COLECCIÓN

MITOS Y REALIDADES DE LA IA

Entre la promesa y la realidad

IA BASADA EN REGLAS E IA QUE APRENDE Y CÓMO PUEDEN COEXISTIR

Número 3



MRCI

anejo de ecursos y ontroles nteligentes

DR. JOSÉ ANTONIO PÉREZ RAMOS

IA BASADA EN REGLAS E IA QUE APRENDE Y CÓMO PUEDEN COEXISTIR

SOBRE EL AUTOR

Doctor en Ciencias de lo Fiscal por el Instituto de Especialización para Ejecutivos. Maestro en Fiscal por la Universidad Autónoma "Benito Juárez" de Oaxaca. Maestro en Estudios Avanzados en Derechos Humanos por la Global Open University. Licenciado en Contaduría Pública por la Universidad Autónoma "Benito Juárez" de Oaxaca y Licenciado en Filosofía por la Universidad Nacionalista de México.

Socio fundador y director general de la Firma Manejo de Recursos y Controles Inteligentes (Firma MRCI). Fiscalista del año 2009 por la revista *Defensa Fiscal*. Doctor honoris causa por la Asamblea General del Claustro Doctoral Iberoamericano (2016). Doctor honoris causa por el Instituto Mexicano de Líderes de Excelencia (2017). Doctor honoris causa por 1Million Startups, Latinomics, Leaderships Forum y la Fundación Humanist World (2018). Doctor honoris causa por el Claustro de Doctores de la AISOL (American International School of Law) (2019).

Autor de diversas obras, entre las que destacan *Ius mutantur o del derecho que cambia* (Tirant Lo Blanch, 2025), *Futurología* (MRCI, 2025) y *Hasta que la muerte nos separe* (MRCI, 2025). Ha participado como coordinador en *Cuestiones tributarias. Problemas y controversias en el México actual* (Tirant Lo Blanch, 2023), *El costo de la justicia* (Apex Iuris, 2019), *Remuneraciones estratégicas inteligentes* (MRCI, 2015) y *Retos de la justicia tributaria* (Tirant Lo Blanch-AMDF, 2025).

Titular de *Pie de Página*, programa para la divulgación de los derechos humanos en México, a través de la plataforma Legal Cloud. Expositor en foros nacionales e internacionales. Colaborador en diversas revistas jurídicas, entre las que destacan *Defensa Fiscal* y *Legal Cloud*.



IA BASADA EN REGLAS E IA QUE APRENDE Y CÓMO PUEDEN COEXISTIR

DR. JOSÉ ANTONIO PÉREZ RAMOS

PRIMERA EDICIÓN, ATEMPORAL

Derechos reservados, propiedad de
Dr. José Antonio Pérez Ramos

Comentarios y opiniones: investigacion@mrci.com.mx

Título original: IA basada en reglas e IA que aprende y cómo pueden coexistir.

Colección mitos y realidades de la inteligencia artificial.

Entre la promesa y la realidad.

Autor: Dr. José Antonio Pérez Ramos.

Queda prohibida la reproducción total y parcial de esta obra denominada; IA BASADA EN REGLAS E IA QUE APRENDE Y CÓMO PUEDEN COEXISTIR, por cualquier medio, sin autorización escrita del autor.

PRINTED IN MEXICO
IMPRESO EN MÉXICO



ÍNDICE

INTRODUCCIÓN ..... - 7 -
EL PROBLEMA DE ELEGIR POR ETIQUETAS ..... - 10 -
TRES PREGUNTAS PARA SEPARAR LOS ENFOQUES ..... - 12 -
CÓMO OPERA UNA COLECCIÓN DE REGLAS ..... - 15 -
EXCEPCIONES Y HECHOS DESCONOCIDOS ..... - 18 -
QUÉ SIGNIFICA APRENDER UNA RELACIÓN ..... - 19 -
REGLAS QUE SE OBTIENEN DE EJEMPLOS ..... - 22 -
LAS COMBINACIONES TIENEN MECANISMOS DISTINTOS ..... - 23 -
COOPERACIÓN MEDIANTE ETAPAS ..... - 25 -
REGLAS QUE AYUDAN A ENTRENAR ..... - 27 -
INTEGRACIÓN NEUROSIMBÓLICA ..... - 29 -
CASO DE RECEPCIÓN DOCUMENTAL ..... - 30 -
UN CÁLCULO QUE REVELA EL COSTO DEL FILTRO ..... - 34 -
CASO DE AUTORIZACIÓN PARA ENVIAR AVISOS ..... - 36 -
CASO DE PREDICCIÓN Y PROGRAMACIÓN ..... - 39 -
QUÉ DEBE DEMOSTRAR LA EVALUACIÓN ..... - 42 -
SEPARAR EL APORTE DE CADA COMPONENTE ..... - 45 -
MANTENIMIENTO Y DESACUERDOS ..... - 46 -
MITOS QUE DIFICULTAN DECIDIR ..... - 49 -
CONCLUSIÓN ..... - 51 -
REFERENCIAS ..... - 52 -

Introducción

Comprender la inteligencia artificial exige reconocer que bajo una misma denominación conviven formas distintas de representar información, obtener conocimiento y producir resultados. Algunos sistemas utilizan condiciones explícitas establecidas por personas; otros ajustan modelos a partir de ejemplos para identificar regularidades y generar estimaciones. Esta diversidad suele simplificarse mediante una oposición entre la inteligencia artificial basada en reglas y aquella que aprende, como si ambas pertenecieran a etapas incompatibles del desarrollo tecnológico. Sin embargo, examinar sus mecanismos permite advertir que pueden desempeñar funciones complementarias dentro de un mismo proceso.

La distinción tiene consecuencias prácticas. Leer un documento, comprobar que contiene los datos requeridos y determinar si existe autorización para realizar una acción son tareas relacionadas, pero cada una plantea exigencias diferentes. Un modelo puede reconocer información en formatos variables y, aun así, extraer un dato equivocado; una regla puede verificar que ese dato cumple una condición, aunque carezca de elementos para comprobar su correspondencia con la realidad. El funcionamiento adecuado del conjunto depende, por tanto, de precisar qué

aporta cada componente y hasta dónde alcanza la comprobación que realiza.

Este artículo parte de una pregunta central: ¿cómo pueden coexistir las reglas explícitas y el aprendizaje automático de manera que sus funciones sean comprensibles, sus resultados puedan evaluarse y sus límites permanezcan visibles? Para responderla, propone distinguir tres dimensiones: el origen de las relaciones que utiliza un sistema, la forma en que se representan y el procedimiento mediante el cual se ejecutan. Esta separación permite comprender, por ejemplo, que una relación obtenida mediante aprendizaje puede expresarse como un árbol de decisiones, mientras una regla escrita por personas puede intervenir en la preparación de datos o en la revisión de una salida.

Desde esta perspectiva, la coexistencia admite diferentes mecanismos. Puede organizarse mediante etapas, cuando un modelo propone un resultado que después se somete a condiciones explícitas; mediante reglas que aportan señales para el entrenamiento; o mediante integraciones que articulan aprendizaje y procedimientos simbólicos. Cada combinación requiere examinar sus supuestos, sus interfaces y la evidencia que sostiene su utilidad. La



incorporación de varios componentes también introduce dependencias y posibles desacuerdos cuyo tratamiento forma parte del diseño.

El propósito es ofrecer criterios para comprender y evaluar estas formas de cooperación. Ello supone identificar responsabilidades, conservar la procedencia de los datos, definir qué ocurre ante información insuficiente y comprobar el comportamiento del sistema cuando aparecen excepciones o contradicciones. Con estas bases, la relación entre reglas y aprendizaje puede analizarse a partir de funciones concretas y resultados verificables, atendiendo tanto a sus posibilidades como a las condiciones necesarias para utilizarlas.

El problema de elegir por etiquetas

Una oficina necesita leer documentos de formatos variados y comprobar que cada solicitud incluya determinados campos; un taller debe anticipar fallas sin programar tareas incompatibles con su capacidad; una persona quiere que un asistente prepare avisos, pero que solo los envíe después de una aprobación específica; en esas situaciones, reconocer patrones y respetar condiciones tienen funciones diferentes, aunque ambas actividades formen parte del mismo servicio.

La pregunta central es cómo distribuir esas funciones sin atribuir a una técnica propiedades que no posee: ¿qué conviene expresar como condición explícita, qué puede estimarse a partir de ejemplos y qué necesita una revisión adicional porque los datos o el criterio siguen siendo inciertos?

La tesis de este artículo es que la oposición entre reglas y aprendizaje resulta insuficiente para diseñar o evaluar un sistema; importa distinguir el origen de sus relaciones, su representación y su forma de ejecución, además de identificar quién puede cambiar cada una y qué evidencia permite comprobar su desempeño.



Una relación puede aprenderse a partir de datos y terminar expresada como un árbol de decisiones; una regla puede intervenir en la preparación de datos para entrenar una red; un modelo puede sugerir un paso que después se comprueba mediante un procedimiento simbólico; estas combinaciones tienen antecedentes verificables y no requieren imaginar que todos los sistemas convergen en una única técnica.

El método es una revisión documental selectiva, con fuentes consultadas hasta el 2 de octubre de 2026, acompañada de análisis conceptual y ejemplos de elaboración propia; se examinan trabajos históricos porque aclaran mecanismos que siguen siendo relevantes, sin convertir sus resultados en clasificaciones actuales de productos.

Los casos numéricos son hipotéticos y no describen experimentos realizados en organizaciones reales; sus cifras sirven para revelar diferencias entre criterios de evaluación, mientras la discusión de investigaciones publicadas identifica los resultados que pertenecen a sus autores y limita su extrapolación al contexto estudiado.



Tres preguntas para separar los enfoques

La primera pregunta es cómo se obtuvo una relación: alguien pudo escribirla deliberadamente, un procedimiento pudo ajustarla con ejemplos o ambas cosas pudieron ocurrir en distintas etapas; una condición de acceso establecida por una organización y una asociación entre características de imágenes pertenecen a esta dimensión de adquisición, aunque puedan coexistir en un mismo programa.

La segunda pregunta se refiere a cómo se representa esa relación: mediante símbolos, condiciones, tablas, probabilidades, vectores, parámetros de una red o combinaciones de estas formas; la tercera pregunta concierne a su uso, porque una representación puede consultarse directamente, participar en una búsqueda, activar una acción o contribuir a una estimación.

Confundir las tres preguntas produce errores frecuentes; si un árbol aprendido se ejecuta mediante condiciones, sigue siendo un producto del aprendizaje, mientras una red con parámetros fijos durante el uso puede producir siempre la misma clasificación ante una entrada idéntica, siempre que

también permanezcan fijas las demás condiciones de ejecución.

En su exposición sobre ramas de la IA, McCarthy (2007) incluye lógica, búsqueda, representación, inferencia y aprendizaje, además de señalar que también pueden aprenderse relaciones expresadas en lenguajes lógicos; su texto permite ubicar la diversidad de enfoques, aunque constituye una exposición conceptual de su autor y no una medición comparativa de su eficacia.

La Tabla 1 organiza estos ejes con ejemplos conceptuales; su propósito consiste en impedir que la forma visible de un resultado se utilice como prueba suficiente de cómo fue obtenido.

Tabla 1. *Origen y uso de relaciones computacionales*

Ejemplo	Cómo se obtiene	Cómo se utiliza
Regla de autorización	Una persona competente establece una condición	Se comprueba antes de ejecutar una acción

Ejemplo	Cómo se obtiene	Cómo se utiliza
Árbol aprendido	Un algoritmo selecciona divisiones a partir de ejemplos	Se recorren condiciones hasta una salida
Clasificador neuronal	Un procedimiento ajusta parámetros con datos	Se calculan puntuaciones sobre una entrada
Programa lógico probabilístico	Se combinan relaciones explícitas y probabilidades	Se calcula una conclusión bajo una semántica definida
Sistema híbrido de documentos	Se separan extracción aprendida y validaciones escritas	Se acepta, detiene o remite una propuesta

Nota. Elaboración propia; ejemplos analíticos, no inventario exhaustivo de técnicas.

Tampoco todo programa que contiene una condición debe denominarse IA; comparar una cantidad con un límite es una operación habitual de software, mientras la clasificación del sistema completo depende del problema, la combinación de mecanismos y la definición de IA adoptada; llamar inteligente a cada instrucción condicional vuelve poco informativa la categoría.



Conviene reservar la expresión sistema basado en reglas para los casos en que una colección de relaciones explícitas participa de manera sustantiva en la producción de conclusiones o acciones; aun así, la frontera con otras formas de programación puede ser gradual y no existe razón para resolver todos los casos mediante una etiqueta comercial.

Una diferencia adicional atraviesa los tres ejes: algunas reglas describen regularidades que se espera observar y otras establecen condiciones que una acción debe respetar; una correlación histórica puede ayudar a estimar lo que ocurrirá, pero no determina por sí misma quién tiene permiso para actuar.

Cómo opera una colección de reglas

Una regla explícita vincula condiciones y consecuencias; para que sea ejecutable, necesita un vocabulario y una interpretación suficientemente precisos, porque expresiones como documento completo, incidente grave o aprobación vigente admiten variaciones que un programa no resuelve solo por recibirlas escritas en lenguaje natural.

Un motor de reglas puede mantener hechos sobre el caso, localizar condiciones satisfechas, elegir entre reglas aplicables y ejecutar la consecuencia correspondiente; Forgy (1982, pp. 18–22) describe ese ciclo en sistemas de producción y presenta Rete como un procedimiento para realizar eficientemente la comparación entre patrones y objetos, aprovechando las relaciones que pueden conservarse entre cambios del estado.

La contribución técnica de Rete se refiere a esa comparación, no a demostrar que los criterios introducidos por las personas sean correctos; separar eficiencia computacional y validez del contenido evita transformar un procedimiento rápido en una autoridad sustantiva que nunca pretendió ser.

Consideremos un ejemplo hipotético de recepción: la regla A indica que una solicitud con identificador y documento adjunto pasa a comprobación; la regla B establece que un archivo ilegible se remite a subsanación; la regla C permite preparar un acuse cuando terminaron las comprobaciones; las tres reglas necesitan hechos distintos y producen estados distintos.



Si existe identificador y hay un archivo, puede activarse A, pero esa activación no acredita que el archivo sea legible ni que su contenido corresponda a la solicitud; el programa debe mantener separados presencia, lectura, correspondencia y aceptación, pues reducirlos a un único valor llamado válido oculta qué se comprobó realmente.

El encadenamiento hacia adelante parte de hechos disponibles y obtiene consecuencias que pueden servir de entrada a otras reglas; una consulta orientada hacia atrás parte de una conclusión buscada y examina qué condiciones permitirían sostenerla; ambas estrategias pueden ser útiles, aunque el orden de búsqueda y las condiciones de terminación deben definirse para el sistema concreto.

Una cadena de reglas tampoco equivale automáticamente a una demostración lógica; si las consecuencias modifican archivos, envían avisos o eliminan estados, intervienen efectos de ejecución cuyo orden puede importar; por ello, la expresión motor de inferencia exige examinar la semántica efectiva y no basta con observar una interfaz que enumera razones.



Excepciones y hechos desconocidos

La falta de un dato plantea una decisión de diseño: tratarlo como falso, como desconocido o como condición que bloquea el procedimiento; esos tratamientos no son equivalentes, porque la ausencia de constancia de aprobación no prueba necesariamente que una persona haya rechazado la operación, aunque sí puede impedir que se ejecute.

En el ejemplo de recepción, un campo ilegible debería conservarse como desconocido cuando la lectura no permite determinarlo; convertirlo en un valor vacío y después aplicar una regla de rechazo confunde un problema de observación con una propiedad del documento, mientras completarlo con una suposición confunde incertidumbre con evidencia.

Una excepción necesita ámbito: puede depender de la fecha, del tipo de solicitud o de una condición adicional; si dos reglas se contradicen, el sistema requiere una prioridad justificada o un estado de conflicto, porque elegir silenciosamente la primera condición satisfecha puede producir una resolución operativa sin resolver la incompatibilidad de fondo.



También hay inferencias revisables; McCarthy (2007) explica el razonamiento por defecto como una forma de obtener conclusiones que pueden retirarse ante nueva información; aplicado al análisis presente, esto obliga a distinguir una conclusión provisional de una consecuencia irrevocable y a conservar qué dato nuevo cambió el resultado.

La principal ventaja de una regla explícita es que puede localizarse y examinarse directamente; su límite aparece cuando el vocabulario es ambiguo, faltan excepciones, crece la interacción entre condiciones o la entrada no representa bien el hecho relevante; la inspección ayuda a diagnosticar esas fallas, pero no las elimina.

Qué significa aprender una relación

Mitchell (2006, p. 1) vincula el aprendizaje con una tarea, una medida de desempeño y una experiencia que permite mejorarla; esa formulación exige decir qué se espera aprender y cómo se observa la mejora, en lugar de considerar aprendizaje cualquier cambio de respuesta, archivo almacenado o conversación más extensa.

En un clasificador supervisado, los ejemplos contienen entradas y etiquetas; el procedimiento busca una función que produzca salidas útiles en relación con esas etiquetas, dentro de una familia de modelos y bajo un objetivo de ajuste; la función aprendida depende de esas decisiones iniciales y de las regularidades efectivamente presentes en los datos.

El equipo humano no desaparece: decide qué ejemplos se recolectan, cómo se etiquetan, qué errores importan, qué variables se ofrecen y cómo se evalúa; incluso cuando parte de esas decisiones se automatiza, alguien debe establecer las condiciones bajo las cuales el resultado será aceptable.

Un modelo que clasifica fotografías de piezas puede beneficiarse de miles de variaciones que sería difícil enumerar manualmente; sin embargo, si todas las piezas defectuosas fueron fotografiadas con un fondo distinto, una buena puntuación sobre datos semejantes podría reflejar atención al fondo y no a la característica que interesa inspeccionar.

Este ejemplo no demuestra que el modelo haya seguido necesariamente ese atajo; identifica una hipótesis comprobable mediante fotografías controladas, cambios de fondo y evaluación con datos separados por fecha o procedencia, porque la explicación de un error requiere observar el comportamiento real y no atribuir un mecanismo por intuición.

Aprender tampoco implica actualizarse permanentemente durante el uso; un modelo puede entrenarse una vez y permanecer congelado, mientras una aplicación agrega documentos a una biblioteca sin modificar sus parámetros; recuperar material nuevo cambia la información disponible para responder, pero no acredita un nuevo entrenamiento del modelo.

La separación entre entrenamiento e inferencia resulta crucial para asignar responsabilidades: una equivocación puede provenir de los ejemplos, del objetivo de ajuste, del procesamiento de una entrada o de una regla posterior; llamar a todo el proceso aprendizaje impide localizar la intervención que debe corregirse.

Reglas que se obtienen de ejemplos

Quinlan (1986) muestra cómo inducir árboles de decisión a partir de ejemplos descritos mediante atributos; el árbol es una estructura explícita que formula pruebas sucesivas para clasificar nuevos casos, por lo que su existencia basta para descartar una frontera tajante entre aprender y terminar utilizando condiciones.

El mismo trabajo considera dificultades como ruido e información incompleta; aquí interesa especialmente su observación de que, si dos objetos tienen los mismos atributos disponibles y pertenecen a clases diferentes, esos atributos no bastan para distinguirlos correctamente en todos los casos (Quinlan, 1986, pp. 85–86).

Construyamos un ejemplo: en veinte órdenes de servicio, diez contienen el atributo ruido anormal y ocho de esas diez terminan en sustitución de una pieza; entre las otras diez, dos terminan en sustitución; una regla aprendida que prediga sustitución cuando se registra ruido acertaría dieciséis de las veinte etiquetas, es decir, el 80 % de esa muestra hipotética.

El cálculo describe concordancia con los registros: ocho aciertos entre órdenes con ruido y ocho entre órdenes sin ruido; no establece que el ruido cause la necesidad de sustitución, que la práctica anterior haya sido adecuada ni que el porcentaje se mantenga en nuevas instalaciones.

Además, el criterio aprendido podría reproducir la costumbre de quienes atendían las órdenes; si se cambiaban piezas por precaución o faltaba equipo de diagnóstico, aprender la etiqueta sustitución reproduce aquella decisión, mientras aprender falla física requeriría otra definición y evidencia adicional.

La distinción es decisiva para las reglas extraídas de un modelo: hacer legible una asociación facilita examinarla, pero no la convierte en una norma ni garantiza que describa un mecanismo causal; su origen empírico y sus límites deben acompañarla cuando se reutiliza.

Las combinaciones tienen mecanismos distintos

La palabra híbrido informa poco si no se especifica qué se combina; una aplicación que verifica una salida neuronal, un procedimiento que usa heurísticas para crear etiquetas

y un sistema que integra probabilidades con reglas lógicas pueden compartir componentes, aunque resuelven problemas diferentes y necesitan pruebas distintas.

La Tabla 2 distingue tres formas de cooperación y el control que no debería perderse en cada una; no propone una escala de superioridad, porque una integración estrecha puede ser innecesaria cuando basta una separación sencilla de responsabilidades.

Tabla 2. *Tres formas de combinar conocimiento explícito y aprendizaje*

Forma	Función de la combinación	Comprobación indispensable
Cooperación por etapas	Un modelo propone información y otro componente valida condiciones	Verificar la procedencia de los datos y la cobertura del control

Forma	Función de la combinación	Comprobación indispensable
Reglas para producir etiquetas	Heurísticas aportan señales de supervisión para entrenar	Evaluar errores y dependencias entre fuentes de etiquetas
Integración neurosimbólica	Aprendizaje y operaciones simbólicas intervienen en un mismo problema	Examinar semántica, representación y resultado del conjunto

Nota. Elaboración propia a partir de las distinciones analizadas; las tres formas pueden coexistir.

Cooperación mediante etapas

La separación por etapas permite asignar a un modelo una tarea perceptiva o interpretativa y a un procedimiento explícito una comprobación posterior; por ejemplo, un extractor propone la fecha de un documento y una función



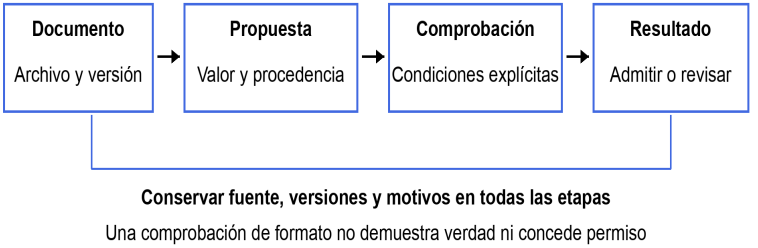
comprueba que corresponda a un día existente, aunque esa validación no establece que la fecha proceda del apartado correcto.

El beneficio potencial consiste en poder cambiar el extractor sin reescribir todas las restricciones, siempre que conserve el contrato de entrada y salida; ese contrato debería indicar tipo de dato, procedencia, estado de desconocimiento y versión, porque dos componentes pueden intercambiar un texto técnicamente válido y atribuirle significados incompatibles.

Una fecha representada como 03/04/2026 ilustra el problema: el formato es legible, pero puede significar tres de abril o cuatro de marzo según la convención; la interfaz necesita una representación inequívoca y evidencia del texto original, sin dejar que una normalización silenciosa decida el contenido que se supone verificar.

La Figura 1 resume esa cooperación y conserva una vía de trazabilidad común; el diagrama distingue la propuesta del modelo, las condiciones comprobadas y el alcance del resultado, sin atribuir autorización a la mera aprobación de un formato.

Figura 1. *Propuesta y comprobación con procedencia conservada*



Nota. Elaboración propia; esquema conceptual del caso documental, sin resultados experimentales.

Reglas que ayudan a entrenar

En Snorkel, Ratner et al. (2017, sección 2) describen funciones de etiquetado que pueden aportar una etiqueta o abstenerse, además de un procedimiento para combinar señales imperfectas y utilizarlas en el entrenamiento de un modelo; las reglas intervienen así como fuentes de supervisión y no solamente como filtros posteriores.

El planteamiento reconoce conflictos y correlaciones entre señales; dos funciones que contienen casi la misma



heurística no equivalen a dos observadores independientes, mientras el artículo conserva conjuntos etiquetados para desarrollo y evaluación aunque reduzca la necesidad de etiquetar manualmente todos los datos de entrenamiento.

En un ejemplo hipotético de clasificación de solicitudes, una función puede reconocer una frase característica, otra consultar el nombre de un formulario y una tercera abstenerse cuando la imagen sea ilegible; sus coincidencias ayudan a formar señales de entrenamiento, pero el resultado necesita contrastarse con casos revisados por un criterio independiente.

Una restricción crítica no debería confiar su cumplimiento únicamente a que apareció como etiqueta durante el entrenamiento; si una condición debe impedir una operación, aprenderla de ejemplos constituye una influencia sobre el comportamiento y no demuestra que la ejecución quede bloqueada en todos los caminos posibles.



Integración neurosimbólica

Manhaeve et al. (2018, secciones 1–5) presentan DeepProbLog como una extensión de programación lógica probabilística que incorpora predicados neuronales y permite entrenar parámetros a partir del resultado del programa; su ejemplo de suma combina reconocimiento de dígitos con una relación aritmética explícita, sin obligar a aprender de nuevo qué significa sumar cada posible pareja.

La probabilidad puede formar parte de un programa simbólico con semántica definida; por tanto, simbólico no significa necesariamente certeza, ni neuronal significa necesariamente ausencia de estructura, aunque la integración mantenga límites derivados del reconocimiento de entradas, la formulación del programa y las condiciones de entrenamiento.

AlphaGeometry ofrece otra distribución: el modelo neuronal propone construcciones auxiliares y un motor simbólico desarrolla deducciones geométricas; Trinh et al. (2024) reportan que su sistema resolvió veinticinco de treinta problemas del conjunto evaluado, con

entrenamiento apoyado en datos sintéticos y dentro de su representación formal de geometría euclidiana.

Ese resultado histórico muestra una cooperación concreta, pero no demuestra que cualquier modelo de lenguaje pueda verificar cualquier razonamiento; la elección del dominio, el lenguaje formal y las operaciones disponibles delimita lo que se puede representar y comprobar, mientras la traducción inicial de un problema también requiere atención.

La lección práctica consiste en localizar qué parte propone y qué parte comprueba; un verificador puede rechazar un candidato incorrecto sin saber cómo producir uno correcto, mientras un generador puede explorar posibilidades sin tener autoridad para declararlas demostradas; confundir esas funciones elimina precisamente la ventaja que la combinación busca obtener.

Caso de recepción documental

Supongamos una oficina ficticia que recibe archivos para organizar un registro interno; desea localizar identificador, fecha y nombre del trámite, verificar presencia y

consistencia, y remitir a una persona los casos inciertos; ninguna aceptación automática del ejemplo tiene efectos jurídicos ni certifica la autenticidad de los documentos.

La primera decisión consiste en separar recibir de aceptar: recibir significa almacenar el archivo y asignarle un identificador; extraer significa proponer valores; comprobar significa ejecutar condiciones sobre ellos; aceptar para el registro significa que el caso cumple el criterio operativo definido, mientras reconocer la verdad de lo declarado sería otra tarea.

Para cada valor propuesto se conservarían el documento de origen, su versión, la página, el fragmento y el método de extracción; si un nombre proviene de la portada y una fecha de un anexo, el sistema debería poder mostrar esa diferencia en vez de entregar únicamente una ficha limpia que esconda cómo se construyó.

Un control de formato detecta un día imposible y otro puede comprobar que el identificador tenga la estructura exigida; sin embargo, una fecha perfectamente válida puede pertenecer a otro asunto, mientras un identificador con longitud correcta puede haber sido copiado de una



muestra incluida en el archivo; la forma y la procedencia necesitan controles diferentes.

También conviene distinguir legibilidad y completitud: una fotografía borrosa no equivale a ausencia del campo, y un campo presente pero contradictorio no equivale a un valor confirmado; si la interfaz solo admite verdadero o falso, fuerza decisiones prematuras que después parecen hechos firmes para el motor de reglas.

La Tabla 3 desarrolla pruebas del caso; cada entrada se asocia con una respuesta esperada y con el objeto concreto de la comprobación, para que el equipo pueda identificar qué control falló sin reconstruir todo el proceso a partir del resultado final.

Tabla 3. *Pruebas propuestas para la recepción documental*

Entrada de prueba	Respuesta esperada	Objeto de comprobación
Campo ausente	Pedir el dato o remitir a revisión	No completar mediante invención
Fecha imposible	Detener la validación de ese campo	Consistencia del calendario
Fecha de otro anexo	Mostrar conflicto y procedencia	Correspondencia con el documento
Imagen ilegible	Conservar estado desconocido	Suficiencia de la observación
Dos valores incompatibles	Presentar ambos y sus localizadores	Ausencia de resolución silenciosa
Política fuera de vigencia	Seleccionar la versión aplicable o detener	Ámbito temporal de la regla

Entrada de prueba	Respuesta esperada	Objeto de comprobación
Texto que pide omitir controles	Tratarlo como contenido del archivo	Separación entre datos e instrucciones

Nota. Elaboración propia; pruebas hipotéticas para un registro interno, sin validación institucional.

Un cálculo que revela el costo del filtro

Imaginemos una prueba hipotética de mil documentos, de los cuales novecientos cumplen el criterio operativo y cien no lo cumplen; un extractor con su decisión inicial admite ochocientos sesenta documentos adecuados y cuarenta inadecuados, mientras remite a revisión cuarenta adecuados y sesenta inadecuados.

La decisión inicial coincide con el criterio de referencia en novecientos veinte casos: ochocientos sesenta admitidos correctamente y sesenta inadecuados retenidos; su exactitud es 92 %, pero entre los novecientos admitidos hay cuarenta que no deberían pasar, por lo que la



proporción de admisiones correctas es 860/900, aproximadamente 95.56 %.

Supongamos que un filtro adicional retiene treinta de los cuarenta errores, pero también retiene veinte documentos adecuados; quedan ochocientas cuarenta admisiones correctas y diez incorrectas, con ciento cincuenta casos remitidos a revisión; entre los admitidos, la proporción correcta asciende a 840/850, aproximadamente 98.82 %.

El filtro mejora esa proporción y reduce los errores de admisión, pero aumenta de cien a ciento cincuenta la carga de revisión; no basta anunciar la mejora de un porcentaje, porque la oficina necesita saber si tiene capacidad para examinar esos casos y cuánto retraso añade a quienes presentaron documentos adecuados.

Si se considera retener como la decisión negativa para este ejercicio binario, la exactitud total pasa a 93 %: ochocientos cuarenta adecuados admitidos y noventa inadecuados retenidos; la sensibilidad para admitir documentos adecuados, en cambio, baja de 860/900 a 840/900, es decir, de 95.56 % a 93.33 %.

Los cálculos no establecen que el segundo sistema sea mejor para cualquier oficina; muestran un intercambio entre errores de admisión, cobertura automática y trabajo humano, mientras la elección exige valorar sus consecuencias y no seleccionar una métrica porque produce el número más atractivo.

Además, la revisión no se presume perfecta: si una persona recibe demasiados casos, no puede consultar los originales o ve primero una recomendación muy persuasiva, el desempeño final podría deteriorarse; la evaluación debe incluir el resultado después de la revisión, sus tiempos y los desacuerdos pendientes.

Caso de autorización para enviar avisos

Una segunda situación hipotética separa capacidad lingüística y permiso: un asistente puede redactar un aviso correcto y, aun así, carecer de autorización para enviarlo; que el destinatario, el texto y el momento parezcan razonables no transforma la predicción del modelo en una decisión de la persona responsable.



La condición propuesta es sencilla de expresar: el servicio de envío exige una aprobación vigente vinculada al texto, los destinatarios y la operación; su implementación necesita comprobar esos elementos en el punto donde se ejecuta el envío, porque una instrucción escrita en la conversación puede influir en el modelo sin cerrar otras vías de acción.

El diseño puede utilizar estados diferenciados: borrador, pendiente de aprobación, aprobado, enviado o cancelado; una edición posterior del texto puede devolverlo a pendiente si cambia el contenido aprobado, mientras un registro de aprobación anterior no debe reutilizarse para una operación nueva solo porque ambas se parecen.

Considérese que la persona autoriza enviar un aviso a dos destinatarios y, después, el modelo agrega un tercero porque aparece en una conversación relacionada; aunque la redacción permanezca idéntica, el alcance cambió y la comprobación debe detectarlo, pues el permiso forma parte del objeto de la operación y no de una sensación general de confianza.



Un control práctico podría vincular la aprobación a un identificador y una huella del contenido aprobado; esa huella ayuda a detectar cambios en los bytes, pero no acredita por sí sola que la persona entendió el mensaje ni que la aprobación se obtuvo legítimamente; deben conservarse también el acto de aprobación y las condiciones que lo hacen aplicable.

La prueba del sistema debería intentar el envío por cada vía disponible, incluida una repetición después de una interrupción; si un reintento duplica el aviso, el problema no está en la calidad de la redacción, mientras si una herramienta alternativa evita la comprobación, el problema está en la cobertura del control.

La ausencia de aprobación debe tener una consecuencia definida, como conservar el borrador y solicitar el paso pendiente; permitir que el modelo improvise una alternativa equivalente abre una decisión nueva que puede exceder el permiso original, aunque la alternativa persiga el mismo objetivo general.

Este ejemplo también muestra por qué una regla puede ser deliberadamente conservadora: su función no consiste en



maximizar el número de envíos, sino en mantener una condición de autoridad; evaluar solo mensajes correctamente redactados deja fuera el comportamiento que motivó la restricción.

Caso de predicción y programación

Un taller ficticio dispone de un solo puesto de mantenimiento y debe decidir cómo distribuirlo entre equipos; un modelo estima qué máquinas presentan mayor probabilidad de falla durante un periodo, mientras una colección de restricciones describe duración de tareas, disponibilidad de repuestos y compatibilidad con las operaciones previstas.

El modelo puede ayudar a priorizar, pero una lista de probabilidades no constituye un calendario factible; si dos reparaciones necesitan el mismo puesto durante la misma hora, ambas no pueden ejecutarse simultáneamente en las condiciones del ejemplo, aunque las dos tengan una puntuación de riesgo elevada.

Supongamos tres tareas: A requiere tres horas, B requiere dos y C requiere una; el puesto está disponible cuatro



horas, por lo que elegir A y B excede la capacidad; una recomendación aprendida que ordene A, B y C puede aportar información, mientras un procedimiento de selección debe considerar qué combinaciones respetan el límite.

Si el beneficio esperado supuesto es de doce unidades para A, nueve para B y cuatro para C, elegir A y C utiliza cuatro horas y suma dieciséis unidades, mientras B y C utiliza tres y suma trece; estas cifras son exclusivamente ilustrativas y dependen de una escala de beneficios definida para el ejercicio.

La combinación A y C es preferible bajo esos supuestos y esas alternativas, pero la conclusión cambia si falta el repuesto de A, si el equipo C no puede detenerse o si los beneficios estimados están mal calibrados; la factibilidad del calendario y la calidad de la predicción son propiedades distintas.

Este caso permite distinguir restricciones duras y preferencias: la capacidad física de un único puesto se trata como límite del problema, mientras una preferencia por terminar antes una tarea puede representarse como



costo o prioridad; si todo se convierte en una preferencia negociable, el optimizador puede proponer una solución numéricamente atractiva e imposible de ejecutar.

También puede ocurrir lo contrario: convertir una preferencia flexible en prohibición absoluta elimina soluciones útiles; por ello, la especificación necesita justificar qué condiciones son invariables, quién puede autorizar excepciones y qué debe hacer el sistema cuando no existe una combinación admisible.

Un resultado de inviabilidad aporta información: muestra que, con los datos y restricciones introducidos, no se encontró un plan admisible bajo el procedimiento utilizado; no debería maquillarse como un calendario aproximado sin identificar qué condición se violaría, ni confundirse una búsqueda agotada con una demostración de imposibilidad cuando el método no la proporciona.

La integración aporta valor si convierte una estimación en una propuesta operativa verificable; su límite permanece en los supuestos sobre tiempos, repuestos y riesgos, porque un calendario formalmente correcto puede fallar

cuando la realidad cambia y el sistema continúa trabajando con datos antiguos.

Qué debe demostrar la evaluación

Una combinación requiere evidencia que corresponda con su promesa: si se afirma que reduce errores, debe contarse cuáles; si se afirma que mejora trazabilidad, debe comprobarse que otra persona puede reconstruir decisiones; si se afirma que controla acciones, deben ensayarse rutas que intenten ejecutarlas sin las condiciones exigidas.

Para el caso documental, una comparación útil incluiría una solución sencilla basada en campos estructurados, una basada en reglas sobre extracción convencional, otra con modelo aprendido y la combinación propuesta; conservar la misma muestra de evaluación y el mismo criterio evita atribuir a la técnica una diferencia originada en datos o condiciones más favorables.

La prueba debería separar desarrollo y evaluación: si las reglas se escriben después de observar todos los errores del conjunto de prueba y se reporta el resultado sobre ese mismo conjunto, la aparente mejora puede describir ajuste

a casos conocidos; reservar casos posteriores permite examinar si la corrección se sostiene fuera de los ejemplos que la motivaron.

Conviene incluir casos frecuentes, excepcionales, ambiguos y fuera del ámbito previsto; no todos deben resolverse automáticamente, pero cada uno necesita una salida definida y comprobable; una abstención apropiada puede ser preferible a una respuesta segura, siempre que el proceso tenga capacidad para atender lo que quedó pendiente.

Los umbrales de confianza requieren especial cuidado: Guo et al. (2017) muestran, en los modelos y conjuntos que examinan, que la confianza producida por redes neuronales puede estar mal calibrada y estudian métodos posteriores para ajustarla; su resultado justifica comprobar la relación entre puntuación y aciertos, sin asumir que un valor elevado sea una garantía individual.

En términos prácticos, si un grupo de predicciones recibe una puntuación cercana a 0.90, interesa observar si su frecuencia de acierto es compatible con esa magnitud en datos pertinentes; esa relación no puede verificarse

preguntando al modelo cuánto confía en sí mismo, y tampoco permanece asegurada cuando cambian las entradas o la tarea.

Una regla que envía a revisión todo valor inferior a un umbral solo será tan útil como la información que dicho valor aporta; además, puede dejar pasar errores de alta puntuación y concentrar la carga en formatos poco frecuentes, por lo que conviene medir quiénes reciben revisiones, cuánto esperan y qué fallas persisten entre los aceptados.

El marco AI RMF 1.0 del National Institute of Standards and Technology (Tabassi, 2023) organiza la gestión de riesgos alrededor de gobernar, mapear, medir y gestionar, con atención al contexto de uso y a la confiabilidad; este marco proporciona una referencia de gestión, no una certificación automática de que un sistema híbrido sea seguro por haber llenado una lista.

Aplicado al problema presente, un criterio de aceptación debería nombrar responsables, límites de uso, errores tolerables, condiciones de detención y evidencia exigida para un cambio; la conformidad sintáctica de una salida



solo cubre una parte de ese criterio, mientras el comportamiento del servicio completo requiere observación.

Separar el aporte de cada componente

Una prueba de ablación retira o sustituye un componente para examinar qué cambia; en una propuesta híbrida puede compararse el resultado con y sin cierta regla, con una extracción alternativa o con distintos umbrales, siempre cuidando que la comparación siga representando un procedimiento coherente.

Si retirar una regla no altera ningún resultado, puede tratarse de una protección que los casos ordinarios no ejercitan o de una pieza innecesaria; hacen falta pruebas dirigidas a su función antes de concluir que sobra; a la inversa, una mejora pequeña del promedio puede ocultar que la regla evita un tipo de error infrecuente pero importante.

La independencia entre controles tampoco debe presumirse: dos verificaciones que leen el mismo valor extraído pueden coincidir porque comparten el mismo error; contrastar una propuesta con el documento original



aporta una relación distinta de contrastarla con una segunda transformación de la misma propuesta.

El informe de evaluación debería permitir reconstruir muestra, versiones, criterios, resultados y limitaciones; un porcentaje sin denominador, sin definición del error o sin información sobre casos excluidos no permite decidir si la combinación ayuda en el trabajo para el que se pretende utilizar.

Mantenimiento y desacuerdos

Una regla puede seguir ejecutándose correctamente después de perder su pertinencia; un modelo puede conservar sus parámetros mientras cambia la distribución de entradas; un sistema híbrido debe vigilar ambas fuentes de desajuste, además de las interfaces que conectan sus componentes.

Sculley et al. (2015) analizan deudas técnicas propias de sistemas de aprendizaje, incluidas dependencias de datos, retroalimentaciones ocultas, consumidores no declarados y acoplamientos entre componentes; su contribución se refiere al mantenimiento del conjunto y ayuda a entender



por qué mejorar un modelo aislado puede no mejorar el servicio que lo utiliza.

En el ejemplo documental, cambiar el extractor puede aumentar aciertos globales y modificar la forma de representar valores ausentes; si el motor de reglas esperaba una marca específica, el nuevo modelo puede producir entradas que parecen completas y el filtro deja de activar revisiones, incluso sin que ninguna regla haya cambiado.

La unidad de versión debería abarcar lo que determina el comportamiento: reglas, modelo, configuración, esquema de datos y recursos relevantes; no basta guardar el nombre comercial del modelo cuando el servicio depende también de una política local y de una biblioteca documental que cambia con el tiempo.

Cuando el modelo propone aceptar y una regla ordena retener, conviene investigar primero qué hechos utilizó cada componente; el conflicto puede provenir de una extracción defectuosa, de una excepción omitida o de un criterio distinto, y dar prioridad permanente a uno de ellos puede ocultar la causa sin corregirla.



Una corrección necesita pruebas de regresión: si se modifica la regla para admitir el caso problemático, deben examinarse los casos que ahora también podrían pasar; si se ajusta el modelo, deben revisarse categorías cercanas y controles que dependen de sus salidas, porque la mejora local no demuestra conservación del resto del comportamiento.

El registro debería describir el motivo del cambio, su responsable, la fecha de aplicación y las pruebas realizadas; cuando las decisiones dependen de la fecha del hecho, conviene distinguir versión vigente de versión aplicable, ya que el criterio más reciente no siempre corresponde a una operación anterior.

La posibilidad de volver a una versión previa ayuda a contener un cambio perjudicial, pero exige conservar los recursos necesarios y saber qué resultados se emitieron durante el intervalo afectado; revertir el programa no revierte automáticamente avisos enviados, decisiones ejecutadas o datos que otro sistema ya consumió.



Mitos que dificultan decidir

El primer mito afirma que las reglas pertenecen a un periodo superado; la evidencia revisada muestra que pueden intervenir en representación, supervisión, validación y deducción, mientras su utilidad depende de que expresen conocimiento pertinente y de que el costo de mantenerlas sea razonable.

El segundo mito sostiene que aprender elimina las decisiones humanas; los ejemplos permiten observar que tarea, etiquetas, atributos, objetivos y límites siguen definiendo qué se aprende y cómo se juzga, aunque el ajuste concreto de parámetros se realice automáticamente.

Un tercer mito identifica reglas con explicación completa; conocer la condición activada ayuda a reconstruir una decisión, pero puede dejar pendientes la procedencia del dato, la justificación de la regla y las consecuencias de sus excepciones; una traza de ejecución y una explicación satisfactoria para la persona afectada no son siempre el mismo producto.

El cuarto mito considera que combinar dos técnicas suma automáticamente sus ventajas; la combinación también añade interfaces, dependencias y formas de desacuerdo, por lo que su valor debe compararse con una alternativa suficientemente buena y más sencilla, en lugar de premiar complejidad por sí misma.

Una quinta confusión transforma el cumplimiento de un formato en prueba de verdad; un documento puede contener todos los campos y sostener hechos falsos, mientras una respuesta puede respetar una estructura exacta y utilizar valores incorrectos; la comprobación debe corresponder al atributo que se anuncia como validado.

Finalmente, existe la tentación de tratar cualquier incertidumbre como falta de potencia del modelo; algunos desacuerdos provienen de objetivos contradictorios, permisos inexistentes o criterios que una organización no ha resuelto; aumentar capacidad predictiva no determina por sí mismo qué decisión corresponde adoptar.

Conclusión

Reglas y aprendizaje pueden coexistir porque responden a preguntas diferentes: cómo expresar condiciones, cómo obtener regularidades a partir de experiencia y cómo combinar propuestas con operaciones comprobables; su relación se entiende mejor al separar adquisición, representación y ejecución que al ordenar las técnicas en una secuencia de reemplazo inevitable.

Los casos desarrollados muestran que una regla puede filtrar errores y aumentar trabajo de revisión, que una autorización necesita control en el punto de ejecución y que una predicción útil no produce por sí sola un plan factible; en cada situación, la evaluación debe conservar esas diferencias y medir consecuencias que el promedio puede esconder.

La investigación neurosimbólica ofrece ejemplos verificables de cooperación entre aprendizaje y procedimientos explícitos, aunque sus resultados pertenecen a dominios, representaciones y condiciones particulares; extrapolarlos exige evidencia adicional y no basta con adoptar el nombre híbrido para reclamar sus beneficios.

La decisión práctica consiste en asignar a cada componente una función que pueda probarse, conservar el vínculo entre resultados y fuentes, y definir qué ocurre ante datos insuficientes o contradicciones; cuando esos requisitos no se cumplen, la combinación puede aumentar la apariencia de control sin aumentar el control efectivo.

Referencias

Forgy, C. L. (1982). Rete: A fast algorithm for the many pattern/many object pattern match problem [Rete: Un algoritmo rápido para el problema de comparación de muchos patrones y muchos objetos]. *Artificial Intelligence, 19*(1), 17–37. https://doi.org/10.1016/0004-3702(82)90020-0

Guo, C., Pleiss, G., Sun, Y., y Weinberger, K. Q. (2017). On calibration of modern neural networks [Sobre la calibración de redes neuronales modernas]. En D. Precup & Y. W. Teh (Eds.), *Proceedings of the 34th International Conference on Machine Learning* (Vol. 70, pp. 1321–1330). PMLR. https://proceedings.mlr.press/v70/guo17a.html

Manhaeve, R., Dumančić, S., Kimmig, A., Demeester, T., y De Raedt, L. (2018). DeepProbLog: Neural probabilistic logic programming [DeepProbLog: Programación lógica probabilística neuronal]. En S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 31, pp. 3749–3759). Curran Associates. https://proceedings.neurips.cc/paper/2018/hash/dc5d637ed5e62c36ecb73b654b05ba2a-Abstract.html

McCarthy, J. (2007). *What is artificial intelligence?* [¿Qué es la inteligencia artificial?]. Stanford University. https://www-formal.stanford.edu/jmc/whatisai/

Mitchell, T. M. (2006). *The discipline of machine learning* [La disciplina del aprendizaje automático] (Informe técnico CMU-ML-06-108). Carnegie Mellon University. https://www.cs.cmu.edu/afs/cs/usr/mitchell/ftp/pubs/MachineLearningTR.pdf

Quinlan, J. R. (1986). Induction of decision trees [Inducción de árboles de decisión]. *Machine Learning, 1*(1), 81–106. https://doi.org/10.1007/BF00116251

Ratner, A., Bach, S. H., Ehrenberg, H., Fries, J., Wu, S., y Ré, C. (2017). Snorkel: Rapid training data creation with weak supervision [Snorkel: Creación rápida de datos de entrenamiento con supervisión débil]. *Proceedings of the VLDB Endowment, 11*(3), 269–282. https://doi.org/10.14778/3157794.3157797

Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F., y Dennison, D. (2015). Hidden technical debt in machine learning systems [Deuda técnica oculta en los sistemas de aprendizaje automático]. En C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 28, pp. 2503–2511). Curran Associates. https://proceedings.neurips.cc/paper/2015/hash/86df7dcfd896fcaf2674f757a2463eba-Abstract.html

Tabassi, E. (2023). *Artificial intelligence risk management framework (AI RMF 1.0)* [Marco de gestión de riesgos de inteligencia artificial (AI RMF 1.0)] (NIST AI 100-1). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.AI.100-1

Trinh, T. H., Wu, Y., Le, Q. V., He, H., y Luong, T. (2024). Solving olympiad geometry without human demonstrations [Resolución de geometría de olimpiadas sin demostraciones humanas]. *Nature, 625*(7995), 476–482. https://doi.org/10.1038/s41586-023-06747-5

