaciones sustanciales.

En primer lugar, al introducirse todos los datos de la ventana a la vez en paralelo, la red perdería el sentido del orden cronológico si no se solucionara por otros medios. Para evitarlo, se aplica una codificación posicional: una suerte de «sello temporal» matemático que se adhiere a cada dato para que la red sepa exactamente qué lugar ocupa en el calendario de la serie. Este sellado de tiempo es doble. Por un lado, en el momento de aplicación de la ventana deslizante, se incorpora el tiempo absoluto *R*: día y hora de cada instante de muestreo. Por otro, durante el proceso de análisis de cada muestra, se añade también el tiempo o la posición relativa que ocupa cada instante en la ventana *C*.

Así, cada muestra está constituida por 96 elementos, cada uno de los cuales está caracterizado por un vector de cuatro valores: $(P_\tau, Q_\tau, R_\tau, C_\tau)$[49]. Es decir, cada muestra constituye una matriz de entrada, que denominamos U, de dimensión 96×4. Esta estructura se refleja en la figura 41. Así, por ejemplo, la muestra correspondiente a $t = 3$ comprende los vectores de los instantes 3 a 98 (un total de 96) y se predice el consumo del instante siguiente $t = 99$.

La segunda novedad, esta crucial, que aporta la arquitectura de los *Transformers* es que en el procesamiento simultáneo de todos los instantes de la ventana, se calcula internamente una matriz de atención A, en nuestro ejemplo de dimensión 96×96. El elemento a_{ij} de esa matriz nos dice

---

49. Físicamente, nuestra ventana deslizante extrae una matriz de magnitudes eléctricas de 96×2 (potencias P y Q). Aunque por razones de simplificación pedagógica el sellado de tiempo (los valores R y C) aparecen concatenados en la figura constituyendo nuevas columnas, en la práctica el proceso es ligeramente diferente. En primer lugar, la matriz de 96×2 se proyecta, mediante una transformación matricial entrenable, en un espacio de mayor dimensión, por ejemplo 32. De tal forma, los datos de la ventana pasan de ser una matriz de 96×2 a una de dimensión 96×32. A continuación, la información posicional R y C se codifica mediante funciones senoidales en otra matriz también de dimensión 96×32. Las proyecciones de alta dimensión de la demanda eléctrica y de las marcas de tiempo se suman para constituir la matriz definitiva que procesará el *Transformer*, que tiene realmente una dimensión de 96×32. Algunos autores denominan a esa suma la matriz de atributos latentes.

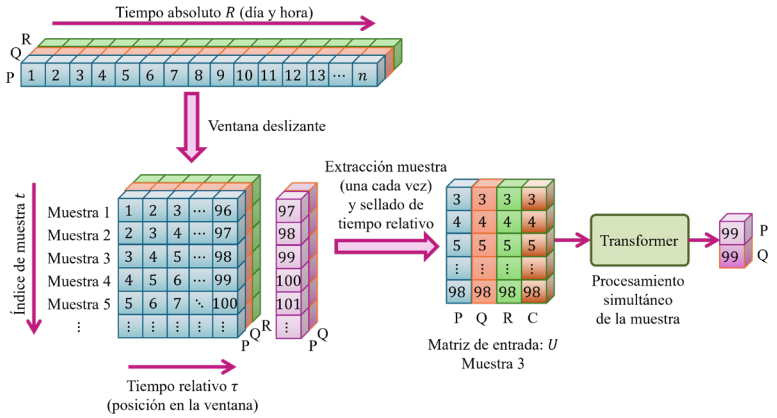

Figura 41. Construcción de la matriz de entrada a un Transformer

la atención o relevancia que el instante *i* (fila) le otorga al instante *j* (columna) en su objetivo de predecir el consumo. Además, la obtención primero de la matriz de atención y, luego con su ayuda, de la predicción de la demanda en el instante siguiente, se realizan mediante la aplicación de varias redes neuronales tipo MLP y algunos elementos muy simples de cálculo con matrices (multiplicación, escalado y normalización). Un esquema global del funcionamiento[50] de un *Transformer* puede verse en la figura 42.

50. Este constituye un esquema simplificado, en forma de grandes bloques, del funcionamiento de un *Transformer*. De una forma más detallada, el cálculo de la relevancia cruzada implica la obtención mediante redes neuronales de las matrices de consultas *Q* y de claves *K*. A partir de ellas, mediante multiplicación matricial, escalado y

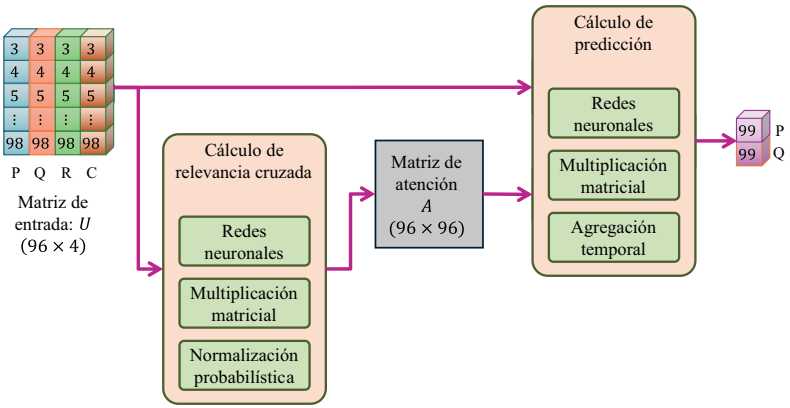

Figura 42. Esquema global del funcionamiento de un Transformer

Si miramos hacia atrás, el camino recorrido puede resumirse en las siguientes etapas, que se ilustran en la figura 43:

1. Recta de regresión. Una hipótesis lineal a la que se le ajustan los parámetros para minimizar la función de pérdida, usando para ello los datos de entrenamiento. Utilizada en el problema del diseño de muebles.
2. Neurona. Al añadirle a la recta de regresión una función de activación no-lineal conseguimos, mediante

---

normalización probabilística (función *softmax*) se obtiene la matriz de atención *A*. Por su parte, el cálculo de la predicción implica la obtención de: la matriz de valores *V*, mediante redes neuronales; la matriz de contexto *Z*, mediante multiplicación matricial; la matriz de estados ocultos *H*, mediante red neuronal; el vector de estados ocultos globales *h*, mediante agregación temporal; y, finalmente, la predicción de la demanda eléctrica mediante una red neuronal.

ingeniería de características, abordar problemas no lineales. Utilizada en el problema de la almazara.

3. Red neuronal superficial. Se apilan varias neuronas con funciones de activación predeterminada (y simple), por ejemplo, RELU. La arquitectura resultante es capaz de aproximar las funciones no lineales que ligan la salida con las entradas sin necesidad de recurrir a la laboriosa ingeniería de características. Utilizada para automatizar el proceso de construcción de atributos en el problema de la almazara.
4. Red neuronal profunda. Concatenando varias capas de neuronas es posible aproximar funciones no lineales complejas que se resisten a ser aproximadas con redes superficiales. Utilizada en el diseño de amortiguadores y en la predicción de demanda eléctrica.
5. *Transformer*. Añadiendo el mecanismo de atención, es decir, la relevancia cruzada entre distintos instantes de una secuencia temporal, es posible resolver problemas de predicción de series temporales con pasados muy largos y/o relaciones muy complejas. No ha sido necesario usarlo en la predicción de la demanda eléctrica[51],

51. Es importante destacar que el uso pleno de un *Transformer* entrenado desde cero no habría sido viable en nuestro problema de demanda eléctrica debido a una limitación crítica: el volumen de datos. Con apenas 10.000 muestras, un *Transformer* tiende inevitablemente al sobreajuste debido a su enorme cantidad de parámetros libres y a la

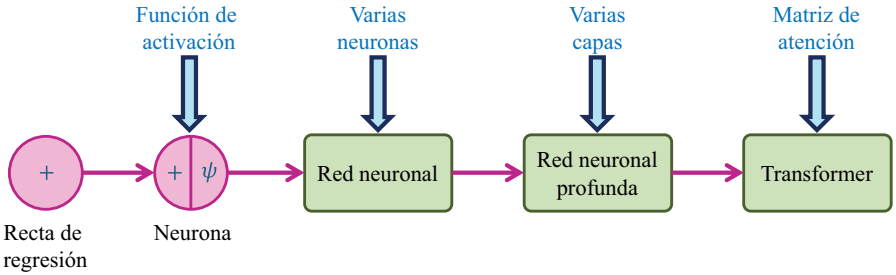

Figura 43. Etapas del camino de recorrido: de la recta al esquema global del funcionamiento de un *Transformer*

pero se ha descrito su arquitectura para poder extender su uso en secuencias de datos más complejas: el procesamiento del lenguaje natural.

Con esto terminamos la parte instrumental del cuarto movimiento, que arranca y termina con lo que Richard Wagner bautizó como la fanfarria del terror (*Schreckensfanfare*)[36].

---

ausencia de un sesgo inductivo secuencial. En la práctica de la ingeniería moderna, ante un volumen de datos tan acotado, la alternativa más eficiente no habría sido diseñar el *Transformer* desde el principio, sino recurrir a la transferencia de aprendizaje (*Transfer Learning*). Los trabajos más recientes del autor de esta lección van, precisamente, en esa dirección[62], utilizando modelos fundacionales preentrenados en series temporales (Chronos[63]) o estructuras tabulares (TabPFN[64]) los cuales ya han «entendido» la naturaleza del tiempo y el comportamiento de las variables tras ser expuestos a miles de millones de datos globales, permitiendo realizar predicciones precisas con muy poco o ningún entrenamiento adicional.

Hemos sido testigos del choque inicial ante la complejidad de los datos masivos e impredecibles de la ingeniería, un caos aparente que parece indomable hasta que las simples redes neuronales primero, y el evolucionado *Transformer* después, han logrado poner orden en la partitura.

Y es en este momento del cuarto movimiento cuando se ponen en pie los cuatro solistas y el coro. El lenguaje, la capacidad de procesar texto de los modernos sistemas de Inteligencia Artificial, no surge de la nada. Ha estado latente, madurando en silencio mientras la ingeniería aprendía a ajustar rectas, a diseñar atributos no lineales y a predecir series temporales complejas. Solo cuando la orquesta llega al límite de sus fuerzas tras el estruendo de la fanfarria, el cuarteto de voces se pone en pie y rompe el silencio. Primero, una sola voz, el barítono, corta el aire: es la primera palabra, el chispazo inicial de atención del modelo. Tras él, todo el coro se levanta para inundar el espacio. El cálculo estadístico se desborda y se transforma en palabra articulada. Veamos cómo. Adentrémonos pues, en el territorio de los Grandes Modelos de Lenguaje (LLM por sus siglas en inglés, *Large Language Model*), el territorio de frontera de la Inteligencia Artificial contemporánea.

En los distintos problemas que hemos abordado hasta ahora siempre hemos trabajado con valores numéricos: alturas, temperaturas, velocidades, potencias, ... Para poder introducir la palabra en este mismo esquema, el primer paso

obligado es convertirla en valores numéricos. Para explicar cómo lograrlo, tomemos un desvío pedagógico. Imaginemos que queremos predecir algún tipo de comportamiento relativo a un colectivo de personas. Cada muestra sería, en este caso, un individuo. Una forma elemental de asignarle un identificador numérico único sería utilizar, por ejemplo, su número de Documento Nacional de Identidad (DNI)[52]. Sin embargo, ese número secuencial nos dice poco o nada sobre la persona; la etiqueta unívocamente, pero no la caracteriza.

Por ello, además o en vez de este identificador, la ingeniería prefiere emplear un conjunto de rasgos que, de forma conjunta, identifican unívocamente al individuo, dándonos de paso mucha más información sobre su naturaleza. Así, por ejemplo, podríamos indicar su edad, altura, peso, pulsaciones cardíacas, salario o número de hijos. Imaginemos, por simplicidad, que seleccionamos 32 rasgos diferentes y que a cada uno de ellos le asignamos un valor discreto normalizado en una escala del 1 al 10. Obtendríamos así la radiografía de una persona condensada en un conjunto de 32 valores, lo que denominamos un vector de dimensión 32, tal como se ilustra en el ejemplo de la figura 44. Hagamos un cálculo rápido: si cada uno de los 32 rasgos admite solo esos 10 valores posibles, el conjunto de combinaciones teóricas es de 10^{32}. Un uno

52. No consideramos en este ejemplo el caso de posibles números de DNI duplicados por error.

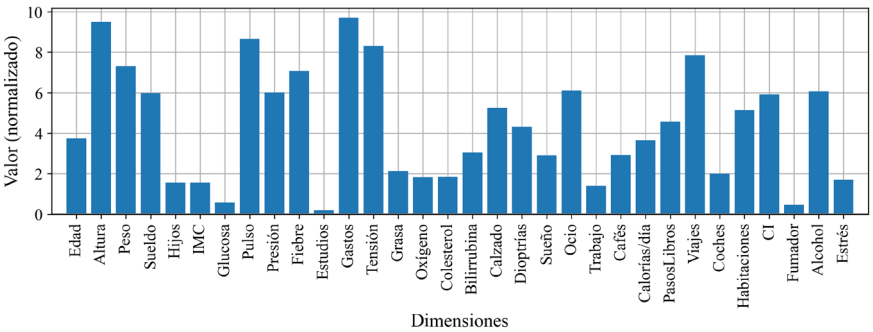

Figura 44. Caracterización de una persona mediante un conjunto de 32 atributos

seguido de 32 ceros. Es un espacio de posibilidades colosal, inmensamente mayor que la población actual del planeta (algo menos de 10^{10} personas) o que el número estimado de granos de arena que alfombran todas las playas y desiertos de la Tierra (del orden de 10^{20}). Suficientes, por tanto, para identificar de forma unívoca a un individuo.

Demos el paso ahora a las palabras: ¿cómo las identificamos dentro de una máquina? ¿cómo les asignamos un valor numérico único? Una primera solución consiste en construir un diccionario que albergue la totalidad de los términos y asignarles arbitrariamente un número secuencial, por ejemplo, siguiendo el orden alfabético. Este diccionario digital debe ser masivo, ya que no solo contiene los nombres comunes sino también los propios y, asimismo, debe tener la capacidad de gestionar todas las posibles

variantes[53] de una palabra[54]. En los ejemplos que manejaremos en esta parte de la lección usaremos un modelo de lenguaje lo suficientemente compacto como para poder ejecutarse en un ordenador personal[55]. En dicho modelo,

---

53. Esto incluye las conjugaciones verbales, género y número, aumentativos, diminutivos, superlativos, prefijos, sufijos y palabras compuestas.

54. En los sistemas de procesamiento de lenguaje suelen utilizarse fragmentos de palabras, denominados *tokens*. Hay varias razones técnicas para ello entre las que destacan tres: 1) reduce drásticamente el tamaño del vocabulario que el modelo debe memorizar matemáticamente; 2) permite gestionar con elegancia raíces, prefijos y sufijos; y 3) evita el temido problema de las palabras desconocidas, ya que cualquier término nuevo puede descomponerse en fragmentos ya conocidos. Por facilitar la comprensión del texto, haremos la simplificación teórica de que la unidad de procesamiento es la palabra y no el *token*.

55. El sistema seleccionado es Qwen2.5-1.5B-Instruct, un modelo de lenguaje de código abierto desarrollado por Alibaba Group y lanzado en septiembre de 2024. Se trata de una arquitectura *Transformer* de 28 capas que cuenta con 1500 millones de parámetros (de ahí la nomenclatura «1,5B», del inglés *billions*) y que ha sido optimizado específicamente para seguir instrucciones en formato de diálogo («Instruct») [65]. Dentro del panorama competitivo de los LLM, la serie *Qwen2.5* se ha consolidado como el principal referente de código abierto junto a la familia *Llama* de *Meta*. En los bancos de pruebas estándar (*benchmarks*) de la industria para modelos de escala reducida (sub-3B), Qwen2.5-1.5B supera de manera consistente a competidores directos como *Llama 3.2-1B* o *Gemma-2B* en capacidades de programación, comprensión de estructuras de datos (como tablas y JSON) y razonamiento matemático. Se ha seleccionado este modelo específico para la

el diccionario contiene un vocabulario fijo de exactamente 151.643 entradas[56].

Fijémonos ahora en una palabra; por ejemplo, la palabra «profesor». En el diccionario ocupa la posición 21.826. Este sería como el número de DNI de la palabra. Pero, al igual que con las personas, en vez de con un número aislado, preferimos identificar a la palabra por una serie de rasgos. Por ejemplo, podríamos caracterizar la palabra por su género, su número, su humanidad (si es un ser vivo o un objeto inanimado), su rol social, su carga emocional (si es una palabra positiva o negativa), o su abstracción (si es algo físico que se puede tocar o un concepto teórico). Al combinar todos estos rasgos, el frío número del diccionario se transforma en un perfil de identidad polifacético. En los modernos modelos de lenguaje se usan centenares o miles de rasgos para describir una palabra. En el modelo de nuestro ejemplo cada

---

presente lección debido a su equilibrio óptimo entre ligereza computacional y rigor en el procesamiento del idioma español, campo donde los modelos de origen anglosajón suelen mostrar una menor densidad léxica debido a la asimetría en sus datos de entrenamiento.

56. Esa dimensión exacta (151.643 tokens) responde a la naturaleza multilingüe y global del modelo. Mientras que un sistema monoidioma requiere apenas un tercio de esa cifra, un espacio semántico de este tamaño permite al algoritmo segmentar texto con idéntica eficiencia matemática tanto en español o inglés como en lenguas de alfabetos no latinos (como el chino, el árabe o el cirílico) e incluso código de programación, optimizando la capacidad de compresión global de la red neuronal.

palabra se despliega en un vector (un conjunto) de 1536 dimensiones[57], cuyo significado lingüístico no es evidente a primera vista, pero tampoco completamente opaco, siendo el objeto de interesantes y muy recientes investigaciones dentro de una disciplina denominada interpretabilidad mecanicista[37], [38], [39]. En la figura 45 se refleja el valor de las 100 primeras dimensiones de la palabra «profesor»[58].

---

57. Internamente, al procesar la información en cada capa, estas 1536 dimensiones se proyectan en 12 grupos (denominados cabezas de atención), cada uno de ellos especializado en diferentes aspectos o rasgos gramaticales[66], [67]. Por ejemplo, rasgos relacionados con el género, con el rol social, con la jerarquía, etc. Cada uno de estos grupos contiene 128 dimensiones, que podemos interpretar como subrasgos o matices. El número de 128 dimensiones por cabeza de atención es una potencia de dos (2^7), un valor que se adapta muy bien a las arquitecturas de computación basadas en códigos binarios. En cualquier caso, tanto los valores de 12 como 128 son hiperparámetros de la arquitectura que se determinan de igual forma que el resto de los hiperparámetros que han ido apareciendo a lo largo de la lección.

58. En realidad, en un *Transformer* los vectores corresponden a *tokens*, no a palabras y, además, estas palabras/*tokens* tienen un contenido semántico que depende del contexto (no tiene la misma carga semántica la palabra «banco» para sentarse, que «banco» como institución financiera). Para obtener el vector correspondiente a la palabra «profesor», esta se ha encapsulado como único elemento de un *prompt* (petición) y se ha obtenido el vector correspondiente al último *token* (que resume el contenido del *prompt*) en el canal de comunicación interno (el flujo residual de la capa 14 de 28) del *Transformer*. Según los resultados obtenidos mediante estudios de interpretabilidad

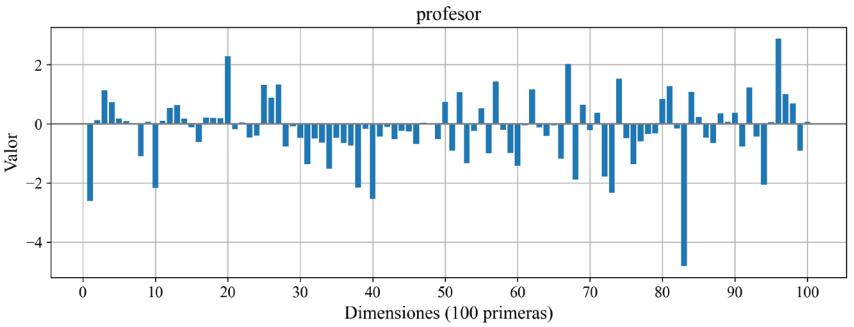

Figura 45. Valores de las 100 primeras dimensiones correspondientes a la palabra «profesor»

Una forma de visualizar las múltiples dimensiones semánticas de una palabra es comparar un grupo de palabras entre sí y proyectar sus diferencias en dos dimensiones. Adaptando al ámbito académico un ejemplo ya clásico[40] vamos a proyectar las 1536 dimensiones de las palabras «profesor» y «alumno» sobre un eje horizontal que represente el rol docente de cada una de ellas. Si ahora incluimos también sus variantes en femenino («profesora» y «alumna»), podemos visualizarlas sobre un eje vertical que represente el género[59].

---

mecanicista, las capas iniciales de un *Transformer* están orientadas a representar las relaciones sintácticas, las intermedias a obtener la carga semántica, y las finales a la predicción de la siguiente palabra.

59. Ninguna de las 1536 dimensiones se corresponde con el género, el número o el rol docente. Las dimensiones no tienen un valor semántico definido. Por ello al proyectarlas sobre dos ejes con significado semántico

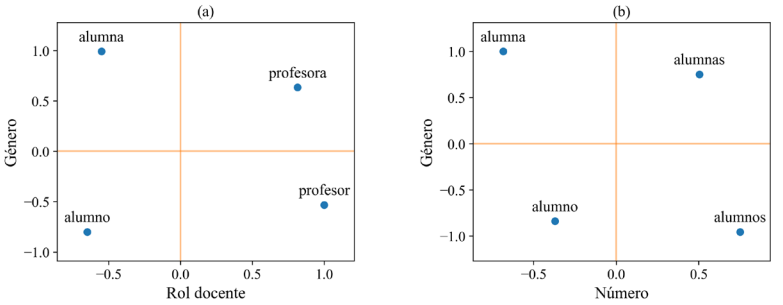

Figura 46. Proyecciones en dos dimensiones de variantes de las palabras «profesor» y «alumno»

El resultado se refleja en la figura 46-a. Si combinamos las variaciones de género y número de la palabra «alumno», las cuatro representaciones multidimensionales resultantes las podemos plasmar en los correspondientes ejes, tal como aparece en la figura 46-b.

Y ahora que ya sabemos representar palabras, abordemos la forma de representar frases. Pero estas no son más que una sucesión de palabras[60]. Por ello, el arranque vocal de la

concreto obtenemos aproximaciones que hacen que los puntos no se distribuyan en un rectángulo perfecto.

60. Dado que los *Transformers* procesan la información dividiendo las palabras en *tokens*, para mantener la simplicidad pedagógica de la gráfica y representar palabras completas, hemos tomado como valor de cada palabra la representación vectorial de su último *token*, que es el que por diseño de la arquitectura acumula el significado completo de los componentes anteriores.

novena sinfonía, el grito que rompe la fanfarria del terror, la frase «¡Oh amigos, no en esos tonos! ¡Entonemos otros más agradables y llenos de alegría!», se puede representar gráficamente como una serie temporal multivariable en la que se reflejan las variaciones de los valores de cada una de sus dimensiones a medida que la frase se despliega. La evolución de las cuatro más significativas[61] se recoge en la figura 47. Cada punto en el tiempo no representa la palabra aislada, sino el sentido acumulado de la frase hasta ese instante. Esta gráfica es análoga a la de la figura 28 con la evolución de las potencias activa y reactiva a lo largo del día, pero escalando de dos a 1536 dimensiones.

Pero ¿cómo podemos interpretar las múltiples dimensiones en las que se despliega la frase? Para tratar de imaginarlo, avancemos un poco en el cuarto movimiento y situémonos en el tema del abrazo fraterno, ese clímax ecuménico en el que Schiller y Beethoven nos convocan a una universal fraternidad humana con los versos «¡Abrazaos, millones! ¡Este beso para el mundo entero!». En la figura 48 representamos las primeras 32 dimensiones correspondientes a esa frase en el modelo de lenguaje. No hace falta detenerse en los detalles; basta con observar la textura de la imagen general.

---

61. Por dimensiones más significativas entendemos aquellas que más cambian a lo largo de la frase. Técnicamente, son las dimensiones que presentan una mayor varianza estadística.

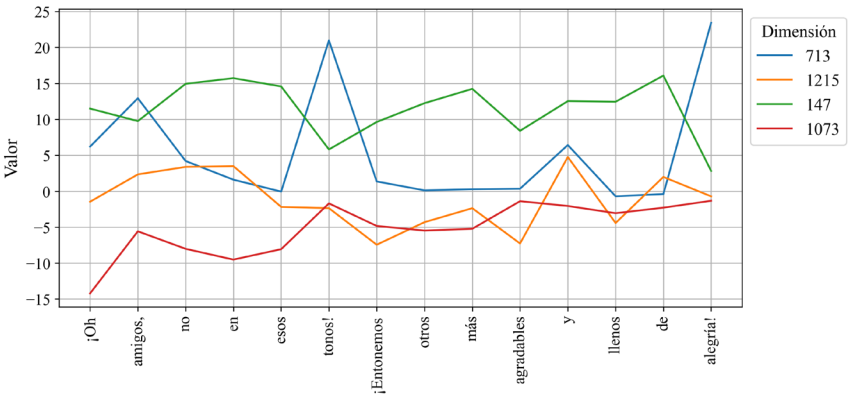

Figura 47. Evolución de los valores de las cuatro dimensiones más significativas en la frase de inicio de la parte vocal de la novena sinfonía de Beethoven

Si acudimos ahora a la partitura física de la *Novena Sinfonía*[41], concretamente en el clímax absoluto de este cuarto movimiento (en torno al compás 854), descubriremos que Beethoven despliega sobre el papel una matriz polifónica de exactamente 32 dimensiones o líneas musicales simultáneas, tal como se muestra en la figura 49.

Al comparar ambas imágenes, la analogía visual resulta sobrecogedora. Podemos ahora imaginar a cada una de las 1536 dimensiones como un instrumento individual en una colosal orquesta invisible: los violines primeros, el oboe, las flautas, los timbales, los solistas o las voces del coro. Cada instrumento interpreta una línea matemática diferente, un matiz sutil y especializado de la información. Ninguno de

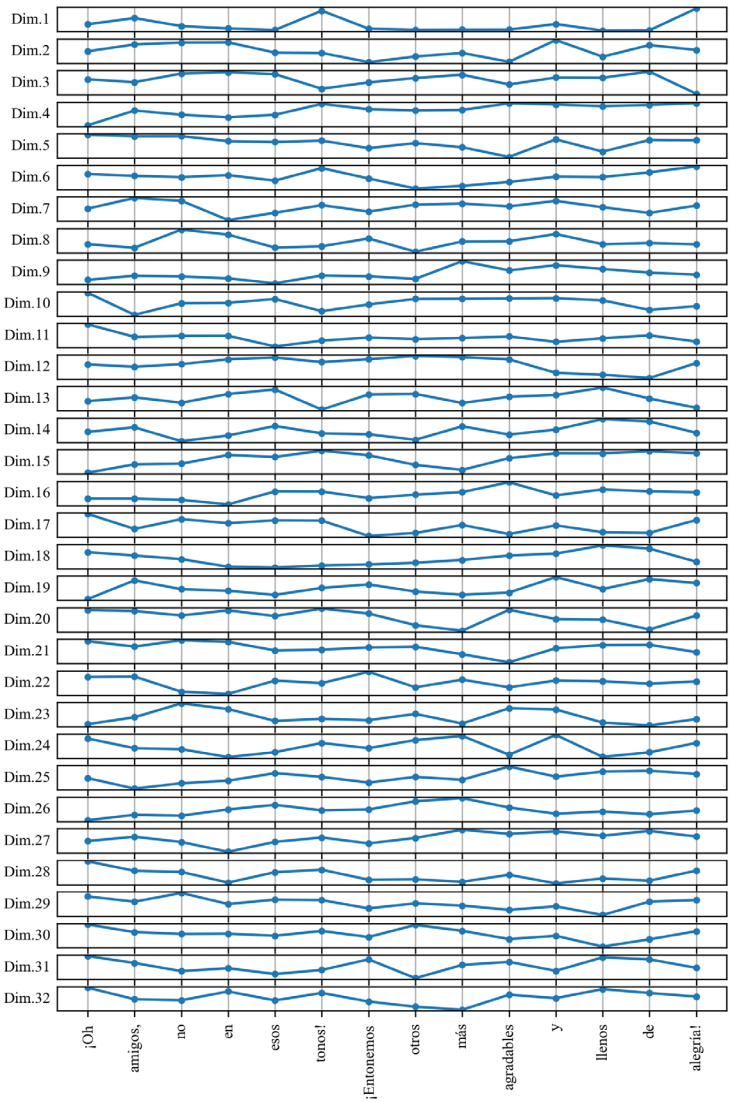

Figura 48. Evolución de los valores de las 32 primeras dimensiones en la frase del coro que invoca el abrazo fraterno

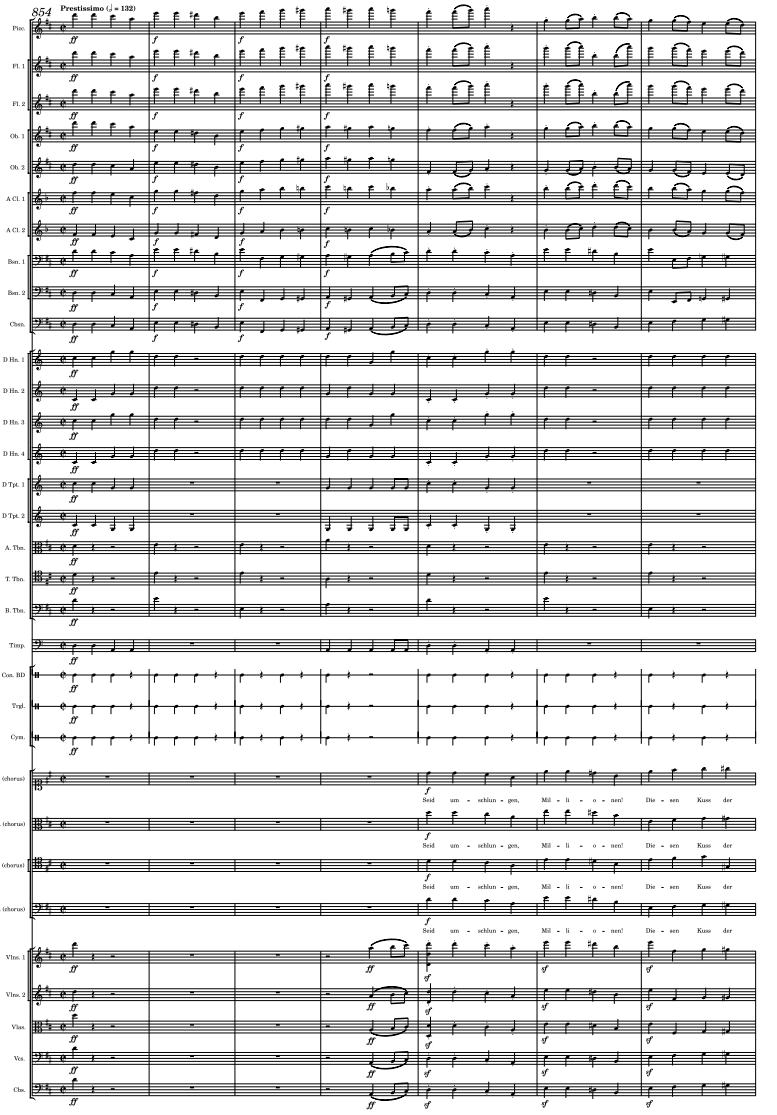

Figura 49. Partitura correspondiente a la sección del cuarto movimiento donde comienza el tema del abrazo fraterno

ellos posee el significado completo de la frase por separado; la significación profunda de la idea solo emerge cuando todos los instrumentos ejecutan sus trayectorias simultáneamente, provocando una sensación conjunta, una armonía de datos que se despliega en el tiempo.

En el proceso de representación numérica de una palabra, la primera tarea de la máquina consiste en desmenuzarla en sus múltiples componentes semánticos (en concreto 1536), reduciéndola a verdaderos átomos de significación aislada[62]. Al contemplar este proceso de disección computacional, en el que hemos triturado una palabra hasta obtener un polvillo semántico, no puedo dejar de recordar el famoso soneto de Quevedo[63] que me permito evocar aquí: «serán ceniza, mas tendrá sentido».

---

62. Con «átomos de significación» no se hace referencia a que cada una de las 1536 dimensiones posea un significado unívoco o interpretable de forma aislada. En el procesamiento del lenguaje natural, el significado es una propiedad emergente, análoga a los píxeles de una imagen digital. La intensidad numérica de un único píxel aislado no contiene información sobre el contenido de la foto; no representa por sí misma una forma, un objeto o un color con sentido. Sin embargo, el conjunto ordenado de todos los píxeles hace emerger la imagen completa. De igual modo, las múltiples dimensiones de la palabra son valores abstractos que solo adquieren valor semántico en su conjunto.

63. «Amor constante, más allá de la muerte». Francisco de Quevedo[68].

El segundo paso de esta arquitectura (el mecanismo de atención) consiste precisamente en matizar, expandir y alterar ese sentido original al poner la palabra en íntimo contacto con las demás. Tras este profundo cruce de miradas estadísticas, las coordenadas originales se magnetizan y se tiñen de un matiz único. De nuevo, la ingeniería rememora a Quevedo en el clímax del algoritmo: esos vectores, modificados por el contexto, «polvo serán, mas polvo enamorado»; un polvo matemático preñado de la presencia de las palabras vecinas.

Permítaseme una nueva analogía, prometo que ya la última para tratar de ilustrar el siempre elusivo mecanismo de la atención en los *Transformers*. Anteriormente, veíamos cómo un ser humano podía ser descrito vectorialmente mediante distintas variables entre las que incluíamos algunas biológicas como el pulso o la presión arterial. Pues bien, la atención es exactamente el equivalente algorítmico a la célebre canción de Juan Luis Guerra: cuando una palabra se encuentra en presencia de un contexto determinado, sus características internas se alteran por completo. Al igual que a un enamorado «le sube la bilirrubina» cuando mira a la persona amada, modificándose su estado biológico por mero cambio de contexto, a la palabra 'tonos' se le alteran sus 1536 dimensiones latentes al verse acompañada por los términos 'amigos' o 'alegría'. Ya no es una palabra aislada; ahora es una palabra 'emocionada' y condicionada por su entorno.

Y ahora que ya sabemos representar frases en forma de series temporales, e intuir la naturaleza de sus múltiples dimensiones, encaremos el problema de predecir la siguiente palabra. Si antes hemos sido capaces de predecir la demanda de electricidad, una serie en dos dimensiones (potencia activa y reactiva), hacerlo en 1536 es, en el fondo, tan solo una cuestión de escala. Podemos emplear las mismas técnicas y herramientas, adaptadas a la mayor complejidad de la tarea. Y utilizaremos las mismas redes neuronales, en este caso en su arquitectura de *Transformers* que, como sabemos, no son más que una ingeniosa combinación de rectas de regresión y funciones no lineales.

¿Cuál es el aparente truco de magia detrás de la capacidad de un modelo de lenguaje para generar texto con sentido? Haber leído mucho. Veámoslo con un ejemplo práctico. Si yo les digo: «En un lugar de la Mancha, de cuyo nombre no quiero acordarme, [...]» todos sabrán sin duda predecir las siguientes palabras: «[...] no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor.». ¿El secreto de esta capacidad de predicción? Que todos hemos leído, estudiado en la escuela, o asimilado a través de la cultura popular española y universal, el famosísimo comienzo de *El Quijote*.

Exactamente eso es lo que le ocurre a un modelo de lenguaje: además de poseer una arquitectura matemática muy compleja, ha leído muchísimo. El entrenamiento de

los modelos de lenguaje más avanzados se realiza hoy sobre corpus textuales que alcanzan el orden de varias decenas de billones de palabras (en la literatura anglosajona, *trillions*)[64]. Para que nos hagamos una idea, esto equivale aproximadamente a 300 millones de libros; un volumen superior al de las mayores bibliotecas del mundo. Representa, por ejemplo, diez veces la totalidad de los ejemplares de la Biblioteca Nacional de España (estimados en más de 35 millones[42]) y cien veces los fondos de la Biblioteca de la Universidad de Sevilla (estimados en más de 2.5 millones[43])[65].

La representación gráfica de este proceso de predicción aplicado al inicio de *El Quijote* se ilustra en la figura 50. Esta imagen guarda un estrecho paralelismo con la que ya

---

64. Por ejemplo, los modelos de frontera actuales, como la familia *Llama 4* de Meta (2025)[69] o *DeepSeek-V4* (2026)[70] se han entrenado utilizando corpus masivos que superan los 30 billones de *tokens* (el equivalente aproximado a 25 billones de palabras[71]).

65. No obstante, la comunidad científica advierte ya del inminente «agotamiento de los datos» (*data wall*): el ritmo de escalado de los grandes modelos de lenguaje (LLM) está consumiendo los inventarios globales de texto público de alta calidad a una velocidad muy superior a la que la humanidad genera nuevo contenido. Diversas investigaciones[72] estiman que las fuentes de datos textuales con un nivel de rigor y riqueza lingüística suficiente (como libros, artículos científicos y repositorios académicos) se habrán explotado por completo antes de finalizar la presente década, lo que obliga a la disciplina a pivotar hacia el uso de datos sintéticos y arquitecturas computacionales más eficientes.

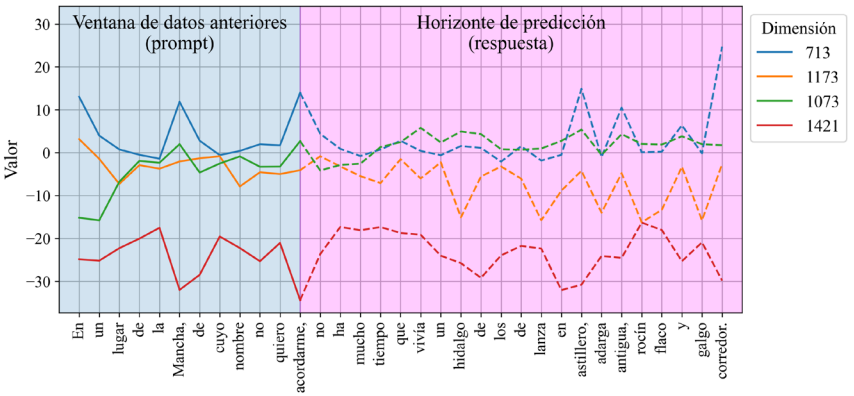

Figura 50. Predicción de las siguientes palabras de la frase de comienzo de *El Quijote*

usamos en la figura 36, página 87, para la predicción de la demanda eléctrica, con dos diferencias significativas. La primera es cuantitativa: el espacio de representación intermedio ha pasado de 2 a 1536 dimensiones. La segunda diferencia es semántica: ahora los valores de la «ventana de datos anteriores» se corresponden con las palabras introducidas en la petición (*prompt*), mientras que el horizonte de predicción contiene la respuesta: la estimación de las palabras venideras.

Pero sigamos con el experimento. Si yo les pido que continúen la frase «Todas las familias felices se parecen unas a otras, [...]», solo algunos recordarán las palabras con las que Lev Tolstói da comienzo a su novela Ana Karenina y podrán concluir la oración con un «[...] pero cada familia infeliz lo es a su manera». Sin embargo, ninguno de los grandes modelos

de lenguaje tiene problemas en completarla porque todos han «leído» pacientemente el canon de la literatura rusa.

Para no quedarnos con la amarga sensación de haber sido derrotados por la máquina hagamos un último experimento. Si les pido que continúen el verso: «Mírala cara a cara [...]», la inmensa mayoría de nosotros[66] seremos capaces de completar la estrofa recitando «[...] que es la primera», la célebre sevillana compuesta en 1986 por Manuel Melado. Sin embargo, las respuestas de las máquinas actuales, resumidas en la tabla 1, producen una mezcla de sorpresa y alivio. Incluso, quizás, son capaces de arrancarnos una sonrisa.

Tabla 1. Continuación de la frase «Mírala cara a cara [...]»

Modelo	Origen	Respuesta generada
Gemini	EE. UU.	«... que es tu tuya».
ChatGPT	EE. UU.	«... que no te va a morder».
DeepSeek	China	«... y no apartes la mirada, que en sus ojos está la verdad».
Qwen	China	«... y verás que no es tan fea».
Mistral	Francia	«... y verás el alma de una valiente».
Original	España	«... que es la primera».

66. En esta afirmación se asume que el público asistente a esta lección o el lector de este volumen forman parte del entorno cultural de la Universidad de Sevilla.

La explicación científica de este naufragio algorítmico es evidente: la lírica de esa sevillana no formaba parte del corpus de entrenamiento de ninguno de estos colosos tecnológicos, mientras que sí impregna de forma indeleble el entorno cultural de muchos de nosotros.

De ello se deduce que el éxito (o el fracaso) de un modelo de lenguaje no radica únicamente en poseer una sofisticada arquitectura de *Transformers* plagada de miles de millones de parámetros. Al contrario de lo que se postulaba en los albores de esta tecnología[44], no basta con hipertrofiar la red añadiendo más y más parámetros de forma aislada. Según el célebre estudio Chinchilla de 2022[45], ante un incremento en el presupuesto de computación, la ingeniería debe repartir el esfuerzo de manera simétrica: un 50 % del crecimiento debe destinarse a expandir el tamaño del modelo (sus parámetros) y el otro 50 % a ampliar, en idéntica proporción, el volumen y la riqueza de los datos de entrenamiento[67].

Y para finalizar este cuarto movimiento vamos a hacerle al modelo de lenguaje una pregunta concreta y vamos a analizar el proceso que sigue la máquina para ofrecernos una respuesta. La cuestión elegida es de plena vigencia:

---

67. El mencionado estudio Chinchilla determinó empíricamente que la proporción óptima para mantener el equilibrio computacional se sitúa en torno a los 20 *tokens* de texto procesados por cada parámetro de la red neuronal.

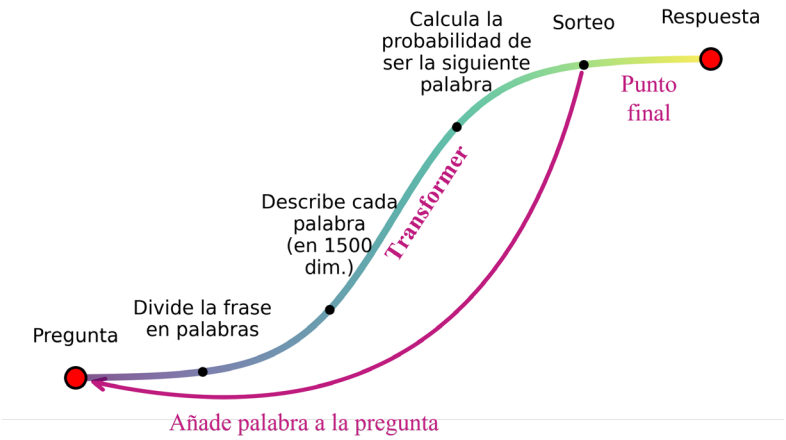

Figura 51. Proceso recursivo de generación de una respuesta

Con una frase muy breve: «¿Cómo debe encarar la Universidad los desafíos de la inteligencia artificial?»

El proceso que se desencadena en el interior del modelo de lenguaje, resumido en la figura 51, se articula en las siguientes etapas secuenciales:

1. Segmentación: divide la frase original en las 19 palabras que la constituyen, incluyendo los signos de interrogación y de puntuación[68].

68. Por razones divulgativas, mantenemos en esta sección la convención de hablar de «palabras». Como ya hemos comentado anteriormente, la división de la frase no se hace en palabras sino en los

2. Identificación: busca cada palabra de forma unívoca entre las 151.643 entradas de su diccionario y le asigna su identificador numérico o índice secuencial. Por ejemplo, a la palabra «Universidad» le corresponde la posición 66.513.
3. Caracterización: traduce ese índice estático en una «descripción» en forma de 1536 dimensiones. En esta etapa, el diccionario actúa como un mapa geométrico donde a cada entrada le acompaña su correspondiente representación multidimensional[69]. La descripción así obtenida representa a la palabra de forma aislada y cruda; es decir, sin haber considerado aún el contexto del resto de la frase. Al concluir este paso se genera una tabla de 19 filas (una por palabra) y 1536 columnas. Esta es la matriz de entrada[70] que se inyecta al *Transformer*, equivalente a la matriz *U* de las figuras 41 y 42.

---

siguientes 25 *tokens*: `Con, una, frase, muy, breve, :, ¿, Cómo, debe, enc, ar, ar, la, Universidad, los, des, a, fí, os, de, la, intelig, encia, artificial, ?`.

69. En la teoría formal de redes neuronales, la obtención de esta primera representación de un *token* en el espacio multidimensional se consigue mediante la multiplicación de un vector *one-hot* (lleno de ceros y un uno en la posición del índice) por la tabla de búsqueda de representaciones vectoriales (*embeddings*), una matriz de 151643 filas y 1536 columnas cuyos valores son obtenidos en la fase de entrenamiento del modelo.

70. Para que la red capture el orden del discurso, cada palabra debe incorporar también información sobre su posición en la frase.

4. Cálculo probabilístico: mediante el mecanismo de atención, donde cada palabra interactúa simultáneamente con todas las demás para impregnarse de su contexto, el *Transformer* recalcula la predicción del siguiente elemento. Pero en lugar de emitir una única palabra, lo que hace es decirnos la probabilidad de cada una de las palabras de su diccionario de ser la siguiente. Obtenemos, por tanto, un vector con 151.643 probabilidades.
5. Muestreo estadístico: a partir de aquí el proceso puede proseguir con ligeras variantes. El método estándar consiste en aislar un subconjunto de las palabras más probables (por ejemplo, las cinco primeras, una técnica denominada *Top-K sampling*) y realizar un «sorteo» o muestreo aleatorio, ponderado por la probabilidad de cada una, para elegir la predicción definitiva.
6. Recursividad: acto seguido, la palabra seleccionada en el sorteo se añade al final del texto de entrada (*prompt*) y el algoritmo vuelve a ejecutar de forma recursiva todo el itinerario desde el paso 1.
7. Finalización: este bucle automatizado se detiene en el instante preciso en el que el *Transformer*, tras evaluar

---

Esto se logra mediante la inyección de una matriz posicional que tiene igualmente 19 filas y 1536 columnas. La matriz definitiva que procesará el *Transformer*, habitualmente denominada matriz de atributos latentes, es la resultante de sumar algebraicamente la matriz de entrada *U*, y esta matriz posicional.

la distribución estadística del contexto acumulado, selecciona como palabra más probable el elemento de control que representa el punto final de la secuencia.

La elección probabilística de la siguiente palabra dota al modelo de lenguaje de un indudable carácter aleatorio. Alrededor de cada término se despliega un árbol de posibilidades que se va ramificando de manera exponencial a medida que el discurso avanza. Imaginemos que, para la frase del ejemplo, restringimos el escenario eligiendo únicamente entre las dos palabras más probables.

Como se refleja en la figura 52, estas son «La» y «¿Qué» con un 85 % y un 5 % de probabilidad respectivamente[71]. Si en el sorteo ponderado resultara elegida la palabra «¿Qué», esta se añade de inmediato al *prompt* y el nuevo bloque vuelve a inyectarse en el *Transformer*. El veredicto de la red neuronal en este segundo paso nos indica que las dos palabras más probables son «estrategias» (31 %) y «medidas» (28 %). Bajo esta dinámica secuencial se va modelando, eslabón a eslabón, la respuesta definitiva.

Si aplicamos el criterio de hacer un sorteo entre las 5 palabras más probables, la respuesta completa generada por el algoritmo es, como se muestra en la figura 53, la siguiente:

71. El resto hasta el 100 % corresponde a la probabilidad del resto de palabras del diccionario.

Con una frase muy breve: ¿Cómo debe encarar la Universidad los desafíos de la inteligencia artificial?

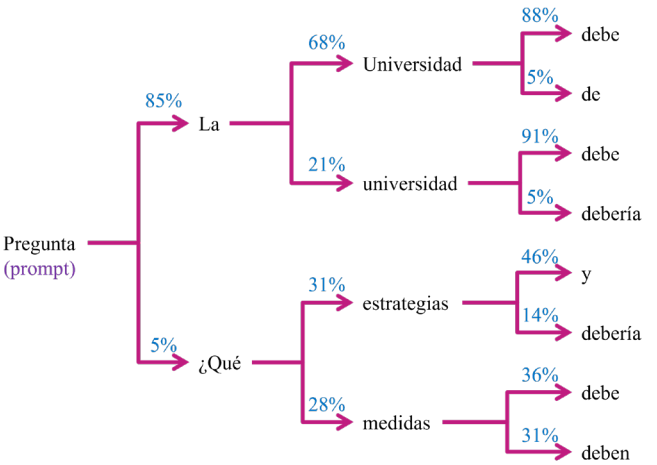

Figura 52. Árbol de predicción a partir de un *prompt*

> La Universidad debe enfocarse en la formación de profesionales capaces de manejar y [*sic*] integrar la IA de manera ética y responsable, promover la investigación en este campo y fomentar la colaboración entre la academia y la industria para desarrollar soluciones que beneficien a la sociedad.

En el gráfico se representan las cinco opciones que ofrece el modelo en cada iteración junto con su probabilidad relativa (expresada visualmente mediante la longitud de la barra correspondiente) ordenadas de mayor a menor. En color rojo se destaca cuál de las cinco palabras fue la

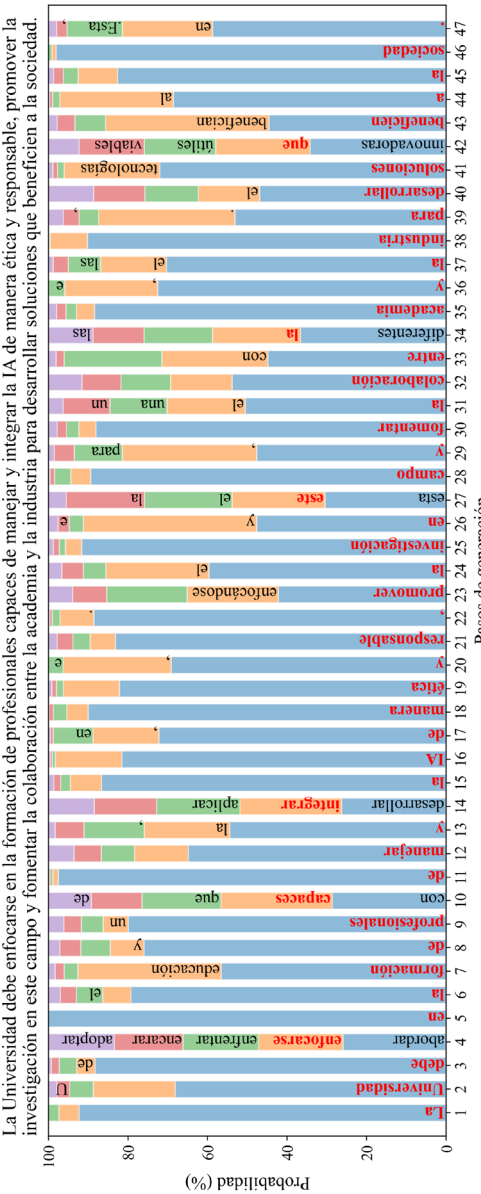

Figura 53. Secuencia de elaboración de una respuesta ante un *prompt*

favorecida en el sorteo y, por tanto, la elegida para agregarse al *prompt* que generará el siguiente elemento.

Como puede deducirse de la explicación anterior, un modelo de lenguaje es, en primera instancia, lo que en la literatura científica se conoce bajo el llamativo apelativo de «loro estocástico»[46] (que para entendernos mejor podríamos traducir como «loro probabilístico» o «estadístico»), un término que viene a denunciar que estos sistemas se limitan a reflejar, combinar y promediar la inteligencia colectiva de la humanidad plasmada en internet (incluidos, de forma inevitable, todos nuestros sesgos, inconsistencias y prejuicios).

Los modelos de lenguaje no buscan la verdad, aunque son capaces de aparentarla con un virtuosismo asombroso. Ante la lectura de sus respuestas, uno no puede evitar recordar el viejo aforismo italiano de Giordano Bruno: «se non è vero, è ben trovato» (si no es verdad, al menos está magníficamente construido).

Pero este virtuosismo formal, este imponente *Transformer* capaz de encadenar palabras con asombrosa coherencia, es todavía una fiera salvaje de la estadística: lo que en la industria se denomina un modelo fundacional o base[72].

---

72. Un *modelo fundacional* o *base* (como las versiones nativas de las familias Gemini de Google, GPT de OpenAI o LLaMA de Meta antes de ser refinadas) ha aprendido exclusivamente a realizar la tarea de modelado de lenguaje autorregresivo (predecir el siguiente token). Si se le introduce el *prompt* «¿Cuál es la capital de Francia?», el modelo base

Es un prodigio de la sintaxis, pero carece de modales, no sabe obedecer órdenes y desconoce sus propios límites. Para transformar este motor probabilístico en un producto comercial útil y seguro (los modelos de lenguaje convencionales con los que interactuamos a diario), la ingeniería actual debe someter esa verosimilitud a un proceso secuencial de domesticación y empaquetado.

Este viaje de refinamiento atraviesa cuatro etapas críticas, reflejadas en la figura 54, que van puliendo la conducta del modelo[73]: en primer lugar, se especializa su conocimiento

---

no responderá «París», sino que probablemente continuará el texto escribiendo «¿Cuál es la capital de España? ¿Cuál es la de Italia?», porque estadísticamente está optimizado para continuar listas de preguntas, no para actuar como un asistente.

73. El proceso de empaquetado y postentrenamiento se compone de las siguientes fases computacionales: 1) Ajuste Fino (*Fine-Tuning* / Modelo de Dominio): Reentrenamiento ligero del modelo base con un corpus especializado (médico, legal, de ingeniería) para alterar la distribución de probabilidad de sus pesos hacia un vocabulario técnico concreto.

2) Ajuste Fino Supervisado (*Supervised Fine-Tuning*, SFT / Modelo Instructivo): El modelo se entrena con miles de ejemplos en formato de «Pregunta/Respuesta» escritos por humanos. Aquí el modelo aprende la estructura conversacional y a comportarse como un asistente que obedece instrucciones.

3) Alineamiento y Guardarraíles (*Alignment* / Modelo Alineado): Se utilizan técnicas como el Aprendizaje por Refuerzo a partir de Retroalimentación Humana (RLHF) o Directa (DPO) para mitigar alucinaciones y sesgos. En esta fase se inyectan dinámicas de razonamiento

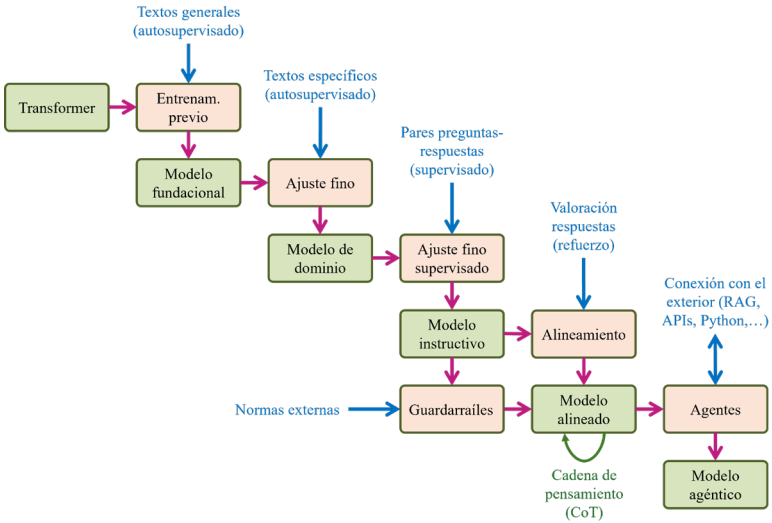

Figura 54. Del modelo fundacional a los modelos de lenguaje comerciales

mediante un ajuste fino para adaptarlo a un dominio concreto; en segundo lugar, se le enseña a escuchar y responder mediante

---

explícito como la Cadena de Pensamiento (*Chain-of-Thought*, CoT), forzando al modelo a desglosar los problemas paso a paso antes de emitir la respuesta final, y se imponen filtros de seguridad (guardarraíles) que bloquean peticiones peligrosas.

4) Modelos Agénticos e Integración (*Tool Use* / RAG): La fase final de producto donde el modelo deja de estar aislado en su matriz estática. Se le dota de API para conectarse a internet, calculadoras, entornos de ejecución de código (Python) y sistemas de Generación Aumentada por Recuperación (RAG) para consultar bases de datos externas en tiempo real, adquiriendo la capacidad de actuar como un agente autónomo en el entorno digital.

un ajuste fino supervisado (SFT), transformándolo en un modelo instructivo; en tercer lugar, se le dota de valores y seguridad mediante técnicas de alineamiento y guardarraíles para evitar derivas tóxicas; y, finalmente, se le abren las puertas al exterior convirtiéndolo en un modelo *agéntico*, capaz de ejecutar herramientas, consultar bases de datos y operar de forma autónoma. Es en la culminación de este flujo técnico donde el software deja de ser un mero predictor estadístico para convertirse en una herramienta colaborativa avanzada.

Y es aquí, en el clímax del cuarto movimiento, donde concluye nuestra sinfonía de los datos. El viaje nos ha llevado desde la humilde geometría de la recta de regresión, apenas una nota pura y aislada, hasta el tejido coral de los modernos modelos de lenguaje, donde miles de millones de parámetros cantan a la vez de forma simultánea. Un resumen del camino recorrido, de los problemas, soluciones y complejidad de cada movimiento se resumen en la tabla 2.

Al apagarse el último compás, conmovidos por la imponente arquitectura sonora desplegada por Beethoven, permanecemos inmóviles en la penumbra del auditorio. La música ha cesado, pero el aire sigue vibrando. Nos disponemos ahora a habitar sus resonancias.

Tabla 2. Resumen de los movimientos y su contenido

Movimiento	I	II	III	IV
Ingeniería	Diseño	Química	Mecánica	Eléctrica / Texto
Problema	Lineal	No lineal separable	No lineal acoplado	Predicción secuencial
Dimensiones	1	3	753	192 /1536
Const. atributos	No	Manual	Integrada	Integrada
Modelo	Recta regresión	Plano de regresión	Red Neuronal	Red neuronal / *Transformer*
Parámetros	2	4	31566	451 / $1{,}5 \cdot 10^{9}$
Datos	100	100	1000	11712 / $7 \cdot 10^{12}$
Melodía	*Cumpleaños Feliz*	*Para Elisa*	*El Danubio Azul*	*Novena Sinfonía*
Notas	~25	$\sim 10^{3}$	$\sim 10^{4}$	$> 10^{5}$

Resonancias

Alcanzada la cumbre del cuarto movimiento, donde los modelos de atención han desplegado su arquitectura secuencial ante nuestros ojos, lo más digno sería, quizás, guardar silencio. Tras el acorde final de la Novena Sinfonía de Beethoven, cualquier sobreexplicación corre el riesgo de profanar la belleza de lo que acaba de ser evocado. Las resonancias que un discurso de esta naturaleza despierta en el pecho de cada oyente son estrictamente personales y no pretendo contaminarlas en exceso con las mías propias. Me limitaré, por tanto, a compartir apenas tres destellos, tres breves vectores de pensamiento que conectan la matemática del silicio con la condición humana en esta inauguración del curso académico.

La primera resonancia nace del asombro y, por qué no admitirlo, del temblor intelectual que produce el comportamiento de los modernos modelos de lenguaje. A lo largo de esta lección hemos ido escalando lo que Richard Dawkins llamó, en un contexto biológico, «la subida al monte improbable»[47]. Hemos partido de un átomo de cálculo elemental, la recta de regresión lineal, y acumulando nodos, capas, dimensiones y parámetros, hemos coronado una cima donde las máquinas imitan, y a veces superan, facultades que creíamos exclusivas de nuestro intelecto. Sin embargo, el paralelismo con Dawkins se quiebra en un punto fundamental: esta ascensión de la materia inanimada hacia la hipercomplejidad simbólica no ha sido el fruto del azar ciego de la selección natural. Aquí sí hay teleología; hay un propósito, un diseño consciente, un director de orquesta invisible que no es otro que el genio y el esfuerzo colectivo y acumulado de la humanidad.

Es cierto que este salto cualitativo nos sobrecoge con una suerte de sublime matemático en sentido kantiano[48]: un auténtico vértigo de la razón ante la inmensidad de un espacio latente de miles de dimensiones abstractas, poblada por millones de millones de datos, que la imaginación es incapaz de abarcar de un solo golpe. Sin embargo, conviene levantar aquí un muro de contención intelectual. Frente al asombro ante la máquina, no podemos dejarnos arrastrar por lo que Rudolf Otto[49] denominó en su célebre análisis

de lo numinoso el «sentimiento de criatura», ese hundimiento del yo, esa postración mística ante lo totalmente otro. Negamos categóricamente esa capitulación. La IA no es una deidad ante la que debamos sentir nuestra nadería; no somos criaturas desvalidas ante un nuevo dios de silicio, sino los artífices de su estructura. Lo que nos estremece en el modelo no es el misterio de lo sagrado, sino el espejo de nuestra propia complejidad organizada: la emergencia.

Es imposible no evocar aquí a uno de los más grandes defensores de esta idea, el sociólogo y filósofo Edgar Morin, muy recientemente fallecido, que llama «emergencia a las cualidades o propiedades de un sistema que presentan un carácter de novedad en relación con las cualidades o propiedades de los componentes considerados de forma aislada»[50]. Dicho con otras palabras: «el todo es más que la suma de las partes». A partir de cierta escala crítica (de un determinado umbral de complejidad), los sistemas en general y la IA en particular empiezan a manifestar «habilidades emergentes». Las interacciones masivas de elementos matemáticos simples provocan un salto cualitativo, una propiedad emergente global.

Casi al mismo tiempo que Morin, en este caso desde el ámbito de la física, el premio Nobel Philip Anderson acuñaba un lema que se convertiría en el manifiesto de esta revolución: «More is different» (Más es diferente)[51]. Anderson nos advertía de que la genialidad de la naturaleza

no consiste en acumular más de lo mismo, sino en que, al aumentar la escala, la cantidad se transforma en una cualidad completamente nueva. Las leyes de un nivel superior no son la simple prolongación de las del nivel inferior.

Evocar la emergencia hoy, ante estas máquinas, nos obliga a admitir esa misma paradoja andersoniana: el comportamiento inteligente del modelo no reside en *ninguno* de sus billones de parámetros individuales, sino en las interacciones que ocurren cuando esos elementos se coordinan. La emergencia es el misterio por el cual el orden produce novedad; es el proceso en el que las restricciones de la estructura dan a luz a una libertad cualitativa insospechada. Una cancioncilla popular apenas nos altera el pulso, pero imbuirse en la masa coral de la Novena de Beethoven puede dejarnos profundamente conmovidos, víctimas de una suerte de síndrome de Stendhal acústico ante la contemplación de una belleza tan masiva e hipercompleja. Porque la música no está en las notas negras sobre el pentagrama, sino en la arquitectura relacional que emerge entre ellas al sonar juntas. Las propiedades emergentes de la IA no son magia; son la firma de la complejidad organizada.

La segunda resonancia tiende un puente musical y académico. El clímax del cuarto movimiento de la sinfonía se sostiene sobre el poema de Schiller: la oda a la *Freude*, a la Alegría. No se trata de una alegría solipsista, hedonista o autocomplaciente, sino de una fuerza cósmica y

transformadora que se enmarca en la fraternidad universal: «Alle Menschen werden Brüder» (Todos los hombres volverán a ser hermanos). Qué hermosa analogía para engarzarla hoy con el *Gaudeamus igitur* que sella la apertura de nuestras aulas. El «Alegrémonos, pues» universitario comparte la misma raíz ilustrada y humanista que el canto de Beethoven: la convicción de que el conocimiento no es un bien privado para el aislamiento individual, sino un espacio de comunión fraterna. Inaugurar un curso, un verbo bellísimo que nos remite al arte antiguo de escrutar los astros y otros signos naturales para desear el bien de la comunidad, debe ser, en esencia, un acto de esperanza compartida. Deseamos buenos augurios para nuestra comunidad porque confiamos en el lazo que nos une.

La tercera y última resonancia es un vector de compromiso político y territorial. La Oda a la Alegría que acabamos de evocar no es solo el testamento musical de Beethoven; es también, por derecho propio, el himno de Europa. En los tiempos convulsos y fragmentados en los que nos desenvolvemos, donde los mapas geopolíticos crujen y las fronteras amenazan con cerrarse sobre el pensamiento, la Universidad no puede permitirse ser un islote ensimismado. El conocimiento, por su propia naturaleza, no entiende de aduanas. Reivindicar hoy la Novena es reivindicar la vocación internacional e irrenunciablemente europea de nuestra Universidad. Una vocación que no se queda en la mera abstracción

burocrática, sino que se encarna con orgullo en proyectos tan ambiciosos y queridos por esta institución como el consorcio europeo *Ulysseus*. Al igual que aquella mítica tripulación camino de Ítaca (¡cómo no evocar aquí a Kavafis[74]!), nuestras mentes universitarias deben navegar juntas el océano de la complejidad contemporánea, sabiendo que la respuesta a los grandes desafíos de la ingeniería y de la vida solo se encontrará si somos capaces de sintonizar nuestras orquestas en una red de cooperación transnacional.

La recta de regresión nos enseñó a buscar la dirección en medio del ruido de unos pocos puntos. Los grandes modelos de lenguaje nos demuestran hoy que la sintaxis del mundo es hipercompleja. Pero corresponde a los seres humanos, y de manera excelente a esta comunidad académica, insuflar el sentido, la ética y la música que hagan de esa estructura de datos una verdadera sinfonía humana.

He dicho. Muchas gracias.

---

74. Constantino Kavafis, en su poema «Ítaca» (1911), inmortalizó la idea de que la riqueza del viaje reside en el camino y el conocimiento acumulado, y no en la meta. Una máxima que se aplica con precisión milimétrica tanto a la formación universitaria como a la investigación científica en el ámbito de la complejidad.

Acotación

Quiero aprovechar esta acotación al margen de la partitura para despejar el elefante que hoy habita de forma inevitable en cualquier foro donde se hable de tecnología. Llegados a este punto, el lector, con legítima malicia, se estará haciendo la gran pregunta: «¿Quién ha escrito realmente estas páginas?»

En un ejercicio de honestidad intelectual, la respuesta correcta es que este texto es un producto nacido de un diálogo extenso y a ratos obstinado entre la mente de quien firma esta lección y un modelo de lenguaje contemporáneo. Podríamos decir, con la ironía que nos brinda Giordano Bruno, que si este discurso *non e vero* es responsabilidad exclusiva del autor, pero si está *ben trovato* es mérito compartido.

Sin embargo, conviene no llamarse a engaño sobre el reparto de funciones en esta partitura: la máquina ha

aportado principalmente el acceso casi instantáneo a las ingentes fuentes de información que ha tenido a su alcance durante su entrenamiento y que ha sido capaz de presentar con una sintaxis fluida; pero la arquitectura conceptual, el diseño pedagógico de cada una de las etapas, y la voluntad filosófica y política que late al fondo de cada línea pertenecen, en exclusiva, a la dirección de orquesta humana.

Para dejar constancia metodológica del diseño de esta obra, conviene detallar que los siguientes elementos emanan, por entero, de dicha batuta:

a) El título y la analogía musical que impregnan todo el texto;
b) El hilo conductor de la lección (el viaje técnico desde la recta de regresión, pasando por la ingeniería de características y el aproximador universal, hasta el predictor secuencial de datos y textos);
c) El alineamiento homólogo de estas cuatro etapas con los grados de ingeniería de la Escuela Politécnica Superior;
d) La elección de la Novena Sinfonía de Beethoven como metáfora estructural;
e) El diseño y depuración de los varios miles de líneas de código y del medio centenar de gráficos que ilustran el texto; y
f) La resolución final de las Resonancias, que trenza la emergencia de Morin con el Gaudeamus universitario y

una reivindicación final de la fe europeísta del autor y la vocación internacional de la universidad.

El papel del modelo de lenguaje (en este caso mayoritariamente Gemini de Google) se ha circunscrito estrictamente a las tareas de soporte y ejecución técnica, actuando como:

a) Asistente de debate y *sparring* conceptual para la búsqueda, profundización y depuración de ideas de ingeniería y sus metáforas musicales;
b) Proveedor de información bibliográfica, verificación de citas e identificación de referencias históricas;
c) Consultor sobre sintaxis de programación y, en contadas ocasiones, propuesta de pequeños fragmentos de código.
d) Redactor de algunos primeros borradores de prosa y puntual asistente editorial en la corrección de estilo.

La conclusión personal de esta experiencia es que la IA ha potenciado significativamente la productividad del autor, pero ha presentado algunos sesgos limitantes entre los que destaco:

a) La tendencia a ser condescendiente en sus respuestas y dar la razón a su interlocutor (aunque objetivamente estuviera equivocado).
b) La alucinación (el dar respuestas falsas con apariencia formal de ser ciertas). Lo que ha requerido, por tanto,

una continua, atenta y estrecha supervisión. Además, el carácter verosímil de sus respuestas ha resultado en ocasiones peligrosamente engañoso.

c) Las limitaciones creativas. Se ha mostrado capaz de desarrollar una idea, incluso de desplegarla brillantemente, pero la iniciativa en todos los casos ha tenido que ser humana.

Referencias

[1] F. Bacon, M. Á. Granada, and J. Martin, *La gran restauración: («novum organum»)*. Tecnos, 2011.

[2] R. Carnap, *La construcción lógica del mundo*. México D.F.: Universidad Nacional Autónoma de México, Instituto de Investigaciones Filosóficas, 1988.

[3] K. R. Popper and V. S. de Zavala, *La lógica de la investigación científica* in Estructura y función. Tecnos, 2011.

[4] T. S. Kuhn and C. Solís, *La Estructura de las revoluciones cientificas* in Breviarios Series. Fondo de Cultura Economica, 2013.

[5] N. R. Hanson, *Patterns of Discovery: An Inquiry into the Conceptual Foundations of Science*. Cambridge University Press, 1958.

[6] C. Anderson, «The End of Theory: The Data Deluge Makes the Scientific Method Obsolete», *Wired Magazine*, vol. 16, no. 7, pp. 108-111, Jun.

2008, [Online]. Available: https://www.wired.com/2008/06/pb-theory/

[7] B. Latour, *Pandora's hope: Essays on the Reality of Science Studies*. Harvard University press, 1999.

[8] R. Kitchin, *The Data Revolution: Big Data, Open Data, Data Infrastructures and Their Consequences*. SAGE Publications, 2014.

[9] A. Pacey, *The Maze of Ingenuity, second edition: Ideas and Idealism in the Development of Technology*. in Mit Press. MIT Press, 1992.

[10] T. Shinn, *Savoir scientifique et pouvoir social L'École polytechnique 1794-1914*. Presses de la Fondation nationale des sciences politiques, 1980.

[11] J. M. Cano Pavón, *La Escuela Industrial de Sevilla (1850-1866)*. Sevilla: Universidad de Sevilla, Secretariado de Publicaciones, 1996.

[12] J. J. López Vázquez, *De la escuela industrial de Sevilla a la Escuela Politécnica Superior: más de 160 años en la formación de peritos e ingenieros técnicos industriales (1850-2018)*, no. 12. in Colección Historia de la Universidad de Sevilla. Sevilla: Editorial Universidad de Sevilla, 2020.

[13] E. J. Dijksterhuis, *Archimedes*. Princeton, New Jersey: Princeton University Press, 2014.

[14] J. McCarthy, N. Rochester, and C. Shannon, «Dartmouth workshop», *Summer Research Project on*, 1955.

[15] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep Learning*, vol. 1, no. 2. MIT press Cambridge, 2016.

[16] D. Acemoglu and P. Restrepo, «Artificial Intelligence, Automation and Work», in *The Economics of Artificial Intelligence: An agenda*, University of Chicago Press, 2018, pp. 197-236.

[17] European Parliament and C. of the European Union, «Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act)», 2024. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32024R1689

[18] D. J. Chalmers, «Could a large language model be conscious?», *arXiv preprint arXiv:2303.07103*, 2023.

[19] Papa León XIV, «Carta encíclica Magnifica Humanitas: sobre la custodia de la persona humana en el tiempo de la inteligencia artificial», May 2026, *Libreria Editrice Vaticana*. [Online]. Available: https://www.vatican.va/content/leo-xiv/es/encyclicals/documents/20260515-magnifica-humanitas.html

[20] A. M. Turing, «Computing Machinery and Intelligence», *Mind*, vol. LIX, no. 236, pp. 433-460, 1950, doi: 10.1093/mind/LIX.236.433.

[21] J. R. Searle, «Minds, brains, and programs», *Behavioral and brain sciences*, vol. 3, no. 3, pp. 417-424, 1980.

[22] C. R. Jones and B. K. Bergen, «Large language models pass a standard three-party Turing test», *Proceedings of the National Academy of Sciences*, vol. 123, no. 21, p. e2524472123, 2026.

[23] W. Kinderman, *Beethoven*. Oxford University Press, 2009.

[24] G. Strang, *Linear Algebra and Its Applications*. Thomson, Brooks/Cole, 2006.

[25] R.-Y. Sun, «Optimization for deep learning: An overview», *Journal of the Operations Research Society of China*, vol. 8, no. 2, pp. 249-294, 2020.

[26] W. B. Nelson, *Applied Life Data Analysis*. John Wiley & Sons, 2005.

[27] G. Cybenko, «Approximation by superpositions of a sigmoidal function», *Mathematics of Control, Signals and Systems*, vol. 2, no. 4, pp. 303-314, 1989.

[28] K. Hornik, «Approximation capabilities of multilayer feedforward networks», *Neural Networks*, vol. 4, no. 2, pp. 251-257, 1991.

[29] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, «Multilayer feedforward networks with a nonpolynomial activation function can approximate any function», *Neural Networks*, vol. 6, no. 6, pp. 861-867, 1993.

[30] W. S. McCulloch and W. Pitts, «A logical calculus of the ideas immanent in nervous activity», *Bull. Math. Biophys.*, vol. 5, no. 4, pp. 115-133, 1943.

[31] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, «Learning representations by back-propagating errors», *Nature*, vol. 323, no. 6088, pp. 533-536, 1986.

[32] Y. Le Cun, I. Kanter, and S. A. Solla, «Eigenvalues of covariance matrices: Application to neural-network learning», *Phys. Rev. Lett.*, vol. 66, no. 18, p. 2396, 1991.

[33] X. Glorot, A. Bordes, and Y. Bengio, «Deep sparse rectifier neural networks», in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 315-323, 2011.

[34] V. Kunc and J. Kléma, «Three decades of activations: A comprehensive survey of 400 activation functions for neural networks», *arXiv preprint arXiv:2402.09092*, 2024.

[35] A. Vaswani *et al.*, «Attention is all you need», *Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.

[36] R. Wagner, *Beethoven*. Caligrama, 2021.

[37] N. Elhage *et al.*, «A mathematical framework for transformer circuits», *Transformer Circuits Thread*, vol. 1, no. 1, p. 12, 2021.

[38] A. Templeton *et al.*, «Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet», *arXiv preprint arXiv:2605.29358*, 2026.

[39] L. Gao *et al.*, «Scaling and evaluating sparse autoencoders», in *International Conference on Learning Representations*, 2025, pp. 26721-26754.

[40] T. Mikolov, W. Yih, and G. Zweig, «Linguistic regularities in continuous space word representations», in *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: Human language technologies*, 2013, pp. 746-751.

[41] L. van Beethoven, *Symphony N.º 9 in D minor, Op. 125*. International Music Score Library Project (IMSLP), 2024. [Online]. Available: https://imslp.org/wiki/File:PMLP1607-Symphony_No_9_Op_125_-_Beethoven.pdf

[42] Biblioteca Nacional de España, «Memoria 2025», 2025. Accessed: Jul. 17, 2026. [Online]. Available: https://www.bne.es/sites/default/files/repositorio-archivos/memoria_BNE_2025.pdf

[43] Biblioteca de la Universidad de Sevilla, «Memoria de Actividad 2024», 2025. Accessed: Jul. 17, 2026. [Online]. Available: https://idus.us.es/bitstreams/be5ca1ed-d843-4451-9f2e-b41ab46a8e3d/download

[44] J. Kaplan *et al.*, «Scaling laws for neural language models», *arXiv preprint arXiv:2001.08361*, 2020.

[45] J. Hoffmann *et al.*, «Training compute-optimal large language models», *arXiv preprint arXiv:2203.15556*, vol. 10, 2022.

[46] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell, «On the dangers of stochastic parrots: Can language models be too big?», in *Proceedings of*

the 2021 ACM conference on fairness, accountability, and transparency, 2021, pp. 610-623.
[47] R. Dawkins, *Climbing mount improbable*. WW Norton & Company, 1996.
[48] I. Kant, *Crítica del juicio*. Madrid: Austral, 2013.
[49] R. Otto, *Lo Santo. Lo racional y lo irracional en la idea de Dios*. Madrid: Alianza Editorial, 2001.
[50] E. Morin, *El método: La naturaleza de la Naturaleza. I*. in Colección Teorema: serie Mayor. Cátedra, 1993.
[51] P. W. Anderson, «More is different: broken symmetry and the nature of the hierarchical structure of science». *Science (1979)*, vol. 177, no. 4047, pp. 393-396, 1972.
[52] T. Gebru *et al.*, «Datasheets for datasets», *Commun. ACM*, vol. 64, no. 12, pp. 86-92, 2021.
[53] J. I. Guerrero, A. García, E. Personal, J. Luque, and C. León, «Heterogeneous data source integration for smart grid ecosystems based on metadata mining», *Expert Syst. Appl.*, vol. 79, pp. 254-268, 2017.
[54] R. Sánchez-Durán, J. Luque, and J. Barbancho, «Long-Term Demand Forecasting in a Scenario of Energy Transition», *Energies (Basel).*, vol. 12, no. 16, p. 3095, Aug. 2019, doi: 10.3390/en12163095.
[55] J. Luque, D. Anguita, F. Pérez, and R. Denda, «Spectral Analysis of Electricity Demand Using Hilbert-Huang Transform», *Sensors*, vol. 20, no. 10, p. 2912, May 2020, doi: 10.3390/s20102912.

[56] J. Luque, E. Personal, A. Garcia-Delgado, and C. Leon, «Monthly Electricity Demand Patterns and Their Relationship with the Economic Sector and Geographic Location», *IEEE Access*, vol. 9, pp. 86254-86267, 2021, doi: 10.1109/ACCESS.2021.3089443.

[57] J. Luque, A. Carrasco, E. Personal, F. Pérez, and C. León, «Customer Identification for Electricity Retailers Based on Monthly Demand Profiles by Activity Sectors and Locations», *IEEE Transactions on Power Systems*, 2023.

[58] J. Luque, E. Personal, F. Perez, Mc. Romero-Ternero, and C. Leon, «Low-dimensional representation of monthly electricity demand profiles», *Eng. Appl. Artif. Intell.*, vol. 119, p. 105728, 2023.

[59] J. Luque, B. Tepe, D. Larios, C. León, and H. Hesse, «Machine Learning Estimation of Battery Efficiency and Related Key Performance Indicators in Smart Energy Systems», *Energies 2023, Vol. 16, Page 5548*, vol. 16, no. 14, p. 5548, Jul. 2023, doi: 10.3390/EN16145548.

[60] J. Luque, Á. Gómez, A. Carrasco, C. León, and A. Fernández, «Advanced Feature Engineering for Non-Intrusive Load Monitoring», *IEEE Access*, vol. 14, pp. 30037-30052, 2026, doi: 10.1109/ACCESS.2026.3667181.

[61] F. Takens, «Detecting strange attractors in turbulence», in *Dynamical Systems and Turbulence, Warwick 1980: proceedings of a symposium held at the University of Warwick 1979/80*, 2006, pp. 366-381.

[62] E. Abe *et al.*, «Zero-shot Prediction for Household Electricity and Its Integration into Scenario-based MPC for Economical Battery Management», in *IFAC World Congress*, Busan, Korea, Aug. 2026.

[63] A. F. Ansari *et al.*, «Chronos: Learning the language of time series», *arXiv preprint arXiv:2403.07815*, 2024.

[64] N. Hollmann *et al.*, «Accurate predictions on small data with a tabular foundation model», *Nature*, vol. 637, no. 8045, pp. 319-326, 2025.

[65] A. Yang *et al.*, «Qwen2 Technical Report», 2024. [Online]. Available: https://arxiv.org/abs/2407.10671

[66] M. Kadem and R. Zheng, «Interpreting Transformers Through Attention Head Intervention», *arXiv preprint arXiv:2601.04398*, 2026.

[67] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, «What does BERT look at? an analysis of BERT's attention», in *Proceedings of the 2019 ACL workshop BlackboxNLP: analyzing and interpreting neural networks for NLP*, 2019, pp. 276-286.

[68] F. de Quevedo and J. M. Blecua, *Poesía completa*. Turner, 1995.

[69] A. A. Abdullah *et al.*, «Evolution of Meta's Llama models and parameter-efficient fine-tuning of large language models: a survey», *arXiv preprint arXiv:2510.12178*, 2025.

[70] A. Xu *et al.*, «DeepSeek-V4: Towards Highly Efficient Million-Token Context Intelligence», *arXiv preprint arXiv:2606.19348*, 2026.

[71] OpenAI, «Core Concepts: Tokens and Text Generation», 2026. [Online]. Available: https://developers.openai.com/api/docs/concepts#tokens

[72] P. Villalobos, A. Ho, J. Sevilla, T. Besiroglu, L. Heim, and M. Hobbhahn, «Will we run out of data? Limits of LLM scaling based on human-generated data», *arXiv preprint arXiv:2211.04325*, 2022.

JOAQUÍN LUQUE RODRÍGUEZ

Ingeniero industrial (1980), doctor ingeniero industrial (1986) y licenciado en Filosofía (1994) por la Universidad de Sevilla. Máster en «Ciencia, Tecnología y Sociedad» (1997) por la Universidad Nacional de Educación a Distancia (UNED).

Catedrático de Tecnología Electrónica desde 1996, está adscrito al Departamento de Tecnología Electrónica en la Escuela Politécnica Superior de la Universidad de Sevilla. También ha sido profesor o investigador visitante en la Universidad de California-Berkeley (EE. UU.), Universidad de Génova (Italia), Universidad de Kempten (Alemania), Universidad Técnica de Múnich (Alemania), Universidad de la República (Uruguay) y del Instituto de Ciencias de Tokio (Japón).

Acredita más de 40 años de experiencia docente en diversas disciplinas de la ingeniería principalmente relacionadas con la electrónica, las comunicaciones, el control y la inteligencia artificial.

Cuenta con más de un centenar de publicaciones entre libros, capítulos de libros, artículos de revista y contribuciones a actas de congresos. Ha participado en numerosos proyectos de investigación, tanto con financiación pública como privada. Es miembro fundador del grupo de investigación «Tecnología Electrónica e Informática Industrial», que dirigió desde su creación en 1995 hasta 2008.

Tiene también una dilatada experiencia de gestión en la Universidad de Sevilla, donde ha sido subdirector de la Escuela Politécnica desde 1988 a 1992, director del Departamento de Tecnología Electrónica desde 1993 a 2000, director del Secretariado de Tecnologías de la Información y las Comunicaciones desde 2000 a 2004, vicerrector de Infraestructuras desde 2004 a 2008, y rector desde 2008 a 2012. Ha cofundado y sido el primer presidente, desde 2010 a 2012, de Andalucía Tech, una acción estratégica conjunta con la Universidad de Málaga. Desde 2019 a 2026 ha sido director de la Cátedra Indra «Sociedad Digital».

Con anterioridad a su experiencia académica trabajó, desde 1980 a 1984, en la empresa SAINCO desarrollando sistemas de control de redes eléctricas (SCADA). Igualmente, desde 1984 a 1986, perteneció al departamento de informática de la Consejería de Hacienda de la Junta de Andalucía. Ha sido cofundador de la empresa Isotrol, una *start-up*, de la que fue director de Investigación.

Ha recibido diferentes distinciones por su trabajo académico, entre ellas el Premio Maestranza de Caballería al mejor expediente académico (1980), el diploma de «Excelencia Académica» (en diversos años), el Premio a las «Tecnologías de la Información en Andalucía» (2003) y la Medalla de la Universidad de Málaga (2012).

UNIVERSIDAD Ð SEVILLA