CAPÍTULO 5

# Grandes éxitos de la inteligencia artificial

En este capítulo haremos un repaso de algunas de las aplicaciones más exitosas de la IA focalizando nuestra atención en los últimos 20 años. Varias de estas aplicaciones han sido muy mediáticas, entre ellas están Deep Blue, AlphaGo, Watson, vehículos autónomos, robots en Marte, videojuegos, etc. En todos los casos describimos cuáles han sido las técnicas de IA aplicadas. Creemos que las aplicaciones seleccionadas son claramente representativas de las capacidades de la IA en el segundo decenio del siglo XXI.

## Juegos por ordenador

Los juegos de ordenador han sido un objetivo de la IA desde sus inicios (Russell y Norvig, 2010). En particular, el juego del ajedrez ha sido durante décadas un objetivo permanente; dentro de la IA también se han estudiado otros juegos como las damas, Othello o Go. Estos son juegos denominados de *información perfecta*, porque no hay elementos ocultos ni estocásticos: toda la información disponible para un jugador también está disponible para el otro. Estos juegos se denominan *de suma cero* porque si un jugador obtiene una utilidad $x$, el otro recibe $-x$. Otros juegos, como el póquer o el *bridge*,

son de información imperfecta porque hay elementos del juego que son conocidos por unos jugadores, pero no por otros. En otros casos hay elementos estocásticos, como el lanzamiento de dados. Consideramos el escenario en donde un jugador es humano y el ordenador es el otro jugador; lo restringimos a dos jugadores, aunque hay extensiones a esquemas multijugador. En lo que sigue, llamaremos a nuestros jugadores A y B.

La aproximación clásica de la IA a este tipo de juegos es la denominada *minimax* (Ginsberg, 1993; Russell y Norvig, 2010). Se representa internamente mediante un *árbol de juegos*. Supongamos que le toca jugar a A: la raíz del árbol es el estado actual, con tantos sucesores como jugadas distintas posibles puede hacer A. Para cada una de esas jugadas, consideramos todas las posibles respuestas de B; a su vez, para cada una de estas respuestas, consideramos todas las posibles respuestas de A, etc. En el árbol de búsqueda los jugadores están alternados por niveles, y crece multiplicativamente en cada nivel.

Si pudiéramos desarrollar el árbol de juegos hasta posiciones ganadoras o perdedoras, que etiquetaríamos con +1 (ganadora) o -1 (perdedora) —con 0 para posiciones de empate—, y propagáramos hacia arriba —desde las hojas hasta la raíz— el mejor movimiento posible para cada jugador (están alternados por niveles, el mejor movimiento para A es el máximo de sus sucesores, mientras que el mejor movimiento para B es el mínimo de sus sucesores), al final el jugador A conocería la mejor jugada para él en esa tirada. El recorrido del árbol propagando hacia arriba los valores de las hojas del árbol lo realiza el conocido algoritmo minimax en su versión de búsqueda en profundidad. Si somos capaces de, mediante heurísticas, ordenar los nodos sucesores de un estado, de forma que las posiciones ganadoras se agrupen a la izquierda del árbol, la exploración mediante el algoritmo *alfa-beta* permite podar zonas ramas del árbol que representan zonas del espacio de búsqueda en donde no hay soluciones mejores que las que ya hemos obtenido. En la práctica, alfa-beta produce mejoras sustanciales con respecto al rendimiento de minimax.

La realidad es que no se puede desarrollar completamente el árbol de juegos porque es demasiado grande (un simple cálculo es útil aquí: el factor de ramificación del ajedrez es en torno a 35 y las partidas a menudo alcanzan 50 movimientos por jugador; el árbol de juegos tiene en torno a $35^{100}$ nodos, imposible de ser recorrido); solo se puede buscar hasta cierta profundidad, en donde no hay posiciones ganadoras o perdedoras. Los nodos a esa profundidad se han de evaluar mediante una *función de evaluación*, que retorna un número en el intervalo [+1,-1] y se propaga hacia la raíz mediante los algoritmos mencionados anteriormente. La profundidad a la que se explora el árbol de búsqueda no ha de ser necesariamente fija, puede variar en función de la zona. Un ejemplo es la *no quiescencia*, cuando el valor de la función de evaluación cambia rápidamente entre nodos a diferentes niveles (en ajedrez sucede cuando se capturan piezas). En ese caso, se recomienda continuar la búsqueda hasta que se recupera la quiescencia, y la función de evaluación vuelve a cambiar suavemente entre niveles.

El *efecto horizonte* aparece precisamente como resultado de buscar hasta determinada profundidad. Si una jugada aparentemente buena esconde un mal final detrás del horizonte de búsqueda (la profundidad a la que se busca), estamos indefensos frente a esto (que únicamente tiene un remedio: buscar más profundo, pero ¿dónde?). Una forma de enfrentar el efecto horizonte son las *extensiones singulares*. Si el algoritmo elige una jugada que solo está soportada por un nodo con una clara diferencia con sus nodos hermanos, se recomienda buscar más profundo bajo ese nodo. Se trata de confirmar la evaluación de esa jugada.

Hay juegos en los que la idea de función de evaluación, imprescindible en juegos de gran tamaño, no funciona (en el sentido de que no se conoce como construirla). Para esos juegos, una estrategia alternativa a la búsqueda minimax es la MCTS (Monte Carlo Tree Search) (Browne *et al.*, 2012).

MCTS desarrolla el árbol de juegos muy parcialmente (solo se memorizarán nodos que estén cerca de la raíz),

suponiendo que la bondad de un nodo se puede estimar basándose en simulaciones del juego que parten de ese nodo hasta llegar a un nodo terminal (posición ganadora o perdedora). MCTS realiza un número de iteraciones hasta que se alcanza un máximo (en tiempo, número de iteraciones, consumo de memoria). Una iteración tiene cuatro fases:

1. *Selección*: partiendo de la raíz (estado actual) se selecciona un nodo hijo siguiendo una *política de árbol* (*tree policy*); este proceso se itera hasta encontrar un nodo no terminal con descendientes no visitados (nodo expandible).
2. *Expansión*: se añade un nuevo descendiente de ese nodo expandible, representa la ejecución de una acción.
3. *Simulación*: a partir de ese nuevo nodo, se eligen acciones alternativamente para los jugadores A y B hasta llegar a un nodo terminal (se simula la realización del juego entre A y B). La selección de acciones sigue la *política por defecto* (*default policy*).
4. *Actualización*: el resultado se registra en los nodos que conectan el nodo terminal con la raíz, actualizando sus estadísticas.

Este proceso depende de esas dos políticas mencionadas:

1. *Política de árbol*: se trata de seleccionar el nodo hijo del nodo actual, hasta llegar a nodos expandibles. Una política que ha dado buenos resultados es la UCB1, que elige la acción con más éxito hasta ahora, combinada con un elemento de exploración.
2. *Política por defecto*: en el paso 3, simulación, se trata de seleccionar acciones para los jugadores A y B hasta llegar a un nodo terminal. El caso más sencillo es una política aleatoria: escoger las acciones al azar.

MCTS desarrolla el árbol de juegos de forma asimétrica (las mejores acciones se seleccionan con más frecuencia). En

algunos casos, se han desarrollado estrategias de poda. Cuando se alcanza el límite, se selecciona la acción con más éxito.

Otro conjunto de juegos incluyen elementos estocásticos —como tirar un dado— que determinan la evolución posterior del mismo. Un ejemplo es el Backgammon, donde las jugadas posibles de cada jugador dependen del resultado del lanzamiento de dos dados. El concepto de árbol de juegos permanece, aunque se modifica su estructura: a cada posible jugada de A, se consideran todos los posibles resultados de tirar los dados y, para cada resultado, todos los posibles movimientos de B. El árbol de juegos tiene niveles intermedios entre las posibles jugadas de A y las de B, que representan los distitos resultados de los dados. Este árbol de juegos se recorre con un algoritmo alfa-beta para propagar hacia arriba los valores de los nodos terminales con el siguiente criterio: si el nodo corresponde a A, se propaga el máximo de los sucesores; si el nodo corresponde a B, se propaga el mínimo de los sucesores, y si el nodo corresponde a una tirada de dados, se propaga el valor esperado de minimax, que es la suma de los valores de cada sucesor multiplicado por su probabilidad. El estado actual de juegos por ordenador aparece reflejado en Russell y Norvig (2010). A continuación, presentamos tres juegos que han atraído mucha atención por parte de los investigadores de IA.

Sobre el ajedrez, en 1997 un equipo de IBM compuesto por seis personas (Feng-Hsiung Hsu, Joe Hoane, Chung-Jen Tan, Joel Benjamin, Jerry Brody y Murray Campbell), que habían trabajado tres años desarrollando el programa Deep Blue de ajedrez por ordenador, ganó al campeón del mundo en ese momento, Gari Kasparov, en un torneo celebrado a seis partidas según condiciones internacionales. Deep Blue era capaz de explorar 200 millones de posiciones por segundo, realizaba un alfa-beta paralelo, utilizaba una extensa biblioteca de aperturas y finales (4.000 aperturas, todos los finales con 5 piezas y muchos con 6 piezas), e incluía novedades como las extensiones singulares. Disponía de una extensa base de datos con 700.000 partidas a nivel de gran maestro para extraer recomendaciones. Realizaba de forma habitual

búsquedas a profundidad 14, aunque en algún caso llegó a profundidad 40. La función de evaluación incluía 8.000 variables. Kasparov, poco feliz con el resultado del torneo, tuvo que reconocer que "la cantidad se había vuelto calidad".

Sobre el juego de damas, el actual campeón (en su versión inglesa) es el programa CHINOOK, que implementa también una búsqueda alfa-beta y ganó al campeón mundial en 1990. Desde 2007, es capaz de jugar perfectamente, es decir, que nunca puede perder, combinando alfa-beta con una base de datos de millones de posiciones finales.

El juego del Go es, sin duda, el más complicado por el gran número de acciones posibles en cada jugada (el clásico árbol de juegos tiene un factor de ramificación en torno a 250), con partidas muy largas (hasta 150 movimientos). Para este juego, el esquema minimax no funciona (no se conocen buenas funciones de evaluación, el factor de ramificación es muy grande). Sin embargo, el esquema MCTS sí es útil. Con políticas genéricas, hay programas que alcanzan un nivel de aficionado débil. Cuando estas políticas se entrenan con movimientos de expertos humanos, el nivel de juego se incrementa hasta un aficionado experto. En 2015 se desarrolló el programa AlphaGo, que ganó por 5 a 0 al humano campeón mundial *oficioso* de este juego. AlphaGo combina MCTS con dos redes neuronales profundas que permiten mejorar las políticas que se aplican; el resultado de MCTS se combina con el de una red neuronal. Los resultados obtenidos son excelentes.

## Robots

En esta sección hablaremos de algunos éxitos indiscutibles de aplicaciones de la robótica. No vamos a incluir ninguna aplicación a la robótica que no tenga que ver claramente con el uso de la IA[33]. De entre los numerosos éxitos, nos limitaremos

33. Para otros éxitos de la robótica no relacionados con la IA, de nuevo remitimos al lector al libro de García-Armada (2015).

a los que consideramos más representativos. De hecho, tres de los robots descritos en esta sección (Spirit, Opportunity y Stanley), junto con Shakey y Robby, el robot de ficción de la película de 1956 *El planeta prohibido*, constituyen, según la revista *Wired*, los cinco mejores robots de la historia.

### Robots en el espacio

El primer robot que la NASA envió a Marte en 1997, Sojourner, medía solamente 65 cm de largo por 48 de ancho y 30 de alto, y pesaba unos 10 kg. Durante los aproximadamente 83 días que estuvo operativo (correspondientes a 85 días de la Tierra), recorrió aproximadamente 100 metros, a una velocidad de un centímetro por segundo, y nunca se alejó más de unos 12 metros de la estación Pathfinder. Envió 555 fotografías y tomó muestras del suelo y de rocas en 16 lugares distintos para poder analizar sus propiedades químicas mediante un instrumento que usaba espectrometría por rayos X. Aunque también estaba equipado con dos cámaras para captar posibles obstáculos durante sus limitados desplazamientos, de hecho no planificaba autónomamente su navegación, sino que desde la NASA se enviaban las instrucciones paso a paso al sistema que controlaba los giros y desplazamientos del robot; por consiguiente, no estaba equipado con ningún sistema de IA, pero sirvió para abrir el camino para futuros robots que, como veremos a continuación, ya incorporaron la IA en mayor o menor grado.

En 2004, la NASA envió a Marte los robots Spirit y Opportunity. Ambos unas cinco veces más grandes que Sojourner y con un peso de unos 180 kg. En principio debían estar operativos durante 90 días, pero ambos han excedido con creces estas expectativas. De hecho, Spirit estuvo operativo hasta el año 2009, cuando quedó bloqueado contra un obstáculo mientras navegaba. Sin embargo, Opportunity continuaba estando parcialmente operativo a finales de 2016, ha tomado más de 200.000 fotografías y ha recorrido más de 40 km durante estos 12 años, aunque sus espectrómetros

dejaron de funcionar. El sistema de navegación de Spirit y Opportunity, aunque no es completamente autónomo, es mucho más sofisticado que el de Sojourner, que funciona de la siguiente manera: desde la NASA se envían las coordenadas precisas de localizaciones muy cercanas a la posición del robot, que forman una secuencia hacia el objetivo final a alcanzar. El sistema de navegación del robot avanza hacia dichas localizaciones, evitando posibles obstáculos no detectados por el teleoperador. Cada vez que el robot alcanza una de estas localizaciones, el teleoperador envía las coordenadas de la siguiente y así sucesivamente hasta alcanzar la posición objetivo final. Se trata todavía de una IA limitada, como lo demuestra el hecho que Spirit quedó bloqueado para siempre por un obstáculo que no pudo evitar.

La experiencia adquirida con estos robots permitieron a la NASA enviar de nuevo un robot a Marte en 2012 que a día de hoy sigue funcionando plenamente. El robot, llamado Curiosity, pesa unos 900 kg y en lugar de paneles solares se alimenta de energía nuclear. Se encuentra en una zona en las antípodas del robot Opportunity. Su misión también consiste en analizar suelo y rocas y encontrar indicios de presencia de agua en el pasado. Desde el punto de vista de la IA, Curiosity es mucho más sofisticado que sus antecesores en Marte. Por una parte, su *software* de navegación y su sistema de visión estereoscópica, equipado con un sofisticado *software* de análisis de imágenes basado en detección de contornos y otras características tales como brillo, tamaño, forma, orientación, etc., le permiten desplazamientos autónomos mucho más largos (del orden de decenas de metros) que los de Spirit y Opportunity. Además, a mediados de 2016, la NASA incorporó remotamente un nuevo *software* denominado AEGIS. Este *softwar*e, desarrollado en el Jet Propulsion Laboratory, permite a Curiosity tomar decisiones sobre los objetivos científicos a los que dirigirse para llevar a cabo medidas usando la instrumentación con que está equipado. AEGIS permite apuntar el láser, que efectúa la medición obteniendo datos geoquímicos, sobre objetivos

muy pequeños con una precisión superior a la que puede lograr un operador humano teleoperando desde una sala de control de la NASA. Además, es capaz de priorizar sus objetivos gracias a informaciones previamente proporcionadas por los científicos sobre las características que deben tener las rocas a analizar. Esta capacidad de decisión autónoma es muy importante, ya que reduce muy significativamente el tiempo necesario para llevar a cabo las mediciones y experimentos científicos, debido a que Curiosity no tiene que estar horas esperado a que lleguen las instrucciones transmitidas desde la NASA.

Otra presencia de IA en el espacio fue la sonda espacial llamada *Deep Space 1* equipada con el *software* llamado Agente Remoto (Remote Agent, RA), que la pilotaba. *Deep Space* fue lanzada en 1998 con el único objetivo de verificar el comportamiento de una serie de tecnologías en el espacio, entre ellas el agente remoto RA. Este *software* era un agente inteligente que planificaba y ejecutaba las acciones de la nave. El planificador recibía instrucciones del Gestor de Misiones (Mission Manager, MM) y usaba búsqueda heurística para generar los planes que a continuación se ejecutaban, mediante un sistema en tiempo real. También disponía de un módulo que monitorizaba la ejecución de las acciones del plan con el fin de identificar posibles fallos e intentar corregirlos. Algunos ejemplos de acciones eran conectar y desconectar los sistemas de propulsión que permitían orientar la nave, posicionar la cámara, etc. Un aspecto interesante de este sistema es que los operadores desde una sala de control en la NASA podían enviar instrucciones relativas a objetivos de alto nivel al MM como, por ejemplo: "Durante las próximas 48 horas toma fotografías de los asteroides X, Y, Z, utilizando un 90% del empuje del motor durante todo ese tiempo", en lugar de tener que proporcionar una secuencia detallada de acciones distribuidas en el tiempo a lo largo de dichas 48 horas. Gracias a este objetivo de alto nivel, era el planificador quien generaba las acciones detalladas para cumplir el objetivo.

## Vehículos autónomos

A mediados de los años ochenta, la agencia americana DARPA lanzó su primer programa de I+D sobre vehículos terrestres autónomos con fines principalmente militares. En este programa participaron diversas universidades de primer nivel en Estados Unidos bajo la coordinación de la empresa Martin Marietta y del ejército estadounidense. Ello dio como resultado un vehículo (ALV) todoterreno, de 8 toneladas de peso, equipado con una cámara y un escáner láser que proporcionaban información sobre las posiciones de los bordes de la carretera que servían para generar un modelo de lo que había frente al vehículo. Un módulo de navegación y de búsqueda de objetivos visuales llevaba a cabo un proceso de razonamiento basado en reglas y planificaba las acciones que el piloto automático ejecutaba para guiar el vehículo, procurando no salirse de la carretera y evitando obstáculos. Dada la limitada capacidad de cálculo de los procesadores de aquella época y las limitaciones de los sistemas de visión artificial, este vehículo únicamente se podía desplazar a velocidades por debajo de los 10 km/h y los desplazamientos eran de muy pocos kilómetros de distancia. Además, los obstáculos eran artificiales y se colocaban a no menos de 100 metros los unos de los otros. Los obstáculos más pequeños que ALV podía detectar medían cerca de 1 metro de alto. La visión artificial era el problema principal, ya que en aquellos años la velocidad de procesamiento disponible estaba varios órdenes de magnitud por debajo de la necesaria para detectar robustamente en tiempo real la gran diversidad de obstáculos naturales y la carretera. De hecho, el 85% de la carga total de procesamiento lo usaba el sistema de visión y aun así era muy poco robusto, ya que era muy sensible a las variaciones de las condiciones de iluminación debidas a la posición del sol, la presencia de sombras, etc. En testeos sucesivos llevados a cabo con pocas horas de diferencia los resultados podían ser completamente distintos, no solamente detectando obstáculos, sino también detectando los bordes de la carretera. En 1988,

DARPA canceló el programa, pero el interés por desarrollar vehículos autónomos continuó tanto en DARPA como en universidades y centros de investigación. En 2002, DARPA anunció lo que llamó *el gran desafío* (DARPA Grand Challenge), de nuevo con fines militares, ofreciendo un premio de un millón de dólares al primer vehículo autónomo que fuera capaz de recorrer 142 millas por el desierto de Nevada siguiendo una trayectoria marcada. Todo vehículo que saliera de los límites marcados quedaba descalificado. De los 15 vehículos que tomaron la salida el 13 de marzo de 2004 ninguno completó el recorrido. La mayoría no llegó a recorrer ni una milla y el que llegó más lejos recorrió menos de ocho millas. En la mayoría de casos no fueron capaces de detectar los múltiples obstáculos naturales y en otros casos el sistema de percepción interpretaba las sombras como obstáculos. De nuevo la gran dificultad fueron los sistemas de percepción. Los sistemas de planificación por muy elaborados que sean de nada sirven si falla la detección y localización de los obstáculos. De este fracaso se aprendieron lecciones que permitieron que el siguiente *gran desafío*, que DARPA lanzó un año después, fuera un gran éxito ya que, en 2005, de los 23 vehículos que tomaron la salida en el desierto de Nevada, 5 alcanzaran la meta completando un recorrido de 132 millas. El vencedor fue un equipo de la universidad de Stanford, liderado por Sebastian Thrun, con Stanley, un Volkswagen Touareg modificado que completó el recorrido en 6 horas y 54 minutos. Stanley estaba equipado con un sistema informático compuesto por seis procesadores y un conjunto de sensores que incluían cinco escáneres láser, una cámara de vídeo, un sistema GPS, así como giroscopios y acelerómetros. Desde el punto de vista de la IA incorporaba un novedoso algoritmo de aprendizaje probabilístico que le permitía determinar con qué probabilidad el terreno era transitable, a través de un conjunto de entrenamiento con ejemplos sobre las características visuales asociadas a terrenos transitables y no transitables. Este algoritmo era robusto a posibles errores de los sensores y fue capaz de identificar casi un 100% de los obstáculos, todo ello a una velocidad

que en ocasiones llegó a unos 50 km/h. Otro componente de IA que Stanley incorporaba era otro algoritmo de aprendizaje supervisado para aprender a ajustar la velocidad en función del tipo de terreno de forma que en tramos rectos y con suelo firme aceleraba de forma automática y en tramos que suponían riesgo reducía la velocidad.

En 2007, DARPA organizó un *desafío urbano* (DARPA Urban Challenge), instalando una reproducción de una zona urbana (calles, semáforos, señales de tráfico, etc.) en los terrenos de una base aérea con el fin de comprobar el comportamiento de coches autónomos en dicho entorno. Los equipos participantes tenían que pasar por un conjunto previamente dado de puntos de control recorriendo un total de 60 millas en un tiempo máximo de seis horas y respetando las normas de circulación. Para finalizar con éxito era necesario efectuar satisfactoriamente maniobras tales como adelantar otros vehículos más lentos, cruzar intersecciones, aparcar, desaparcar incorporándose al tráfico, etc. De los 11 equipos finalistas, seis terminaron dentro del límite de tiempo establecido. El vencedor invirtió un poco más de cuatro horas en completar el recorrido. Este desafío planteaba complejos problemas para los algoritmos de planificación y de visión, ya que era necesario reconocer señales de tráfico, detectar otros vehículos, encontrar huecos para aparcar y maniobrar para aparcar, respetar las reglas de preferencia de paso en intersecciones sin semáforo y reaccionar ante imprevistos como, por ejemplo, calzadas bloqueadas por obras o vehículos detenidos en medio de la calzada. El coche vencedor, un Chevrolet modelo Tahoe desarrollado por un equipo de la Universidad de Carnegie Mellon, iba equipado con una docena de láseres, cámara y radares. Un *software* de planificación de alto nivel determinaba el mejor camino entre dos puntos dados de la red de tráfico y otro planificador de movimientos de bajo nivel respondía en tiempo real a obstáculos tanto dinámicos como estáticos.

Los éxitos de esta serie de desafíos de DARPA dispararon el interés de los fabricantes de automóviles para

desarrollar coches autónomos. Actualmente, no solo todos los grandes fabricantes de coches, sino también empresas de otros sectores como Google, Apple, Amazon o Huber están desarrollando tecnología para coches autónomos. Según ha anunciado recientemente el Departamento de Transporte de Estados Unidos, durante 2017 se destinarán 4.000 millones de dólares del presupuesto oficial para el desarrollo de automóviles autónomos. El objetivo a corto plazo es que 2.500 de estos vehículos puedan circular dentro de 2 años. El futuro que está en juego no es únicamente el del automóvil, sino el del transporte en general. Hay especialmente tres tecnologías que están jugando un importante papel en los automóviles: el reconocimiento y síntesis de voz, la cartografía digital avanzada y los sensores y cámaras. Actualmente, las compañías automovilísticas abordan el desafío de adopción de estas tecnologías con estrategias distintas: mientras que Ford prefiere asociarse con fabricantes específicos de componentes, Toyota ha optado por crear una *spin-off* de IA, en colaboración con las universidades de Stanford y MIT, en la que ya ha invertido mil millones de dólares contratando a 200 personas. El objetivo principal es crear un coche incapaz de causar accidentes, pero el gran problema a resolver para tener coches autónomos es la robustez de las tecnologías empleadas, desde el *software* a los sensores, la electrónica y las comunicaciones. En todos esos casos estamos muy lejos de tener tecnologías robustas y, en particular, en el caso del *software*, ya que debe ser *safety critical* y por lo tanto deberá tener garantías absolutas de fiabilidad y de protección ante ciberataques, siendo necesario ir mucho más allá del estado actual de la ingeniería del *software*.

Teniendo en cuenta que las decisiones que dicho *software* va a tener que tomar tendrán que tomarse en milésimas de segundo, es mucho más complejo que, por ejemplo, el *software* que actualmente controla el pilotaje automático de un avión, ya que en el espacio aéreo el tiempo de reacción es por lo menos tres órdenes de magnitud más lento que en el tráfico rodado gracias a que el sistema de control de tráfico

aéreo asegura prácticamente que los aviones más cercanos siempre estarán a cientos de metros de distancia, suposición que no se cumple con el tráfico rodado. Por consiguiente, la autonomía completa, sin ninguna intervención por parte del humano (incluso eliminando el volante, el acelerador y el freno), plantea enormes problemas tecnológicos. Por otra parte, también plantea serios problemas legales, éticos y económicos, además de posibles problemas de aceptación por parte de los usuarios.

Centrándonos en los aspectos técnicos, Google en 2016 dio cifras de cerca de 400 fallos que obligaron a desconectar el piloto automático y, de estos, unos 300 fueron fallos técnicos (fallos de sensores, pérdida de comunicación con Internet, problemas de dirección, problemas de frenado, etc.). Ese mismo año, Nissan confiesa que desconectó el piloto automático también en más de 400 ocasiones, y Mercedes más de 1.000 veces en solamente unos 3.000 km, unas 500 por fallo técnico y el resto por decisión del conductor que expresó "no sentirse cómodo". En decenas de estos casos se evitaron accidentes gracias a la intervención del conductor humano. A corto e incluso a medio plazo, los humanos seguiremos jugando un papel fundamental en la conducción del vehículo, actuando como copilotos y supervisores, pasando el control del vehículo al piloto automático cuando ello sea posible, pero no siempre. A más largo plazo, cuando la tecnología haya resuelto las limitaciones actuales, posiblemente llegue el momento de plantearse seriamente la conducción totalmente autónoma. Otra fuente de problemas es debida a que las personas somos impredecibles en nuestro comportamiento y, en muchos casos, insensatos a la hora de tomar decisiones que pueden ser peligrosas. En estos casos tenemos que tener un *software* suficientemente inteligente como para tomar decisiones que eviten estos peligros. En IA existe un área de investigación llamada *planificación tolerante a fallos*, que puede gestionar situaciones en las que el comportamiento del conductor se desvía del plan correcto. Pero para aplicarlo es necesario hacer estudios sobre aquellos elementos sutiles del comportamiento

humano que deben de tenerse en cuenta para programar un robot (o un coche) para que pueda trabajar semiautónomamente y colaborativamente con un humano. Una vez estos elementos estén suficientemente identificados se deben incorporar al *software* de control, de forma que planifique sus acciones y sea capaz de crear un plan alternativo en situaciones de emergencia. Por ejemplo, cuando gracias a los sensores que monitorizan el grado de atención y fatiga del conductor el coche detecta que el conductor está cansado, el plan alternativo consistiría en comunicar al conductor que debe ceder el control y, si es necesario, proceder a cambiar la ruta inicialmente planificada y circular por carreteras donde es más fácil y seguro conducir de forma automática. En caso de no recibir respuesta, el coche debe tener la capacidad de, autónomamente, tomar la decisión de disminuir la velocidad, situarse en el arcén y parar el coche.

Entonces, ¿cuál es el camino a recorrer hacia el coche autónomo? Se habla de 5 niveles de automatización; los coches más avanzados actualmente se situarían en el nivel 3, lo que significa que el coche puede hacerse con el volante y los pedales, estar atento al entorno, pero depende por completo de la supervisión humana. A partir de 2020 se cree que se logrará el nivel 4, en el que será el coche quien lleve la voz cantante, aunque en ocasiones específicas requerirá intervención humana. El último nivel implicaría que el humano nunca tendrá que hacer nada. Para alcanzar el nivel 4 queda mucho trabajo por hacer. Uno es tener sistemas de localización mucho más precisos que ahora. Para ello se necesitará cartografía digital de muy alta precisión, que ubique objetos y elementos fijos muy concretos tales como los árboles a los lados de una carretera. Estos mapas, además, tendrán que ser en 3D con una resolución inferior a un metro y, para evitar la obsolescencia de las cartografías digitales actuales, será el coche el que los cree sobre la marcha mediante lo que en IA se conoce por SLAM (Simultaneous Localization and Mapping), un algoritmo que se usa extensivamente en robótica que permite que un robot explore un entorno desconocido, haga un mapa del entorno y, simultáneamente, calcule su localización en dicho mapa,

todo ello mediante una aproximación probabilística que tiene en cuenta la incertidumbre inherente en los sensores, los motores y la dinámica del entorno.

En cuanto a los sensores para construir mapas del entorno y detectar obstáculos en movimiento, hay varias tecnologías que se están considerando, como, por ejemplo, las cámaras de muy alta resolución (posiblemente combinadas con nuevas generaciones de faros capaces de iluminar mucho más lejos que ahora sin deslumbrar a otros conductores), los sensores basados en láser (Ford está desarrollando estos sensores en colaboración con la empresa Velodyne, con un alcance de 200 m frente a los 70 de anteriores modelos), y los infrarrojos (muy indicados para detectar personas y animales, actualmente hasta unos 40 m de distancia, quizá insuficiente todavía). Lo que es seguro es que serán necesarios todos estos distintos tipos de sensores.

Por último, en los coches completamente autónomos, otra dificultad de naturaleza no técnica vendrá por aspectos relacionados con la responsabilidad legal. En caso de accidentes, ¿quién será el responsable? La respuesta no es evidente y quizá será necesario equipar los coches con *cajas negras* similares a las de los aviones con el fin de poder dilucidar responsabilidades (¿falló algún sensor?, ¿falló el *software*?, etc.). También será imprescindible reflexionar sobre aspectos éticos. Por ejemplo, en el *software* de un coche completamente autónomo se tendrá que haber previsto qué hacer ante alternativas como, por ejemplo, ¿salvar la vida de los pasajeros o de otras personas?

A la vista de tan numerosos y complejos problemas es lógico preguntarse si realmente algún día tendremos coches completamente autónomos. En cualquier caso, aunque así fuera no es de esperar que a medio plazo los veamos circulando en cualquier tipo de entorno; sí los veremos a corto/medio plazo en zonas controladas, ciertos tramos de autopista, campus universitarios, parques industriales, áreas comerciales, etc. Un ejemplo es la iniciativa británica de que circulen unos 40 pequeños vehículos de dos ocupantes a 8 km/h por los 250 km de calles

peatonales y carriles para bicicletas de la ciudad de Milton Keynes. Estos vehículos harán las funciones de taxi al que se llamará mediante una *app* en el móvil. Pruebas similares se van a llevar a cabo en otras ciudades británicas y en otras partes del mundo como Singapur y Pittsburgh. Además, estamos viendo que ya se van incorporando funcionalidades cada vez más sofisticadas de ayuda a la conducción, como, por ejemplo, mantener automáticamente una distancia de seguridad con el vehículo que va delante, mantener el coche dentro de su carril, etc., con el fin de hacer la conducción humana más segura.

## Lenguaje

Entre los éxitos de la IA en el área de lenguaje destacamos los sistemas Watson, Mastor y Siri, que pasamos a describir a continuación. Estos sistemas, en sus respectivos campos, alcanzan un rendimiento igual o superior al humano, y en algún caso podrían ser tomados por un ser humano real.

### Watson

Watson es un programa desarrollado por IBM que ganó un concurso de respuestas a preguntas de cultura general en febrero de 2011, realizado en la televisión estadounidense. El programa *Jeopardy!* formula preguntas a tres concursantes, que han de responder rápidamente: una vez que el presentador lee la pregunta, el primer concursante que tiene una respuesta aprieta un pulsador y a continuación pronuncia su respuesta. Respuestas incorrectas penalizan, por lo que un concursante ha de tener una cierta confianza de su calidad antes de apretar el pulsador. Las preguntas de *Jeopardy!* son enrevesadas, incluyen pistas y juegos de palabras, y en ocasiones indican el tipo de respuesta. Dada la amplitud de los temas tratados y el formato de preguntas, participar en *Jeopardy!* suponía un desafío muy serio para cualquier

equipo que pretendiera desarrollar un contrincante computacional.

Técnicamente, Watson se encuadra en los *sistemas de respuesta a preguntas en dominios abiertos* (*open-domain question answering*). La arquitectura interna de Watson se basa en la tecnología DeepQA, desarrollada por IBM, que combinaba técnicas de recuperación de información, lenguaje natural, representación del conocimiento y razonamiento, y aprendizaje. Dada la amplitud de los temas tratados en *Jeopardy!*, se tenían que manejar múltiples fuentes de información sin estructura (como la que se encuentra en un texto libre). La idea básica de Watson era disponer de grandes cantidades de contenidos (texto, voz, imágenes) que se anotaban con significado (semántica) en determinadas regiones. Estas anotaciones las realizaban de forma cooperativa una serie de programas denominados anotadores; cada uno realiza tareas relativamente simples que, en conjunto, producían un resultado complejo. Watson era autocontenido, no estaba conectado a Internet. Antes del juego, su memoria fue *alimentada* con enciclopedias, obras de referencia, toda la Wikipedia, etc. Para el juego, toda esta información (en torno a cuatro terabytes) se cargó en una memoria RAM, pues el acceso a disco enlentecía demasiado el sistema.

Una parte importante de DeepQA se dedicó a comprender bien la pregunta, así como el tipo de respuesta esperada y las pistas que contenía. A partir de ahí, DeepQA no buscaba una comprensión absoluta con respecto a un modelo determinado, ontología o base de datos. Los resultados del análisis de la pregunta (que incluía un análisis sintáctico) eran múltiples interpretaciones, que enfrentadas a los contenidos del sistema generaban un amplio conjunto de respuestas candidatas. Cada respuesta candidata junto con la pregunta representaba una hipótesis independiente que se puntuaba, incluyendo indicios adicionales que se basaban en técnicas gramaticales (sintácticas y semánticas) y en técnicas estadísticas de proceso del lenguaje (capítulo 4). Además,

había un tratamiento específico para preguntas que exigían un manejo de datos estructurados.

En ocasiones, las preguntas de *Jeopardy!* eran especiales, en el sentido de que incluían referencias implícitas que se debían resolver para responder con precisión. Otra técnica era la descomposición de una pregunta en subpartes lógicas, que podían ser tratadas de forma independiente y combinarse sus resultados.

Para una pregunta, DeepQA podía reportar 100 respuestas candidatas, cada una de ellas soportada por unos 100 indicios, y cada par indicio-respuesta podía ser puntuado de 100 formas diferentes; para cada candidato había que resolver en torno a 10.000 puntuaciones diferentes. La combinación de estas puntuaciones con sus confianzas respectivas en una única probabilidad se hacía mediante técnicas estadísticas de aprendizaje. La independencia de los análisis era útil: si diferentes algoritmos convergían en la misma respuesta, aumentaba su probabilidad de que fuera correcta. A veces se hacían verificaciones cruzadas en tiempo o espacio para descartar candidatos.

Para conseguir una respuesta rápida (unos 3 segundos de media, a nivel humano), se recurrió al paralelismo masivo (2.880 procesadores). Sobre DeepQA se desarrollaron dos módulos para realizar el juego: un módulo de estrategia (qué preguntas elegir, cuánto apostar en algunas preguntas, cuándo responder y cuándo no, etc.) y otro módulo de interfaz. Watson no ve ni oye, las preguntas se las proporcionan por escrito y un sintetizador de voz pronuncia las respuestas.

En *Jeopardy!*, Watson ganó a brillantes contrincantes humanos elegidos entre jugadores del concurso. Lo hizo de forma clara, con las mismas reglas de juego que los jugadores humanos. Antes, Watson había jugado un conjunto de partidas de ensayo contra contrincantes humanos, que sirvieron para corregir puntos débiles. En el futuro, IBM conjetura la utilización de esta tecnología para tareas que exigen consultar un gran número de textos o referencias, como el

razonamiento médico, el razonamiento legal o servicios *online* de atención al cliente.

### Siri

Siri es un *asistente personal virtual* (*virtual personal assistant*) desarrollado para los iPhones de Apple. Se comunica mediante el habla: escucha las preguntas del usuario y pronuncia sus respuestas. Responde preguntas, hace recomendaciones y realiza acciones mediante los servicios web del aparato (por ejemplo, encontrar un restaurante cercano o comprar unas entradas de cine). Claramente, Siri procesa el lenguaje natural para sus interacciones con el usuario (recibir peticiones y formular respuestas). Además, ofrece una interacción conversacional con otras aplicaciones como los recordatorios, el estado del tiempo, la bolsa, la mensajería, el correo electrónico, el calendario, los contactos, las notas, la música, el reloj, el navegador web y los mapas. Siri también puede devolver llamadas, leer los mensajes del buzón de voz, etc., y dispone de soporte en varios idiomas. Siri no incluye una tecnología radicalmente nueva: su principal contribución es que ha juntado diferentes tecnologías complementarias existentes en un único sistema.

Un elemento clave en Siri es el reconocimiento del habla. Siri tiene dos modos básicos: reconocimiento de comandos y dictado. Cuando Siri pregunta, puede acotar bastante en ámbito de la respuesta con un vocabulario limitado. En esos casos utiliza un sistema basado en gramática para procesar la respuesta. Por el contrario, cuando recibe un dictado del usuario, utiliza un sistema de reconocimiento de voz intentando transcribir cada palabra que oye. Aunque no es necesario una fase de entrenamiento a la voz del hablante, Siri mejora su comprensión a medida que el usuario lo utiliza (cómo pronuncia las palabras, qué palabras son más frecuentes, cómo pronuncia nombres y cifras). La parte cognitiva de Siri ha de ser capaz de *comprender* los elementos de los que se habla (identificar nombres con personas,

duraciones de eventos, etc.). Tanto por su parte cognitiva como la de reconocimiento del lenguaje, Siri ha sido entrenado con miles de datos obtenidos de la web.

Las colecciones de datos que son explotados por Siri son demasiado grandes para los iPhones. Siri utiliza conexiones de alta velocidad para reconocimiento de voz e interroga a otros proveedores de información (consultando fuentes como Bing, Wikipedia y Twitter) sobre preguntas concretas. Siri puede incluso conectarse a *The New York Times*.

### Mastor

Mastor (Multilingual Automatic Speech-To-Speech Tranlator) es un sistema de *traducción automática de voz* desarrollado por IBM, que combina reconocimiento automático del habla, traducción automática del lenguaje natural y síntesis de voz para facilitar conversaciones en tiempo real entre dos hablantes que no tengan un lenguaje común. Este sistema se ha usado en contextos militares y de emergencias.

Mastor se basa también en métodos estadísticos para realizar la traducción. Sobre el reconocimiento automático del habla, se han utilizado modelos acústicos de los idiomas a tratar combinados con modelos del lenguaje. Estos modelos se basan en fonemas y también en grafemas para idiomas como el árabe, que tiene muchas vocales cortas que no aparecen en la forma escrita y que un hablante nativo puede inferir por el contexto. Estos modelos se procesan a partir de modelos estadísticos basados en las pronunciaciones de muchos textos. Sobre los modelos de los lenguajes, se basan en calcular la probabilidad de secuencias de palabras candidatas en la traducción. Se utiliza un modelo *tri-gram* (probabilidad de ocurrencia de tres palabras consecutivas), que se ha construido a partir del análisis de muchos textos. En árabe, el vocabulario puede ser muy extenso, debido a que la raíz de muchas palabras se puede combinar con prefijos y

sufijos, por lo que se emplean técnicas de análisis morfológico para reducir hasta en un 30% el tamaño del vocabulario.

Con respecto a la traducción automática, la aproximación también es estadística, a partir de un corpus anotado entre los dos idiomas. Mastor elige la secuencia traducida que aparece con más probabilidad. En un intento para evitar la fragilidad de los sistemas puramente estadísticos, Mastor incluye en el cálculo de la probabilidad el (o los) conceptos que aparecen en la frase original y en la frase traducida.

## Otras aplicaciones

Por razones de espacio hemos tenido que dejar en el tintero, o mejor dicho en el teclado, multitud de aplicaciones que también podríamos calificar de grandes éxitos de la IA. Sin embargo, en esta sección vamos a comentar algunas áreas de aplicación que consideramos muy relevantes. Entre ellas están los videojuegos, en particular aquellos en los que el jugador compite con agentes inteligentes, los llamados *non player characters* (NPC): personajes animados presentes en el juego que el jugador no controla. Los NPC tienen que desplazarse de un lugar a otro evitando obstáculos y tienen que tomar decisiones acerca de cuál de sus posibles acciones es la más adecuada en función del contexto, es decir, de la posición y acciones de otros NPC y del personaje controlado por el jugador.

Entre las técnicas de IA que usan los NPC tenemos la búsqueda heurística, la planificación, las redes neuronales, los sistemas deductivos basados en reglas y el aprendizaje de tácticas e incluso estrategias para adaptarse a las habilidades de cada jugador. Los entornos simulados con alto nivel de realismo que ofrecen los videojuegos hacen que estos sean una interesante plataforma para investigar en IA y en robótica. Por ejemplo, investigadores de la Universidad de Michigan en Ann-Arbor han entrenado un *software* para la conducción autónoma de coches mediante el juego *Grand*

*Theft Auto*, ya que, al ser tan realista, constituye una excelente simulación del mundo real. Empresas como Google y Uber entrenan su *software* de conducción autónoma haciendo que los coches recorran millones de kilómetros en tráfico real, a la vez que graban imágenes desde el interior de un coche que posteriormente etiquetan indicando dónde empieza y termina la imagen de cada uno de los coches presentes en cada escena. Se trata, pues, de un proceso muy laborioso que requiere muchas horas. Sin embargo, en juegos como *Grand Theft Auto* todo lo que aparece está ya automáticamente preetiquetado, ya que está generado por el *software* del juego.

Un estudio comparando el resultado de entrenar el mismo *software* con imágenes sintéticas de *Grand Theft Auto* y con imágenes reales muestra resultados muy parecidos, aunque en el caso del videojuego se necesitaron muchas más imágenes de entrenamiento, aunque esto no supone ningún problema, ya que se pueden generar medio millón de imágenes sintéticas en unas horas. El problema es que haría falta sintetizar imágenes de ciudades distintas, simulando diferentes condiciones de iluminación y de tráfico para realmente tener un conjunto de entrenamiento suficientemente amplio como para que lo que aprende el *software* sea suficientemente general, evitando el *overfitting* del algoritmo de aprendizaje.

Otra importante área es la robótica *animaloide*, llamada así porque los robots tienen la forma de animales. El ejemplo más popular son los robots AIBO, en forma de perro, desarrollados por SONY. Ha habido multitud de aplicaciones con estos robots; posiblemente la más conocida y exitosa es la aplicación al fútbol robótico. Efectivamente, en los campeonatos de fútbol robótico conocidos como RoboCup, había una categoría dedicada exclusivamente a equipos formados por robots AIBO. La idea de desarrollar robots jugadores de fútbol como plataforma para progresar en IA, y en particular en la integración de la percepción, el razonamiento y la acción, fue sugerida por el Alan Mackworth en

un artículo publicado en 1993. Paralelamente, también en 1993, varios grupos japoneses de investigación en robótica liderados por Hiroaki Kitano pusieron en marcha una liga de fútbol robótico en su país que, muy pronto, se internacionalizó bajo el nombre de RoboCup, ya que otros grupos de investigadores en robótica, principalmente en Estados Unidos, también estaban desarrollando robots futbolistas. El primer campeonato oficial de RoboCup fue un gran éxito, con la participación más de 40 equipos y con miles de seguidores.

Existen varias categorías de ligas en función del tamaño de los robots participantes, además de una liga de robots simulados. Actualmente muchos países tienen sus ligas nacionales. Esta competición ha permitido progresos muy significativos en IA, ya que, tal como sugirió Mackworth, requiere la integración de diversos componentes fundamentales de la inteligencia como son la percepción, el razonamiento, la planificación, la acción, la comunicación y el aprendizaje. Por otra parte, el aprendizaje permite que los robots puedan jugar de forma colaborativa, planificando jugadas relativamente complejas que impliquen pases de balón.

La medicina es otra área de aplicación que históricamente ha sido clave en el desarrollo de la IA. Una de las técnicas de IA más usadas en aplicaciones médicas son los sistemas expertos. Además, hay muchos otros en este campo que se siguen usando regularmente en hospitales y centros médicos en todo el mundo. Uno de ellos es ATHENA, un sistema de ayuda a la toma de decisiones de los médicos a la hora de gestionar pacientes con problemas de hipertensión. Procesa los datos clínicos de cada paciente y, gracias a su base de conocimientos sobre hipertensión, produce una serie de recomendaciones sobre cómo gestionar mejor la atención clínica personalizada. Otra aplicación muy importante es el sistema GIDEON de ayuda al diagnóstico de un total de 337 enfermedades infecciosas específicas de cada uno de los 224 países de su base de datos. Su base de conocimientos cubre 1.147 taxones microbianos y 306 agentes

antibacterianos y vacunas. La información que maneja incluye también más de 20.000 imágenes, gráficos, mapas interactivos, etc. Todo ello permite alcanzar más del 94% de diagnósticos correctos y lo convierten en uno de los sistemas más usados en el ámbito de la medicina.

Más recientemente han irrumpido con fuerza en la medicina, basada en la evidencia, las técnicas de análisis de grandes cantidades de datos. En poco tiempo se ha extendido enormemente su uso y se han conseguido resultados espectaculares. Vamos a comentar brevemente dos casos de éxito. Analizando grandes cantidades de muestras de células mamarias cancerosas, investigadores de la Universidad de Stanford han descubierto tres nuevos indicadores para evaluar con más confianza la probabilidad de que una biopsia de células mamarias pueda resultar positiva. Por otra parte, científicos de la Universidad de Carnegie Mellon, en colaboración con cuatro hospitales de Chicago, han desarrollado un sistema capaz de predecir infartos con cuatro horas de antelación en enfermos ingresados en la UCI, mejorando en más de tres horas los tiempos de predicción de los cardiólogos. Este *software* fue entrenado con datos de 133.000 pacientes, incorporando 72 parámetros presentes en la historia clínica de los enfermos, incluyendo signos vitales, edad, glucemia y recuentos de plaquetas.

También en biología molecular y farmacología se está aplicando con éxito la IA. Por ejemplo, muchos fármacos tienen efectos secundarios inesperados que pueden resultar muy beneficiosos. Un ejemplo es la Viagra, que, aunque inicialmente se desarrolló para controlar la hipertensión, resultó ser muy efectiva para tratar la disfunción eréctil; o la lovastatina, un efectivo tratamiento de la hipercolesterolemia que ha resultado ser un potente antibiótico. Pues bien, en lugar de esperar que estos beneficiosos efectos secundarios se descubran por casualidad, investigadores en farmacología aplican técnicas de IA para predecir qué fármacos ya existentes pueden tener otros usos terapéuticos, y en particular predecir las propiedades antibióticas de dichos fármacos

con el consiguiente ahorro de tiempo que ello supone frente al desarrollo y aprobación de un nuevo fármaco. Creemos que el papel de la IA en el sector de la salud será cada vez más importante.

En otros ámbitos como, por ejemplo, el medioambiente y el ahorro energético (la aplicación de Deep Mind permite que el centro de datos de Google recorte en un 15% su consumo eléctrico), la IA se convertirá en una tecnología imprescindible para afrontar los grandes desafíos a los que se enfrenta la humanidad.

La economía y la sociología también usan cada vez más modelos de IA, en la línea de *simulación basada en agentes*. Un ejemplo es el modelo propuesto por Farmer y Foley en 2009, que permite simular interacciones entre grandes cantidades de agentes y poder predecir los efectos que causaría introducir elementos nuevos en determinados sistemas (como, por ejemplo, los efectos que tendría sobre la movilidad urbana la construcción de un parque o una zona peatonal, o los efectos sobre la economía y la ecología de la construcción de una autovía o un aeropuerto). De esta forma, las decisiones sobre dichas actuaciones se pueden tomar con mucha más y mejor información disponible.

También existen importantes resultados de aplicaciones de la IA en ciencias y, en particular, en matemáticas. Un ejemplo lo tenemos en la demostración de la conjetura de Robbins. En 1934, Herbert Robbins conjeturó que las álgebras de Robbins son álgebras de Boole. Un álgebra de Robbins es un álgebra que contiene un solo operador binario (la unión $\vee$) y un operador unario (la negación $\neg$) y estos operadores satisfacen los axiomas siguientes: asociatividad $(A \vee (B \vee C) = (A \vee B) \vee C)$, conmutatividad $(A \vee B = B \vee A)$ y la ecuación de Robbins $(\neg(\neg (A \vee B) \vee (A \vee \neg B)) = A$. En 1997, William McCune demostró la conjetura usando el demostrador automático de teoremas Equational Prover, que demuestra teoremas de lógica ecuacional, es decir, cálculo de predicados de primer orden cuyo único predicado es la igualdad.

De hecho, la lista de aplicaciones de la IA es interminable, por lo que simplemente vamos a enumerar algunas más sin dar detalles. Los sistemas de inyección de gasolina en nuestros automóviles están diseñados utilizando algoritmos genéticos, así como las turbinas de los aviones a reacción. En el metro de Hong Kong durante la noche 10.000 ingenieros llevan a cabo 2.600 trabajos de mantenimiento, a lo largo de los 218 km y 152 estaciones de la red, programados por un *software* de planificación. La detección automática de transacciones fraudulentas de tarjetas de crédito se realiza mediante algoritmos de aprendizaje automático. El enrutamiento de las llamadas de teléfonos móviles se basa en IA. La detección de hábitos de consumo se basa en el análisis automático de grandes cantidades de datos mediante algoritmos de aprendizaje automático.