O'REILLY®

Aprendiendo Ten

Aprendizaje automático potente en JavaScript

guante de laborde

Prólogo de Laurence Moroney

O'REILLY®

Learning TensorFlow.js

Given the demand for AI and the ubiquity of JavaScript, TensorFlow.js was inevitable. With this Google framework, seasoned AI veterans and web developers alike can help propel the future of AI-driven websites. In this guide, author Gant Laborde—Google Developer Expert in machine learning and the web—provides a hands-on end-to-end approach to TensorFlow.js fundamentals for a broad technical audience that includes data scientists, engineers, web developers, students, and researchers.

You'll begin by working through some basic examples in TensorFlow.js before diving deeper into neural network architectures, DataFrames, TensorFlow Hub, model conversion, transfer learning, and more. Once you finish this book, you'll know how to build and deploy production-ready deep learning systems with TensorFlow.js.

- Explore tensors, the most fundamental structure of machine learning
- Convert data into tensors and back with a real-world example
- Combine AI with the web using TensorFlow.js
- Use resources to convert, train, and manage machine learning data
- Build and train your own training models from scratch

Gant Laborde is a proud New Orleans native and adventurous engineer. He's an owner of Infinite Red as well as a mentor, adjunct professor, published author, and award-winning speaker. As a Google Developer Expert, he covers TensorFlow.js from multiple perspectives to make the concept approachable.

"What Gant has done wi
this book is to cut to the
chase and teach you th
important stuff you nee
to know while keeping y
firmly within the web de
oper role, using JavaSc
and the browser."

—Laurence M
Lead AI Advocate,

"*Learning TensorFlow.js*
enables you to take your
steps with TensorFlow.js
allowing any JavaScrip
developer to gain supe
powers in their next web
application and beyon

—Jason
Senior Developer Relations E
for TensorFlow.js,

"Gant's ability to naviga
explaining the complex
ities of machine learnin
while avoiding the pitfa
complicated mathema
is uncanny, and you'd b
hard-pressed to find a b
ter introduction to data
ence using JavaScript."

—Lee W
Full Stack JavaScript De

JAVASCRIPT

US $49.99 CAN $65.99

ISBN: 978-1-492-09079-3

Twitter: @oreillymedia
facebook.com/oreilly

nsorFlow.js

Lo que Gant ha hecho con este libro es ir al grano y enseñarle las cosas importantes que necesita saber mientras lo mantiene firmemente dentro del rol de desarrollador web,

usando JavaScript y el navegador.

—Laurence Moroney, defensor principal de IA, Google

El aprendizaje automático tiene el potencial de influir en todas las industrias que existen. Este libro le permite dar sus primeros pasos con TensorFlow.js, lo que permite que cualquier desarrollador de JavaScript obtenga superpoderes en su próxima aplicación web y más allá. Este libro es una excelente introducción al aprendizaje automático con TensorFlow.js que también aplica lo que aprende con ejemplos del mundo real que son fáciles de digerir.

—Jason Mayes, ingeniero sénior de relaciones con desarrolladores de TensorFlow.js en Google

La capacidad de Gant para navegar explicando las complejidades del aprendizaje automático mientras evita las trampas de las matemáticas complicadas es asombrosa, y sería difícil encontrar una mejor introducción a la ciencia de datos usando JavaScript.

—Lee Warrick, desarrollador completo de JavaScript

Estoy encantado de haber leído Learning TensorFlow.js. Sin duda, es una buena manera de salir de mi zona de confort de la ingeniería de back-end y probar la creación de algunas aplicaciones de front-end emocionantes, aprovechando el poder del contenido del libro sobre Tensorflow.js como el marco de referencia para las aplicaciones web de ML.

—Laura Uzcátegui, ingeniera de software, Microsoft

Este libro sirve como la introducción correcta a la construcción de pequeños modelos de aprendizaje profundo aptos para aplicaciones web y móviles. Los ejemplos del libro junto con la explicación detallada harán que tu aprendizaje sea fluido y divertido.

—Vishwesh Ravi Shrimali, ingeniero, Mercedes Benz R&D India

¡Ojalá hubiera tenido este libro para aprender redes neuronales y TensorFlow.js en el pasado! Sorprendentemente simple y bellamente escrito, va de cero a hacer un proyecto final completo en 12 capítulos cortos. Imprescindible en toda biblioteca.

—Axel Sirota, ingeniero de investigación de aprendizaje automático

Esta es una introducción muy necesaria a TensorFlow.js, con excelentes ejemplos, ilustraciones sorprendentes y citas interesantes al comienzo de cada capítulo. Una lectura obligada para cualquiera que se tome en serio la creación de IA con JavaScript.

—Alexey Grigorev, Fundador de DataTalks.Club

El aprendizaje automático para la web aún está en pañales, y libros como el que tiene en sus manos en este momento son muy importantes. Como ingeniero de aprendizaje automático que trabaja en herramientas de ML para el entorno de JavaScript, mi principal recomendación para los desarrolladores web que buscan agregar ML a sus proyectos es definitivamente LearningTensorFlow.js.

—Rising Odegua, cocreador de Danfo.js


O'REILLY

Aprendiendo TensorFlow.js

Aprendizaje automático potente en JavaScript

guante de laborde

Pekín Boston Farnham Sebastopol Tokio

Historial de revisiones de la primera edición

2021-05-07: Primer lanzamiento

Ver

http://oreilly.com/catalog/errata.csp?isbn=9781492090793 para detalles de lanzamiento.

978-1-492-09079-3

[LSI]

Este libro es
dedicado

a la sonrisa más contagiosa y gentil.

A la chispa incontenible y sin fin

alegría de mi corazón. a mi amor

hija, te amo,

Miles.

Machine Translated by Google

Tabla de contenido

Machine Translated by Google

Modelo fuera de la red Modelo de compras Zoológicos Conversión de modelos Su primer modelo personalizado Conozca la máquina enseñable Use la máquina enseñable Recopilación de datos y capacitación Verificación del modelo Aprendizaje automático Gotchas Pequeñas cantidades de datos Datos

La IA y el aprendizaje automático son tecnologías revolucionarias que pueden cambiar el mundo, pero solo pueden hacerlo si hay desarrolladores que utilicen buenas API para aprovechar los avances que estas tecnologías traen.

Uno de esos avances es la capacidad de ejecutar modelos de aprendizaje automático en el navegador, lo que permite que las aplicaciones actúen de manera inteligente.

El auge de TensorFlow.js me dice que la IA ha llegado. Ya no está exclusivamente en el ámbito de los

científicos de datos con supercomputadoras; ahora es accesible para los millones de desarrolladores que codifican en JavaScript diariamente. Pero hay una brecha. Las herramientas y técnicas para construir modelos todavía están en manos de aquellos que conocen los misterios de Python, NumPy, unidades de procesamiento de gráficos (GPU), ciencia de datos, modelado de funciones, aprendizaje supervisado, tensores y muchos más términos extraños y maravillosos. que probablemente no conozcas!

Lo que Gant ha hecho con este libro es ir al grano, enseñándole las cosas importantes que necesita saber mientras lo mantiene firmemente dentro del rol de desarrollador web, usando JavaScript y el navegador. Le presentará los conceptos de IA y aprendizaje automático con un enfoque claro sobre cómo se pueden usar en la plataforma que le interesa.

A menudo, escucho a los desarrolladores preguntar, cuando quieren usar el aprendizaje automático: “¿Dónde puedo encontrar cosas que pueda reutilizar? ¡No quiero aprender a ser un ingeniero de ML solo para descubrir si esto funcionará para mí!”

Gant responde a esa pregunta en este libro. Descubrirá modelos prefabricados que puede tomar de TensorFlow Hub para bloqueo, stock y barril. También aprenderá a pararse sobre los hombros de gigantes tomando porciones seleccionadas de modelos creados con millones de elementos de datos y miles de horas de capacitación, y verá cómo puede transferir lo aprendido de ellos a su propio modelo. Luego, ¡simplemente colóquelo en su página y deje que Javaÿ Script haga el resto!

Los desarrolladores preguntan: "¿Cómo puedo usar el aprendizaje automático en la plataforma que me interesa sin una amplia capacitación?"

Este libro profundiza en eso, mostrándole cómo cerrar la brecha entre Javaÿ Script y los modelos que fueron entrenados usando TensorFlow. Desde la conversión de datos entre primitivas y tensores hasta el análisis de probabilidades de salida en texto, este libro lo guía a través de los pasos para integrarse estrechamente con su sitio.

Los desarrolladores me preguntan: “Quiero ir más allá del trabajo de otras personas y los prototipos simples. ¿Puedo hacer eso como desarrollador web?”

De nuevo, sí. Cuando haya terminado este libro, no solo estará familiarizado con el uso de modelos, sino que Gant le dará todos los detalles que necesita para crearlos usted mismo.
Aprenderá a entrenar modelos complejos, como redes neuronales convolucionales, para reconocer el contenido de las imágenes, y lo hará todo en JavaScript.

Una encuesta realizada en octubre de 2020 mostró que había 12,4 millones de desarrolladores de JavaScript en el mundo. Otras encuestas mostraron que hay alrededor de 300.000 profesionales de IA en todo el mundo. Con la tecnología de TensorFlow.js y las habilidades de este libro, usted, querido desarrollador de JavaScript, puede ser parte de hacer que la IA sea importante. Y este libro es un maravilloso, maravilloso lugar para comenzar.

¡Disfruta el viaje!

—Laurence Moroney
marzo 2021

Prefacio

“Si eliges no decidir, todavía has tomado una decisión”.
—Geddy Lee (Rush)

La retrospectiva es siempre 20/20. "Debería haber comprado algo de bitcoin cuando estaba en X" o "Si tan solo hubiera aplicado en el inicio Y antes de que se hicieran famosos". El mundo está repleto de momentos que nos definen para bien o para mal. El tiempo nunca retrocede, pero hace eco de las lecciones de nuestras elecciones más jóvenes a medida que avanzamos. Tienes la suerte de tener este libro y este momento para decidir.

Los cimientos de la industria del software están cambiando gracias a la inteligencia artificial. Los cambios serán decididos en última instancia por aquellos que toman el control y dan forma al mundo en lo que será mañana. El aprendizaje automático es una aventura hacia nuevas posibilidades, y cuando se unifica con la amplia exposición de JavaScript, los límites desaparecen.

Como me gusta decirle a mi audiencia en mis charlas sobre IA, "No llegaron tan lejos en la creación de software solo para llegar tan lejos". Así que comencemos y veamos a dónde nos lleva nuestra imaginación.

¿Por qué TensorFlow.js?

TensorFlow es uno de los marcos de aprendizaje automático más populares del mercado. Cuenta con el respaldo de las mentes más importantes de Google y es responsable de impulsar a muchas de las empresas más influyentes del mundo. TensorFlow.js es el marco JavaScript indomable de TensorFlow y es mejor que todos los competidores. En resumen, si desea el poder de un marco en

JavaScript, solo hay una opción que puede hacerlo todo.

¿Quién debería leer este libro?

Dos grupos demográficos principales disfrutarán y se beneficiarán del contenido de este libro:

El desarrollador de
JavaScript Si está familiarizado con JavaScript, pero nunca antes ha tocado el aprendizaje automático, este libro será su guía. Se apoya en el marco para mantenerlo activo en creaciones pragmáticas y emocionantes. Comprenderá los conceptos básicos del aprendizaje automático con experiencia práctica a través de la construcción de todo tipo de proyectos. Si bien no nos alejaremos de las matemáticas o conceptos más profundos, tampoco complicaremos demasiado la experiencia con ellos. Lea este libro si está creando sitios web en JavaScript y desea obtener un nuevo superpoder.

El especialista en
IA Si está familiarizado con TensorFlow o incluso con los principios fundamentales del álgebra lineal, este libro le proporcionará innumerables ejemplos de cómo llevar sus habilidades a JavaScript. Aquí encontrará varios conceptos básicos ilustrados, mostrados y representados en el marco TensorFlow.js. Esto le permitirá aplicar su vasto conocimiento a un medio que puede existir de manera eficiente en dispositivos periféricos como navegadores de clientes o Internet de las cosas (IoT). Lea este libro y aprenda cómo llevar sus creaciones a innumerables dispositivos con ricas experiencias interactivas.

Este libro requiere una cantidad moderada de comodidad para leer y comprender JavaScript moderno.

Descripción general del libro

Al esbozar este libro, me di cuenta de que tendría que tomar una decisión. O podría crear una aventura vertiginosa en una variedad de aplicaciones de aprendizaje automático y tocar cada una con ejemplos pequeños y tangibles, o podría elegir un camino único que cuente una historia cada vez mayor de los conceptos. Después de encuestar a mis amigos y seguidores, estaba claro que se necesitaba este último. Para mantener este libro cuerdo y de menos de mil páginas, opté por eliminar cualquier marco de JavaScript y centrarme en un singular viaje pragmático hacia los aspectos visuales de la IA.

Cada capítulo termina con preguntas y un desafío particular para que pongas a prueba tu determinación. Las secciones de Chapter Challenge se han construido cuidadosamente para solidificar las lecciones en su memoria muscular TensorFlow.js.

Los capítulos

Los capítulos 1 y 2 comienzan con conceptos básicos y un ejemplo concreto. Este enfoque de yin y yang refleja el estilo de enseñanza del libro. Cada capítulo se basa en las lecciones, el vocabulario y las funciones mencionadas en los capítulos anteriores.

Los capítulos 3 a 7 le brindan la visión para comprender e implementar las herramientas y los datos de IA existentes. Podrá crear bibliotecas impresionantes y emplear modelos en proyectos que fueron creados por decenas de científicos de datos.

Los capítulos 8 a 11 comienzan a brindarle el poder de creación en TensorFlow.js. Podrá entrenar modelos en JavaScript y creo firmemente que esta es una de las secciones más divertidas y emocionantes de todo el libro.

El capítulo 12 es el desafío final. El capítulo final presenta un proyecto final que lo empodera para tomar todo lo que el libro tiene para ofrecer y expresarlo usando su propia facultad.

La conclusión

Después de leer este libro, independientemente de su experiencia previa, podrá encontrar, implementar, ajustar y crear modelos de aprendizaje automático en TensorFlow.js. Tendrá la capacidad de identificar una aplicación de aprendizaje automático en un sitio web y luego seguir adelante con el cumplimiento de esa implementación.

Las convenciones usadas en este libro

En este libro se utilizan las siguientes convenciones tipográficas:

Cursiva

Indica nuevos términos, URL, direcciones de correo electrónico, nombres de archivo y extensiones de archivo.

Ancho constante Se

utiliza para listas de programas, así como dentro de párrafos para referirse a elementos de programas como nombres de variables o funciones, bases de datos, tipos de datos, variables de entorno, declaraciones y palabras clave.

Negrita de ancho constante

Muestra comandos u otro texto que el usuario debe escribir literalmente.

Cursiva de ancho constante

Muestra el texto que se debe reemplazar con valores proporcionados por el usuario o por valores determinados por el contexto.

Este elemento significa un consejo o sugerencia.

Uso de ejemplos de código

Los materiales complementarios (ejemplos de código, ejercicios, etc.) están disponibles para descargar en https://github.com/GantMan/learn-t s.

Si tiene una pregunta técnica o un problema al usar los ejemplos de código, envíe un correo electrónico a bookquestions@oreilly.com.

Este libro está aquí para ayudarle a hacer su trabajo. En general, si se ofrece un código de ejemplo con este libro, puede usarlo en sus programas y documentación. No es necesario que se comunique con nosotros para obtener permiso a menos que esté reproduciendo una parte importante del código. Por ejemplo, escribir un programa que use varios fragmentos de código de este libro no requiere permiso. Vender o distribuir ejemplos de libros de O'Reilly requiere permiso. Responder una pregunta citando este libro y citando código de ejemplo no requiere permiso. La incorporación de una cantidad significativa de código de ejemplo de este libro en la documentación de su producto requiere permiso.

Apreciamos, pero generalmente no requerimos, atribución. Una atribución suele incluir el título, el autor, el editor y el ISBN. Por ejemplo: “Aprender TensorFlow.js por Gant Laborde (O'Reilly). Copyright 2021 Gant Laborde, 978-1-492-09079-3.”

Si cree que su uso de los ejemplos de código está fuera del uso justo o del permiso otorgado anteriormente, no dude en contactarnos en permisos@oreilly.com.

Aprendizaje en línea de O'Reilly

Durante más de 40 años, O'Reilly Media ha brindado capacitación, conocimientos y perspectivas en tecnología y negocios para ayudar a las empresas a tener éxito.

Nuestra red única de expertos e innovadores comparte su conocimiento y experiencia a través de libros, artículos y nuestra plataforma de aprendizaje en línea. La plataforma de aprendizaje en línea de O'Reilly le brinda acceso a pedido a cursos de capacitación en vivo, rutas de aprendizaje en profundidad, entornos de codificación interactivos y una amplia colección de texto y video de O'Reilly y más de 200 editores más. Para obtener más información, visite http://oreilly.com.

Cómo contactarnos

Dirija sus comentarios y preguntas sobre este libro a la editorial:

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472 800-998-9938 (en
los Estados Unidos o Canadá) 707-829-0515 (internacional
o local) 707-829-0104 (fax)

Tenemos una página web para este libro, donde enumeramos las erratas, ejemplos y cualquier información adicional. Puede acceder a esta página en https://oreil.ly/learning-tensorflow-js.

Envíe un correo electrónico a bookquestions@oreilly.com para comentar o hacer preguntas técnicas sobre este libro.

Para noticias e información sobre nuestros libros y cursos, visite http://oreilly.com.

Encuéntranos en Facebook: http://facebook.com/oreilly

Síganos en Twitter: http://twitter.com/oreillymedia

Míranos en YouTube: http://youtube.com/oreillymedia

Expresiones de gratitud

Quiero dar las gracias a los editores, el personal de producción y los empleados de O'Reilly, con quienes ha sido un placer trabajar mientras escribía este libro.

Por supuesto, gracias al preeminente Laurence Moroney, el autor del prólogo de este libro. Has sido un ícono y una inspiración en mi crecimiento. He aprendido mucho de usted y seguiré prosperando a raíz de sus lecciones y logros de aprendizaje automático.

Los revisores técnicos de este libro fueron algunas de las personas más amigables y minuciosas con las que he tenido el placer de trabajar.

Laura Uzcátegui

Tus amables e inspiradores comentarios fueron encantadores de leer. Tu intuición ayudó a que este libro se sintiera bien.

Vishwesh Ravi Shrimali

Captaste todos mis chistes. Eres obviamente una persona inteligente y amigable, y de esa manera, me empujaste a ser mejor. Aprecio todos sus consejos y sabiduría.

Jason Mayes,

te conocí antes de que fueras seleccionado como editor técnico de este libro, así que te considero un amigo tanto como un compañero de equipo para hacer que este libro sea lo mejor posible. Sus comentarios fueron meticulosos, inteligentes e insustituibles. SInceramente Gracias.

Axel Damian Sirota

Puedo sentir el gran apoyo personal en cada comentario que me diste. Te tomaste el tiempo para ser inteligente y amable. Su arte fue un regalo para este libro.

Lee Warrick

Continúas revisándome y desafiándome a ser mejor en todos mis trabajos. Este libro es el último ejemplo de mis logros que prosperan gracias a su conocimiento.

michele cronin

Disfruté cada reunión que tuvimos. ¡Eres un deleite sostenido! Gracias por ser como eres. Hiciste este libro sin esfuerzo. Cómo lo hiciste, nunca lo sabré.

Un agradecimiento especial a Frank von Hoven III, quien hizo todas las preguntas correctas y me entregó páginas y páginas de notas de comentarios escritas a mano (Figura P-1). Su estilo y honestidad me guiaron para entregar un mensaje fuerte e inspirador.

Figura P-1. Impresionante apoyo y comentarios de Frank

Por último, gracias a mi amada familia, que evitó interrumpirme tanto como fue posible pero también sabía cuándo necesitaba una buena interrupción. Alicia, mi amor, tú me conoces mejor que yo mismo. Me sorprendiste con un café vigorizante cuando necesitaba seguir adelante y con un trago fuerte cuando era hora de parar y descansar. Cada día, lo mejor de mí es gracias a ti.

Machine Translated by Google

CAPÍTULO 1

La IA es mágica

"Cualquier tecnología suficientemente avanzada es indistinguible de la magia."
—Arthur C. Clarke

OK, la IA no es magia real. La verdad es que la IA está un paso por encima y más allá de otras tecnologías hasta el punto de que se siente como magia. Es bastante fácil explicar un algoritmo de clasificación efectivo, pero profundizar en la inteligencia toca el tercer riel y nos lleva a todos a un nivel completamente nuevo de poder tecnológico. Ese poder exponencial es posible gracias a la sabiduría y la aptitud de TensorFlow.js.

Durante más de medio siglo, los científicos e ingenieros han recreado la rueda metafórica en la IA y han perfeccionado la mecánica que la controla. Cuando profundicemos en la IA en este libro, comprenderemos esos conceptos con el marco flexible pero duradero de TensorFlow.js y, con él, llevaremos nuestras ideas a buen término en el dominio en expansión de JavaScript. Sí, JavaScript, el lenguaje de programación más popular del mundo.1 Los conceptos y definiciones

proporcionados aquí lo equiparán con una visión de rayos tecnológicos. Cortará acrónimos y palabras de moda para ver y comprender la infraestructura de IA que ha estado surgiendo en todos los campos que nos rodean. Los conceptos de inteligencia artificial y aprendizaje automático se aclararán, y las definiciones de este capítulo pueden servir como referencia para identificar los principios básicos que impulsarán nuestro despegue académico hacia la iluminación de TensorFlow.js.

1 Estadísticas del lenguaje de programación: https://octoverse.github.com

Lo haremos:

- Aclarar el dominio de la IA y la inteligencia •
- Discutir los tipos de aprendizaje automático •
- Revisar y definir terminología común • Examinar
- conceptos a través de la lente de TensorFlow.js

¡Vamos a empezar!

> Si ya está familiarizado con TensorFlow.js y la terminología, la filosofía y las aplicaciones fundamentales del aprendizaje automático, puede pasar directamente al Capítulo 2.

El camino de la IA en JavaScript

TensorFlow.js, para ponerlo en la definición más mundana posible, es un marco para manejar conceptos específicos de IA en JavaScript. Eso es todo. Afortunadamente para ti, estás en el libro correcto en el momento correcto de la historia. La revolución industrial de la IA acaba de comenzar.

Cuando surgieron las computadoras, una persona armada con una computadora podía realizar tareas casi imposibles a una escala significativa. Podían descifrar códigos, recordar instantáneamente información de montañas de datos e incluso jugar juegos como si estuvieran jugando con otro ser humano. Lo que era imposible de hacer para una persona se volvió no solo posible sino habitual. Desde los albores de ese invento digital clave, nuestro objetivo ha sido empoderar a las computadoras para que simplemente "hagan más". Como seres humanos singulares, somos capaces de cualquier cosa, pero no de todo. Las computadoras ampliaron nuestras limitaciones en formas que nos dieron a todos un nuevo poder. Muchos de nosotros nos pasamos la vida perfeccionando algunas habilidades y, entre ellas, aún menos se convierten en nuestras especialidades. Todos construimos los logros de toda una vida, y algunos de nosotros llegamos a ser los mejores del mundo en una cosa, una habilidad que solo se puede ganar con suerte, información y miles de días ahora.

AI nos permite saltar al frente de la fila; para construir audazmente lo que nunca se ha construido antes. Diariamente vemos empresas e investigadores dar ese salto computacional una y otra vez. Estamos parados en la entrada de una nueva industria que nos invita a participar en cómo cambiará el mundo.

Estás en el asiento del conductor, te diriges a la próxima gran cosa, y este libro es tu volante. Nuestro viaje en el aprendizaje de la magia de la IA estará limitado solo por el alcance de su imaginación. Armado con el aprendizaje automático habilitado para JavaScript, puede

¡acceda con cámaras, micrófonos, actualizaciones instantáneas, ubicaciones y otros sensores, servicios y dispositivos físicos!

Estoy seguro de que te estarás preguntando: “¿Pero por qué la IA no ha estado haciendo esto antes? ¿Por qué es esto importante ahora?” Para apreciar eso, deberá hacer un viaje a la búsqueda de inteligencia reproducida por la humanidad.

¿Qué es la inteligencia?

Se pueden escribir libros sobre libros sobre los conceptos del pensamiento y especialmente sobre el camino hacia la inteligencia artificial. Y como ocurre con todos los esfuerzos filosóficos, cada declaración concreta sobre la inteligencia se puede argumentar en el camino. No es necesario saberlo todo con certeza, pero debemos comprender el dominio de la IA para que podamos entender cómo aterrizamos en un libro sobre TensorFlow.js.

Poetas y matemáticos a lo largo de los siglos afirmaron que el pensamiento humano no era más que la combinación de conceptos preexistentes. La apariencia de vida se consideraba una maquinación de diseño divino; todos estamos simplemente "hechos" de los elementos. Las historias de la mitología griega tenían al dios de la invención, Hefesto, creando robots de bronce automatizados que caminaban y actuaban como soldados. Básicamente, estos fueron los primeros robots. El concepto de robots e inteligencia se ha arraigado en nuestra base como un arte último y divino de esta antigua tradición. talos, el gigantesco guerrero animado, fue programado para proteger la isla de Creta. Si bien no había un robot de bronce real, la historia sirvió como combustible para la aspiración mecánica. Durante cientos de años, la antigüedad animatrónica siempre se consideró un camino hacia lo que parecería ser una "inteligencia" humana, y siglos después, comenzamos a ver que la vida imita al arte. De niño, recuerdo ir a mi Chuck E. Cheese local, un restaurante popular en los Estados Unidos con actuaciones musicales animatrónicas para niños. Recuerdo haber creído, solo por un momento, que el concierto eléctrico impulsado por marionetas que se presentaba todos los días era real. Me inspiró la misma chispa que impulsa a los científicos a perseguir la inteligencia. Esta chispa siempre ha estado ahí, pasó por las historias, el entretenimiento y ahora la ciencia.

A medida que los conceptos de máquinas que pueden funcionar de manera autónoma e inteligente crecieron a lo largo de la historia, nos esforzamos por definir estas entidades conceptuales. Los académicos continuaron investigando la inferencia y el aprendizaje con trabajos publicados, mientras mantuvieron su terminología en el ámbito de "máquina" y "robot". La imitación de la inteligencia de la maquinaria siempre se vio frenada por la falta de velocidad y electricidad.

El concepto de inteligencia permaneció fijo en la mente humana y fuera del alcance de las estructuras mecánicas durante cientos de años, hasta la creación de la máquina definitiva, las computadoras. La informática nació como la mayoría de las máquinas, con un solo propósito para todo el dispositivo. Con el advenimiento de la computación, surgió un nuevo término para ilustrar un avance creciente en la inteligencia que refleja significativamente la inteligencia humana.

intelecto. El término AI significa inteligencia artificial y no se acuñó hasta la década de 1950.2 A medida que las computadoras crecieron para convertirse en un propósito general, las filosofías y disciplinas comenzaron a combinarse. El concepto de imitar la inteligencia saltó de la mitología a un campo de estudio científico. Cada dispositivo de medición electrónico para la humanidad se convirtió en un nuevo órgano sensorial para las computadoras y una oportunidad emocionante para la ciencia electrónica e inteligente.

En un tiempo relativamente corto, tenemos computadoras interactuando con humanos y emulando acciones humanas. La imitación de la actividad similar a la humana proporcionó una forma de lo que estamos dispuestos a llamar "inteligencia". La inteligencia artificial es ese término general para estas acciones estratégicas, independientemente del nivel de sofisticación o técnica. Una computadora que puede jugar tres en raya no tiene que ganar para ser categorizada como IA. AI es una barra baja y no debe confundirse con la inteligencia general de una persona. El fragmento más pequeño de código simplista puede ser IA legítima, y el levantamiento apocalíptico de máquinas inteligentes de Hollywood también es IA.

Cuando se usa el término IA, es un término general para la inteligencia que proviene de un dispositivo inerte y generalmente no biológico. Independientemente del umbral mínimo para el término, la humanidad, armada con un campo de estudio y un uso práctico cada vez mayor, tiene un término unificador y un objetivo directo. Todo lo que se mide se gestiona, por lo que la humanidad comenzó a medir, mejorar y competir hacia una mayor IA.

La historia de la IA

Los marcos para IA comenzaron siendo terriblemente específicos, pero ese no es el caso hoy. Como puede que sepa o no, los conceptos de TensorFlow.js como marco pueden aplicarse a música, videos, imágenes, estadísticas y cualquier dato que podamos acumular. Pero no siempre fue así. Las implementaciones de IA comenzaron como un código específico de dominio que carecía de cualquier tipo de capacidad dinámica.

Hay algunos chistes que circulan por Internet de que la IA es solo una colección de declaraciones IF/THEN y, en mi opinión, no están 100% equivocados. Como ya mencionamos, IA es un término general para todo tipo de imitación de la inteligencia natural. Incluso a los programadores principiantes se les enseña a programar resolviendo IA simple en ejercicios como Ruby Warrior. Estos ejercicios de programación enseñan los fundamentos de los algoritmos y requieren relativamente poco código. El costo de esta simplicidad es que, si bien sigue siendo IA, está bloqueada imitando la inteligencia de un programador.

Durante mucho tiempo, el método destacado de promulgar la IA dependía de que las habilidades y filosofías de una persona que programa una IA se tradujeran directamente en código para que una computadora pudiera ejecutar las instrucciones. La lógica digital está realizando la

2 La inteligencia artificial fue acuñada por John McCarthy en 1956 en la primera conferencia académica sobre el tema.

lógica humana de las personas que se comunicaron para hacer el programa. Este es, por supuesto, el mayor retraso en la creación de IA. Necesita una persona que sepa cómo hacer una máquina que sepa, y estamos limitados por su comprensión y capacidad para traducir esa comprensión. Las IA que están codificadas no pueden inferir más allá de sus instrucciones. Este es probablemente el mayor obstáculo para traducir cualquier inteligencia humana en inteligencia artificial. Si está buscando enseñarle a una máquina a jugar al ajedrez, ¿cómo le enseña la teoría del ajedrez? Si quiere decirle a un programa la diferencia entre gatos y perros, algo que es trivial para los niños pequeños, ¿sabría por dónde empezar con un algoritmo?

A finales de los años 50 y principios de los 60, la idea del maestro pasó de los humanos a los algoritmos que podían leer datos sin procesar. Arthur Samuel acuñó el término aprendizaje automático (ML) en un evento que desquició a la IA de las limitaciones prácticas de los creadores. Un programa podría crecer para adaptarse a los datos y comprender conceptos que los programadores de ese programa no pudieron traducir a código o que ellos mismos nunca entendieron.

El concepto de usar datos para entrenar la aplicación o función de un programa era una ambición emocionante. Aún así, en una era en la que las computadoras requerían habitaciones enteras y los datos eran cualquier cosa menos digitales, también era una solicitud insuperable. Pasaron décadas antes de que las computadoras alcanzaran el punto de inflexión crítico para emular las capacidades de información y arquitectura similares a las humanas.

En la década de 2000, los investigadores de ML comenzaron a usar la unidad de procesamiento de gráficos (GPU) para sortear el "canal solitario entre la CPU y la memoria", conocido como el cuello de botella de von Neumann. En 2006, Geoffrey Hinton et al. aprovechó los datos y las redes neuronales (un concepto que cubriremos en nuestra próxima sección) para comprender los patrones y hacer que una computadora lea los dígitos escritos a mano. Esta fue una hazaña que anteriormente era demasiado volátil e imperfecta para la informática común. El aprendizaje profundo fue capaz de leer y adaptarse a la aleatoriedad de la escritura a mano para identificar correctamente los caracteres a un nivel avanzado de más del 98 %. En estos artículos publicados, la idea de los datos como agente de formación salta de los trabajos académicos publicados a la realidad.

Mientras Hinton estaba atascado en la elaboración de una prueba académica básica de que las redes neuronales funcionan, el "¿Qué número es este?" problema se convirtió en un problema incondicional para los profesionales del aprendizaje automático. El problema se ha convertido en uno de los ejemplos triviales clave para los marcos de trabajo de aprendizaje automático. TensorFlow.js tiene una demostración que soluciona este problema directamente en tu navegador en menos de dos minutos. Con los beneficios de TensorFlow.js, podemos construir fácilmente algoritmos de aprendizaje avanzados que funcionan sin problemas en sitios web, servidores y dispositivos. Pero, ¿qué están haciendo realmente estos marcos?

El mayor objetivo de la IA siempre fue acercarse o incluso superar las capacidades humanas en una sola tarea, y el 98 % de precisión en la escritura a mano logró exactamente eso. La investigación de Hinton encendió un enfoque en estos métodos productivos de aprendizaje automático y acuñó términos de la industria como redes neuronales profundas. Explicaremos por qué en la siguiente sección,

pero este fue el comienzo del aprendizaje automático aplicado, que comenzó a florecer y finalmente encontró su camino en marcos de aprendizaje automático como TensorFlow.js. Si bien se crean nuevos algoritmos de aprendizaje basados en máquinas a diestro y siniestro, una fuente de inspiración y terminología se vuelve bastante clara. Podemos emular nuestros sistemas biológicos internos para crear algo avanzado. Históricamente, nos usamos a nosotros mismos y a nuestra corteza cerebral (una capa de nuestro cerebro) como la musa para el entrenamiento estructurado y la inteligencia.

La red neuronal

La idea de las redes neuronales profundas siempre se inspiró en nuestros cuerpos humanos. Los nodos digitales (a veces llamados red de perceptrones) simulan neuronas en nuestros propios cerebros y se activan como nuestras propias sinapsis para crear un mecanismo equilibrado de pensamiento. Es por eso que una red neuronal se llama neuronal, porque emula la estructura bioquímica de nuestro cerebro. Muchos científicos de datos aborrecen la analogía con el cerebro humano, pero a menudo encaja. Al conectar millones de nodos, podemos construir redes neuronales profundas que son elegantes máquinas digitales para tomar decisiones.

Al aumentar las vías neuronales con más y más capas, llegamos al término aprendizaje profundo. El aprendizaje profundo es una conexión muy estratificada (o profunda) de capas ocultas de nodos. Escuchará estos nodos llamados neuronas, neuronas artificiales, unidades e incluso perceptrones. La diversidad de terminología es un testimonio de la amplia gama de científicos que han contribuido al aprendizaje automático.

Todo este campo de aprendizaje es solo una parte de la IA. Si ha estado siguiendo, la inteligencia artificial tiene un subconjunto o rama que se llama aprendizaje automático, y dentro de ese conjunto, tenemos la idea de aprendizaje profundo. El aprendizaje profundo es principalmente una clase de algoritmos que alimentan el aprendizaje automático, pero no es el único. Vea la Figura 1-1 para una representación visual de estos términos primarios.

Figura 1-1. subdominios de IA

Al igual que un ser humano, se utiliza una iteración de enseñanza o "entrenamiento" para equilibrar y desarrollar adecuadamente las neuronas en función de ejemplos y datos. Al principio, estas redes neuronales a menudo son incorrectas y aleatorias, pero a medida que ven un ejemplo tras otro de datos, su poder predictivo “aprende”.

Pero nuestros cerebros no perciben el mundo directamente. Al igual que una computadora, dependemos de señales eléctricas que han sido organizadas en datos coherentes para ser enviadas a nuestro cerebro.
Para las computadoras, estas señales eléctricas son análogas a un tensor, pero lo cubriremos un poco más en el Capítulo 3. TensorFlow.js incorpora todos estos avances que la investigación y los científicos han confirmado. Todas estas técnicas que ayudan a los cuerpos humanos a funcionar pueden integrarse en un marco optimizado para que podamos aprovechar décadas de investigación que se han inspirado en el cuerpo humano.

Por ejemplo, nuestro sistema visual, que comienza en nuestras retinas, usa ganglios para transmitir información fotorreceptora a nuestro cerebro para activar estas neuronas. Como algunos de ustedes recordarán de la biología de los niños, nos faltan puntos en nuestra visión y, técnicamente, vemos todo al revés. La señal no se envía a nuestro cerebro "tal cual". Este sistema visual tiene tecnología incorporada que aprovechamos en el software actual.

Si bien todos estamos emocionados de obtener nuestro televisor con resolución 8K, es posible que crea que nuestro cerebro y nuestra visión aún están más allá de las capacidades informáticas modernas, pero ese no es siempre el caso. El cable que conecta las señales visuales de nuestros ojos a nuestro cerebro tiene solo unos 10 Mb de ancho de banda. Eso es comparable a las conexiones LAN a principios de la década de 1980. Incluso una conexión de transmisión de banda ancha exige más ancho de banda que eso. Pero percibimos todo de forma instantánea y rápida, ¿no? Entonces, ¿cuál es el truco? ¿Cómo estamos obteniendo señales superiores sobre este hardware superado? La respuesta es que nuestras retinas comÿ

presione y "destaque" los datos antes de enviarlos a nuestra red neuronal profundamente conectada. Así que eso es lo que empezamos a hacer con las computadoras.

Las redes neuronales convolucionales (CNN) funcionan con datos visuales de la misma manera que nuestros ojos y cerebros trabajan juntos para comprimir y activar nuestras vías neuronales. Comprenderá mejor y escribirá su propia CNN en el Capítulo 10. Estamos aprendiendo más sobre cómo trabajamos todos los días y estamos aplicando esos millones de años de evolución directamente a nuestro software. Si bien es genial que entiendas cómo funcionan estas CNN, es demasiado académico escribirlas nosotros mismos. TensorFlow.js viene con las capas convolucionales que necesitará para procesar imágenes. Este es el beneficio fundamental de aprovechar un marco de aprendizaje automático.

Puede pasar años leyendo e investigando todos los trucos y trucos únicos que hacen que la visión por computadora, las redes neuronales y los seres humanos funcionen de manera efectiva. Pero estamos en una era en la que estas raíces han tenido tiempo de crecer, ramificarse y finalmente producir frutos: estos conceptos avanzados son accesibles e integrados en los servicios y dispositivos que nos rodean.

La IA de hoy

Hoy usamos estas mejores prácticas con IA para potenciar el aprendizaje automático. Las convulsiones para la detección de bordes, la atención a algunas regiones más que a otras, e incluso las entradas de múltiples cámaras para un elemento singular nos han brindado una mina de oro masticada de datos sobre una granja de servidores de fibra óptica de máquinas en la nube que entrenan IA.

En 2015, los algoritmos de IA comenzaron a superar a los humanos en algunas tareas visuales. Como habrá escuchado en las noticias, la IA ha superado a los humanos en la detección del cáncer . e incluso superó a los mejores abogados de EE. UU. en la identificación de fallas legales. Como siempre con la información digital, AI ha hecho esto en segundos, no en horas. La "magia" de la IA es impresionante.

La gente ha estado encontrando formas nuevas e interesantes de aplicar IA a sus proyectos e incluso crear industrias completamente nuevas.

La IA se ha aplicado a:

• Generar nuevo contenido en escritura, música y visuales • Recomendar contenido útil • Reemplazar modelos estadísticos simples • Deducir leyes a partir de datos • Visualizar clasificadores e identificadores

Todos estos avances han sido aspectos del aprendizaje profundo. Hoy contamos con el hardware, el software y los datos necesarios para permitir cambios innovadores con profunda

Redes de aprendizaje automático. Cada día, la comunidad y las empresas Fortune 500 lanzan nuevos conjuntos de datos, servicios y avances arquitectónicos en el campo de la IA.

Con las herramientas a su disposición y el conocimiento de este libro, puede crear fácilmente cosas que nunca antes había visto y traerlas a la web. Ya sea por placer, ciencia o fortuna, puede crear una solución escalable e inteligente para cualquier problema o negocio del mundo real.

En todo caso, el problema actual con el aprendizaje automático es que es una nueva superpotencia, y el mundo es enorme. No tenemos suficientes ejemplos para comprender todos los beneficios de tener IA en JavaScript. Cuando las baterías lograron una mejora significativa en la vida útil, habilitaron un mundo completamente nuevo de dispositivos, desde teléfonos más potentes hasta cámaras que podían durar meses con una sola carga. Este único avance trajo innumerables productos nuevos al mercado en solo unos pocos años. El aprendizaje automático hace avances constantemente, dejando un torbellino de avances en nuevas tecnologías que ni siquiera podemos aclarar o reconocer porque el diluvio se ha acelerado exponencialmente. Este libro se centrará en ejemplos concretos y abstractos en la medida adecuada, para que pueda aplicar soluciones pragmáticas con TensorFlow.js.

¿Por qué TensorFlow.js?

Tienes opciones. Puede escribir su propio modelo de aprendizaje automático desde cero o elegir entre cualquier marco existente en una variedad de lenguajes de programación. Incluso en el ámbito de JavaScript, ya existen marcos, ejemplos y opciones que compiten entre sí. ¿Qué hace que TensorFlow.js sea capaz de manejar y transportar la IA actual?

Soporte signi cativo

Google crea y mantiene TensorFlow.js. Cubriremos más sobre esto en el Capítulo 2, pero vale la pena señalar que algunos de los mejores desarrolladores del mundo se han unido para hacer realidad TensorFlow.js. Esto también significa que, sin ningún esfuerzo por parte de la comunidad, TensorFlow.js es capaz de trabajar con los últimos y mejores desarrollos innovadores.

A diferencia de otras implementaciones basadas en JavaScript de marcos y bibliotecas de aprendizaje automático, TensorFlow.js admite código acelerado por GPU optimizado y probado. Esta optimización se transmite a usted y a sus proyectos de aprendizaje automático.

Listo para estar en

línea La mayoría de las soluciones de aprendizaje automático se limitan a una máquina extremadamente personalizada. Si desea crear un sitio web para compartir su tecnología innovadora, la IA suele estar bloqueada detrás de una API. Si bien esto es completamente factible con TensorÿFlow.js ejecutándose en Node.js, también es factible con TensorFlow.js ejecutándose directamente en el navegador. Esta experiencia sin instalación es rara en el mundo del aprendizaje automático, lo que le brinda la posibilidad de compartir sus creaciones sin barreras. Puede versionar y acceder a un mundo de interactividad.

O ine Ready Otro

beneficio de JavaScript es que se ejecuta en todas partes. El código se puede guardar en el dispositivo del usuario como una aplicación web progresiva (PWA), Electron o React Native, y luego puede funcionar de manera consistente sin ninguna conexión a Internet. No hace falta decir que esto también proporciona un aumento significativo de la velocidad y los costos en comparación con las soluciones de IA alojadas. En este libro, descubrirá innumerables ejemplos que existen completamente en navegadores que le ahorran a usted y a sus usuarios demoras en la latencia y costos de hospedaje.

Privacidad

AI puede ayudar a los usuarios a identificar enfermedades, anomalías fiscales y otra información personal.

El envío de datos confidenciales a través de Internet puede ser peligroso. Los resultados en el dispositivo permanecen en el dispositivo. Incluso puede entrenar una IA y almacenar los resultados en la máquina del usuario con

cero información nunca dejando la seguridad del navegador.

Diversity

Applied TensorFlow.js tiene un impacto potente y amplio en el dominio y las plataformas de aprendizaje automático. TensorFlow.js puede aprovechar la ejecución de Web Assembly para CPU o GPU para máquinas más robustas. El espectro de IA con aprendizaje automático hoy en día es un mundo significativo y vasto de nueva terminología y complejidad para los recién llegados. Tener un marco que funcione con una variedad de datos es útil ya que mantiene abiertas sus opciones.

El dominio de TensorFlow.js le permite aplicar sus habilidades a una amplia variedad de plataformas que admiten JavaScript (consulte la Figura 1-2).

Figura 1-2. Plataformas TensorFlow.js

Con TensorFlow.js, puede elegir, crear prototipos e implementar sus habilidades en una variedad de campos. Para aprovechar al máximo su libertad de aprendizaje automático, deberá familiarizarse con algunos términos que pueden ayudarlo a iniciarse en el aprendizaje automático.

Tipos de aprendizaje automático

Mucha gente divide el aprendizaje automático en tres categorías, pero creo que debemos ver todo el ML como cuatro elementos importantes:

- Supervisado
- No supervisado
- Semisupervisado
- Reforzamiento

Cada uno de estos elementos merece libros sobre libros. Las breves definiciones que siguen son referencias simples para familiarizarlo con los términos que escuchará en el campo.

Definición rápida: aprendizaje supervisado En este

libro, nos centraremos en la categoría más común de aprendizaje automático, el aprendizaje automático supervisado (a veces llamado aprendizaje supervisado o simplemente supervisado para abreviar). ML supervisado simplemente significa que tenemos una clave de respuesta para cada pregunta que estamos usando para entrenar nuestra máquina. Es decir, nuestros datos están etiquetados. Entonces, si estamos tratando de enseñarle a una máquina a distinguir si una foto contiene un pájaro, podemos inmediatamente

calificar la IA según si estuvo bien o mal. Como un Scantron, tenemos la clave de respuestas. Pero a diferencia de un Scantron y debido a que es matemática de probabilidad, también podemos identificar cuán incorrecta fue la respuesta.

Si una IA está 90 % segura de que una foto de un pájaro es un pájaro, aunque haya acertado la respuesta, podría mejorar en un 10 %. Esto ilumina el aspecto de "entrenamiento" de la IA con una gratificación inmediata basada en datos.

> No se preocupe si no tiene cientos de preguntas y respuestas etiquetadas listas para usar. En este libro, le proporcionaremos datos etiquetados o le mostraremos cómo generarlos usted mismo.

Definición rápida: aprendizaje no supervisado El

aprendizaje no supervisado no requiere que tengamos una clave de respuestas. Sólo necesitamos preguntas. El aprendizaje automático no supervisado sería ideal, ya que la mayoría de la información en el mundo no viene con etiquetas. Esta categoría de aprendizaje automático se centra en lo que una máquina podría aprender e informar a partir de datos no etiquetados. Si bien este tema puede parecer un poco confuso, ¡los humanos lo realizan todos los días! Por ejemplo, si te doy una foto de mi jardín y te pregunto cuántos tipos diferentes de plantas tengo, podrías decirme la respuesta, y no tienes que saber el género y la especie de cada planta. Es un poco de cómo damos sentido a nuestros propios mundos. Gran parte del aprendizaje no supervisado se centra en categorizar grandes cantidades de datos para su uso.

Definición rápida: aprendizaje semisupervisado La

mayoría de las veces, no vivimos con datos 100% sin etiquetar. Para recuperar el ejemplo del jardín de antes, no conoce el género y la especie de cada planta, pero tampoco es completamente incapaz de clasificar las plantas como A y B. Podrías decirme que tengo diez plantas compuestas de tres flores y siete hierbas. Tener una pequeña cantidad de etiquetas conocidas es muy útil, ¡y la investigación actual está en llamas con avances semisupervisados!

Es posible que haya escuchado el término redes generativas o redes antagónicas generativas (GAN). Estas construcciones de IA populares se mencionan en numerosos artículos de noticias de IA y se derivan de tácticas de aprendizaje semisupervisadas. Las redes generativas se entrenan con ejemplos de lo que nos gustaría que la red creara y, a través de un método semisupervisado, se construyen nuevos ejemplos. Las redes generativas son excelentes para crear contenido nuevo a partir de un pequeño subconjunto de datos etiquetados. Los ejemplos populares de GAN a menudo tienen sus propios sitios web, como https://thispersondoesnotexist.com están creciendo en popularidad y los creativos están teniendo un día de campo con la producción semisupervisada.

Definición rápida: aprendizaje por refuerzo La

forma más sencilla de explicar el aprendizaje por refuerzo es mostrar que es necesario al manejar una actividad más del mundo real, frente a las construcciones hipotéticas anteriores.

Por ejemplo, si estamos jugando al ajedrez y empiezo mi juego moviendo un peón, ¿fue un buen movimiento o un mal movimiento? O si quiero que un robot patee una pelota a través de un aro y comience dando un paso, ¿es eso bueno o malo? Al igual que con un ser humano, la respuesta depende de los resultados. Es una colección de movimientos para obtener la máxima recompensa, y no siempre hay una acción singular que produzca un resultado singular. Entrenar a un robot para que dé el primer paso o mire primero es importante, pero probablemente no tanto como lo hace en otros momentos críticos. Y esos momentos críticos están potenciados por recompensas como refuerzo.

Si estuviera enseñando a una IA a jugar a Super Mario Bros., ¿quiero una puntuación alta o una victoria rápida? Las recompensas enseñan a la IA qué combinación de movimientos es óptima para maximizar el objetivo. El aprendizaje por refuerzo (RL) es un campo en expansión y, a menudo, se ha combinado con otras formas de IA para cultivar el máximo resultado.

Sobrecarga de información

Está bien sorprenderse por la cantidad de aplicaciones de aprendizaje automático que se acaban de mencionar. En cierto modo, es por eso que necesitamos un marco como TensorFlow.js. ¡Ni siquiera podemos comprender todos los usos de estos fantásticos sistemas y sus efectos en las próximas décadas! Mientras pensamos en esto, la era de la IA y el ML está aquí, y vamos a ser parte de ella. El aprendizaje supervisado es un excelente primer paso hacia todos los beneficios de la IA.

Cubriremos juntos algunos de los usos más emocionantes pero prácticos del aprendizaje automático. En algunos aspectos, solo rascaremos la superficie, mientras que en otros, profundizaremos en el corazón de cómo funcionan. Estas son algunas de las categorías generales que cubriremos. Todos estos son conceptos de aprendizaje supervisado:

- Categorización de imágenes
- Procesamiento del lenguaje natural (NLP) •

Segmentación de imágenes

Uno de los principales objetivos de este libro es que, si bien puede comprender los conceptos de las categorías, no se verá limitado por ellas. Nos apoyaremos en la experimentación y la ciencia práctica. Algunos problemas se pueden resolver con un exceso de ingeniería y otros se pueden

resuelto por ingeniería de datos. Pensar en IA y aprendizaje automático es la clave para ver, identificar y crear nuevas herramientas con TensorFlow.js.

La IA está en todas partes

Estamos entrando en un mundo en el que la IA se infiltra en todo. Nuestros teléfonos ahora están acelerados en hardware de aprendizaje profundo. Las cámaras están aplicando detección de IA en tiempo real y, en el momento de escribir este artículo, algunos automóviles circulan por nuestras calles sin un ser humano.

El año pasado, incluso noté que mis correos electrónicos comenzaron a escribirse solos con una opción de "tabulador para completar" para terminar mis oraciones. Esta característica, más a menudo de lo que me gustaría admitir, es más clara y concisa que cualquier cosa que hubiera escrito originalmente. Es un logro visible significativo que eclipsa la IA de aprendizaje automático olvidada que nos ha estado protegiendo del spam en esa misma bandeja de entrada durante años.

A medida que se desarrolla cada uno de estos planes para el aprendizaje automático, se demandan nuevas plataformas. Estamos impulsando los modelos cada vez más en dispositivos de última generación como teléfonos, navegadores y hardware. Tiene sentido que estemos buscando nuevos lenguajes para llevar la antorcha. Era solo cuestión de tiempo antes de que la búsqueda terminara con la opción obvia de JavaScript.

Un recorrido por lo que ofrecen los marcos

¿Cómo es el aprendizaje automático? Aquí hay una descripción precisa que hará Ph.D. los estudiantes se estremecen con su sucinta simplicidad.

Con el código normal, un ser humano escribe el código directamente y una computadora lee e interpreta ese código, o algún derivado del mismo. Ahora estamos en un mundo donde un humano no escribe el algoritmo, entonces, ¿qué sucede realmente? ¿De dónde viene el algoritmo?

Es simplemente un paso adicional. Un humano escribe el entrenador del algoritmo. Con la ayuda de un marco, o incluso desde cero, un ser humano describe en código los parámetros del problema, la estructura deseada y la ubicación de los datos de los que aprender. Ahora la máquina ejecuta este programa de entrenamiento, que continuamente escribe un algoritmo en constante mejora como solución a ese problema. En algún momento, detiene este programa y toma el último resultado del algoritmo y lo usa.

¡Eso es todo!

El algoritmo es mucho más pequeño que los datos que se utilizaron para crearlo. Se pueden usar gigabytes de películas e imágenes para entrenar una solución de aprendizaje automático durante semanas, todo solo para crear unos pocos megabytes de datos para resolver un problema muy específico.

El algoritmo resultante es esencialmente una colección de números que equilibran la estructura perfilada identificada por el programador humano. La colección de números y su gráfico neural asociado a menudo se denomina modelo.

Probablemente haya visto estos gráficos en artículos técnicos, dibujados como una colección de nodos de izquierda a derecha como la Figura 1-3.

Figura 1-3. Ejemplo de una red neuronal densamente conectada

Nuestro marco TensorFlow.js maneja la API para especificar la estructura o arquitectura del modelo, cargar datos, pasar datos a través de nuestro proceso de aprendizaje automático y, en última instancia, ajustar la máquina para que prediga mejor las respuestas a la entrada dada la próxima vez. hora. Aquí es donde se manifiestan los beneficios reales de TensorFlow.js. Todo lo que tenemos que preocuparnos es ajustar correctamente el marco para resolver el problema con suficientes datos y luego guardar el modelo resultante.

¿Qué es un modelo?

Cuando crea una red neuronal en TensorFlow.js, es una representación de código de la red neuronal deseada. El marco genera un gráfico con valores aleatorios seleccionados inteligentemente para cada neurona. El tamaño del archivo del modelo generalmente se fija en este punto, pero el contenido evolucionará. Cuando se hace una predicción empujando datos a través de una red no entrenada de valores aleatorios, obtenemos una respuesta que generalmente está tan lejos de la respuesta correcta como la pura probabilidad aleatoria. Nuestro modelo no se ha entrenado con ningún dato, por lo que es terrible en su trabajo. Entonces, como desarrollador, nuestro código que escribimos está completo, pero el resultado sin entrenamiento es pobre.

Una vez que se han producido iteraciones de entrenamiento durante un período de tiempo significativo, los pesos de la red neuronal se evalúan y luego se ajustan. La velocidad, a menudo llamada tasa de aprendizaje, afecta la solución resultante. Después de dar miles de estos pequeños pasos al ritmo de aprendizaje, comenzamos a ver una máquina que mejora, y estamos diseñando un modelo con una probabilidad de éxito mucho mayor que la máquina original. ¡Hemos dejado la aleatoriedad y convergido en números que hacen que la red neuronal funcione! Esos números asignados a las neuronas en una estructura dada son el modelo entrenado.

TensorFlow.js sabe cómo realizar un seguimiento de todos estos números y gráficos computacionales para que no tengamos que hacerlo, y también sabe cómo almacenar esta información en un formato adecuado y consumible.

Una vez que tenga esos números, nuestro modelo de red neuronal puede detener el entrenamiento y solo usarse para hacer predicciones. En términos de programación, esto se ha convertido en una función simple. Entran datos y salen datos.

La alimentación de datos a través de una red neuronal se parece mucho al azar, como se muestra en la Figura 1-4, pero en el mundo de la computación es una máquina delicada de probabilidad equilibrada y clasificación que tiene resultados consistentes y reproducibles. Los datos se introducen en la máquina y sale un resultado probabilístico.

Figura 1-4. Una metáfora de red equilibrada

En el próximo capítulo, experimentaremos con la importación y predicción con un modelo completamente entrenado. Utilizaremos el poder de horas de capacitación para obtener un análisis inteligente en microsegundos.

En este libro

Este libro está diseñado para que pueda empacarlo para unas vacaciones y, una vez que haya encontrado su pequeño pedazo de cielo, lea junto con el libro, aprenda los conceptos y revise las respuestas. Las imágenes y las capturas de pantalla deberían ser suficientes para explicar la parte más vulnerable de TensorFlow.js.

Sin embargo, para que realmente capte los conceptos, deberá ir más allá de simplemente leer el libro. A medida que se desarrolla cada concepto, debe codificar, experimentar y probar los límites de TensorFlow.js en una computadora real. Para cualquiera de ustedes que es nuevo en el campo del aprendizaje automático, es esencial que consolide la terminología y los flujos de trabajo que está viendo por primera vez. Tómese su tiempo para trabajar con los conceptos y el código de este libro.

Código asociado

A lo largo del libro, hay un código fuente ejecutable para ilustrar las lecciones y la funcionalidad de TensorFlow.js. Si bien en algunos casos se proporciona la fuente completa, en la mayoría de los casos el código impreso se limitará a las partes más destacadas. Es aconsejable que descargue inmediatamente el código fuente que se alinea con este libro. Incluso si planea escribir código desde cero junto con los ejemplos, es probable que las configuraciones pequeñas con las que puede tener problemas ya estén resueltas y referenciadas en el código asociado.

Puede ver la página de origen de GitHub en https://github.com/GantMan/learn-t s.

Si no está familiarizado con GitHub y Git, simplemente puede descargar el código fuente del proyecto más reciente en un solo archivo ZIP y consultarlo.

Puede descargar el archivo ZIP de origen desde https://github.com/GantMan/learn-t s/ archive/master.zip.

El código fuente está estructurado para que coincida con cada capítulo. Debería poder encontrar todos los recursos de capítulos en la carpeta con el mismo nombre. En cada carpeta de capítulos, encontrará como máximo cuatro carpetas que contienen la información de la lección. Esto se revisará en el Capítulo 2 cuando ejecute su primer código TensorFlow.js. Por ahora, familiarícese con el propósito de cada carpeta para que pueda seleccionar el código de ejemplo que mejor se adapte a sus necesidades de aprendizaje.

La carpeta adicional

Esta carpeta contiene todos los materiales adicionales a los que se hace referencia en un capítulo, incluida la documentación o el material de referencia adicional. El material de estas secciones son archivos útiles para cada capítulo.

La carpeta del nodo

Esta carpeta contiene una implementación específica de Node.js del código del capítulo para una solución basada en servidor. Es probable que esta carpeta contenga varios proyectos específicos dentro de ella. Los proyectos de Node.js vendrán con algunos paquetes adicionales instalados para simplificar el proceso de experimentación. Los proyectos de ejemplo para este libro utilizan lo siguiente:

nodemonio

Nodemon es una utilidad que monitoreará cualquier cambio en su fuente y reiniciará automáticamente su servidor. Esto se usa para que pueda guardar sus archivos e inmediatamente ver sus actualizaciones asociadas.

ts-nodo

TypeScript tiene muchas opciones, sobre todo tipeo fuerte. Sin embargo, por accesibilidad, este libro se centra en JavaScript y no en TypeScript. El módulo de nodo ts está en su lugar para compatibilidad con ECMAScript. Puede escribir la sintaxis moderna de JavaÿScript en estos ejemplos de nodos y, a través de TypeScript, el código funcionará.

Estas dependencias se identifican en el archivo package.json. Los ejemplos de Node.js son para ilustrar soluciones de servidor con TensorFlow.js y, por lo general, no es necesario abrirlos en un navegador.

Para ejecutar estos ejemplos, instale las dependencias con Yarn o Node Package Manager (NPM), y luego ejecute el script de inicio:

```
# Instalar dependencias con NPM $
npm i
# Ejecute el script de inicio para iniciar el servidor $
npm run start
# O usa yarn para instalar dependencias $ yarn

# Ejecute el script de inicio para iniciar el servidor $
yarn start
```

Después de iniciar el servidor, verá los resultados de los registros de la consola en su terminal. Cuando haya terminado de revisar los resultados, puede usar Ctrl+C para salir del servidor.

La carpeta simple

Esta carpeta contendrá soluciones donde no se utilizó NPM. Todos los recursos se colocan simplemente en archivos HTML solitarios para ser servidos. Esta es, con diferencia, la solución más sencilla y la más utilizada. Esta carpeta probablemente contendrá la mayoría de los resultados.

la carpeta web

Si está familiarizado con las aplicaciones web de NPM basadas en clientes, se sentirá cómodo con la carpeta web . Es probable que esta carpeta contenga varios proyectos específicos dentro de ella. Los ejemplos de carpetas web se agrupan con Parcel.js. Es un paquete multinúcleo rápido para proyectos web. Parcel proporciona Hot Module Replacement (HMR) para que pueda guardar sus archivos y ver de inmediato que la página refleja los cambios en su código, al tiempo que proporciona un registro de errores amigable y acceso a ECMAScript.

Para ejecutar estos ejemplos, instale las dependencias con Yarn o NPM y luego ejecute el script de inicio:

```
# Instalar dependencias con NPM $
npm i
# Ejecute el script de inicio para iniciar el servidor $
npm run start
# O usa yarn para instalar dependencias $ yarn

# Ejecute el script de inicio para iniciar el servidor $
yarn start
```

Después de ejecutar el paquete, se abrirá una página web con su navegador predeterminado y accederá a una URL local para ese proyecto.

> Si un proyecto usa un recurso como una foto, existirá un archivo credit.txt en la carpeta raíz de ese proyecto para acreditar adecuadamente al fotógrafo y fuente.

Secciones del capítulo

Cada capítulo comienza identificando los objetivos del capítulo y luego se sumerge de inmediato. Al final de cada capítulo, se le presenta un Desafío de capítulo, que es un recurso para que aplique inmediatamente lo que acaba de aprender. Las respuestas para cada desafío se pueden encontrar en el Apéndice B.

Finalmente, cada capítulo termina con una agrupación de preguntas que invitan a la reflexión para verificar que ha interiorizado la información del capítulo. Se recomienda que verifique las respuestas usted mismo por código cuando sea posible, pero las respuestas también se proporcionan en el Apéndice A.

Terminología común de IA/ML Es

posible que esté pensando: "¿Por qué un modelo no se llama simplemente función? Los modelos ya tienen un significado en la programación, ¡y no necesitan otro!" La verdad es que esto proviene del origen de los problemas con los que comenzó el aprendizaje automático. Los problemas de los datos originales tenían sus raíces en las estadísticas. Los modelos estadísticos reconocen patrones como suposiciones estadísticas en datos de muestra, por lo que nuestro producto de esta operación matemática en ejemplos es un modelo de aprendizaje automático. Es bastante común que la terminología del aprendizaje automático refleje en gran medida el campo y la cultura de los científicos que la inventaron.

La ciencia de datos viene con una buena cantidad de terminología matemática. Veremos esto como un tema a lo largo del libro, e identificaremos las razones de cada uno. ¡Algunos términos tienen sentido al instante, algunos chocan con los términos existentes de JavaScript y framework, y alguna terminología nueva choca con otra terminología nueva! Nombrar las cosas es difícil. Haremos nuestro mejor esfuerzo para explicar algunos términos clave de una manera memorable y desarrollaremos la etimología en el camino. Los documentos de TensorFlow y TensorFlow.js están repletos de nuevos

Vocabulario para desarrolladores. Lea la siguiente terminología de aprendizaje automático y vea si puede comprender estos términos fundamentales. Está bien si no puedes. Puede volver a este capítulo y hacer referencia a estas definiciones en cualquier momento a medida que avanzamos.

Entrenamiento El entrenamiento es el proceso de intentar mejorar un algoritmo de aprendizaje automático haciendo que revise los datos y mejore su estructura matemática para hacer mejores predicciones en el futuro.

TensorFlow.js proporciona varios métodos para entrenar y monitorear modelos de entrenamiento, tanto en una máquina como en el navegador de un cliente.

por ejemplo, "Por favor, no toques mi computadora, ha estado entrenando durante tres días en mi último algoritmo de flexión de aire".

Conjunto de

entrenamiento A veces llamado datos de entrenamiento, estos son los datos que le mostrará a su algoritmo para que aprenda. Podría pensar: "¿No son todos los datos que tenemos?" La respuesta es no."

En general, la mayoría de los modelos de ML pueden aprender de ejemplos que han visto antes, pero enseñar la prueba no garantiza que nuestro modelo pueda extrapolar para reconocer datos que nunca antes había visto. Es importante que los datos que usamos para entrenar la IA se mantengan separados para responsabilidad y verificación.

por ejemplo, "Mi modelo sigue identificando perros calientes como bocadillos, por lo que tendré que agregar más fotos a mi conjunto de entrenamiento".

Equipo de prueba

Para probar que nuestro modelo puede funcionar con datos que nunca antes había visto, tenemos que mantener algunos datos a un lado para probar y nunca dejar que nuestro modelo entrene o aprenda de ellos. Esto generalmente se denomina conjunto de prueba o datos de prueba. Este conjunto nos ayuda a probar si hemos hecho algo que se generalice a nuevos problemas en el mundo real. El conjunto de prueba es generalmente significativamente más pequeño que el conjunto de entrenamiento.

por ejemplo, "Me aseguré de que el conjunto de prueba fuera una buena representación del problema para el que estamos tratando de entrenar un modelo para resolver".

Conjuntos de validación

Es importante saber este término incluso si no está en el nivel en el que lo necesita. Como escuchará a menudo, la capacitación a veces puede llevar horas, días o incluso semanas. ¡Es un poco alarmante iniciar un proceso de larga duración solo para regresar y descubrir que ha estructurado algo mal y tiene que comenzar todo de nuevo! Mientras que probablemente

no se encontrará con ninguna de esas mega necesidades de capacitación en este libro, esas situaciones podrían usar un grupo de datos para pruebas más rápidas. Cuando esto está separado de sus datos de entrenamiento, es un "método de exclusión" para la validación. Esencialmente, es una práctica en la que se reserva un pequeño conjunto de datos de entrenamiento para realizar pruebas de validación antes de permitir que su modelo se entrene en una infraestructura costosa o durante un tiempo prolongado. Este ajuste y prueba para la validación es su conjunto de validación.

Hay muchas formas de seleccionar, segmentar, estratificar e incluso plegar sus conjuntos de validación. Esto entra en una ciencia que está más allá del alcance de este libro, pero es bueno saberlo cuando discuta o lea y avance sus propios mega conjuntos de datos.

TensorFlow.js tiene parámetros de entrenamiento completos para identificar y graficar los resultados de validación durante el proceso de entrenamiento.

> por ejemplo, "He tallado un pequeño conjunto de validación para usar mientras construimos nuestra arquitectura modelo".

tensores

Cubriremos los tensores con gran detalle en el Capítulo 3, pero vale la pena señalar que los tensores son la estructura de datos optimizada que permite la aceleración de GPU y Web Assembly para inmensos conjuntos de cálculo de IA/ML. Los tensores son los soportes numéricos de los datos.

> por ejemplo, "He convertido su foto en un tensor de escala de grises para ver qué tipo de aumento de velocidad podemos obtener".

Normalización

La normalización es la acción de escalar los valores de entrada a un dominio más simple. Cuando todo se convierte en números, la diferencia en la escasez y la magnitud de los números puede causar problemas imprevistos.

Por ejemplo, si bien el tamaño de una casa y la cantidad de baños en una casa afectan el precio, generalmente se miden en diferentes unidades con números muy diferentes. No todo se mide en la misma escala métrica y, si bien la IA puede adaptarse para medir estas fluctuaciones en los patrones, un truco común es simplemente escalar los datos al mismo dominio pequeño. Esto permite que los modelos entrenen más rápido y encuentren patrones más fácilmente.

> por ejemplo, "He aplicado algo de normalización al precio de la vivienda y la cantidad de baños para que nuestro modelo pueda encontrar patrones entre los dos más rápidamente".

Aumento de datos

En el software de edición de fotografías, podemos tomar imágenes y manipularlas para que parezcan lo mismo en un entorno completamente diferente. Este método efectivamente hace una foto completamente nueva. Tal vez desee su logotipo en el costado de un edificio o en relieve

en una tarjeta de visita. Si intentáramos detectar su logotipo, la foto original y algunas versiones editadas serían útiles en nuestros datos de capacitación de aprendizaje automático.

A menudo, podemos crear nuevos datos a partir de nuestros datos originales que se ajusten al objetivo de nuestro modelo. Por ejemplo, si nuestro modelo va a ser entrenado para detectar rostros, ¡una foto de una persona y una foto reflejada de una persona son fotos válidas y significativamente diferentes!

TensorFlow.js tiene bibliotecas dedicadas al aumento de datos. Veremos datos aumentados más adelante en este libro.

> por ejemplo, "Hemos realizado algunos aumentos de datos reflejando todas las calabazas para duplicar nuestro conjunto de entrenamiento".

Características y caracterización

Mencionamos las características antes cuando hablamos sobre la forma en que los ojos envían lo que es más importante al cerebro. Hacemos lo mismo con ML. Si estuviéramos tratando de hacer una IA que adivine cuánto vale una casa, entonces tenemos que identificar qué entradas son útiles y qué entradas son ruido.

No faltan datos sobre una casa, desde el número de ladrillos hasta la moldura de corona. Si ve mucha televisión de mejoras para el hogar, sabe que es inteligente identificar el tamaño de una casa, la edad, la cantidad de baños, la fecha en que se actualizó por última vez la cocina y el vecindario. A menudo, estas son características clave para identificar el precio de una casa, y le importará más alimentar un modelo con esa información que algo trivial. La caracterización es la selección de estas características de todos los datos posibles que podrían seleccionarse como entradas.

Si decidimos incluir todos los datos que pudimos, le damos a nuestro modelo la oportunidad de encontrar nuevos patrones a costa de tiempo y esfuerzo. No hay razón para elegir características como la cantidad de briznas de césped, el olor de la casa o la iluminación natural al mediodía, incluso si tenemos esa información o creemos que es importante para nosotros.

Incluso una vez que hemos seleccionado nuestras funciones, a menudo hay errores y valores atípicos que retrasarán el entrenamiento de un modelo práctico de aprendizaje automático. Algunos datos simplemente no mueven la aguja hacia un modelo predictivo más exitoso, y la selección de funciones inteligentes hace que la IA sea más inteligente y se entrene rápidamente.

> por ejemplo, "Estoy bastante seguro de que contar la cantidad de signos de exclamación es una función clave para detectar estos correos electrónicos de marketing".

Revisión del capítulo

En este capítulo, hemos atrapado los términos y conceptos del término genérico IA. También hemos abordado los principios clave de lo que cubriremos en este libro. Idealmente, ahora tiene más confianza en los términos y estructuras que son esenciales en el aprendizaje automático.

Preguntas de revisión

Tomemos un momento y asegurémonos de que haya comprendido completamente los conceptos que mencionamos. Tómese un momento para responder las siguientes preguntas:

1. ¿Puede dar una definición adecuada de aprendizaje automático?
2. Si una persona identifica una idea para un proyecto de aprendizaje automático, pero no tiene datos etiquetados, ¿qué recomendaría?
3. ¿Qué tipo de ML sería útil para vencer a tu videojuego favorito?
4. ¿Es el aprendizaje automático la única forma de IA?
5. ¿Un modelo contiene todos los datos de ejemplo de entrenamiento que se usaron para que funcione?
6. ¿Cómo se dividen los datos de aprendizaje automático?

Las soluciones a estos ejercicios están disponibles en el Apéndice A.

Hasta la próxima…

En el Capítulo 2 , preparará su máquina para ejecutar TensorFlow.js y comenzará a ensuciarse las manos con la implementación de bibliotecas reales de TensorFlow.js. Tomará todos los conceptos que cubrimos en este capítulo y comenzará a ver esos términos en bibliotecas de JavaScript populares que son mantenidas por Google.

2

v.js

Hemos estado hablando un poco sobre TensorFlow.js y lo que puede hacer, pero realmente no hemos profundizado en lo que realmente es un marco de aprendizaje automático como TensorFlow.js. En

este capítulo, abordaremos el concepto de un marco de aprendizaje automático y luego nos sumergiremos rápidamente en la escritura de código. Sé que es importante escribir código que tenga algún tipo de resultado tangible, por lo que en este capítulo finalmente logrará que su computadora ejecute TensorÿFlow.js y produzca resultados.

Lo haremos:

• Mire el concepto de TensorFlow.js •

Configure TensorFlow.js • Ejecute un paquete

modelo de TensorFlow.js • Eche un vistazo

profundo a lo que hizo la IA

Comencemos con el marco que usaremos para que todo suceda.

Hola, TensorFlow.js

Dado que nuestro capítulo anterior analizó las filosofías de la antigüedad y el nacimiento del aprendizaje automático como campo, esperaría que los marcos de trabajo de IA tengan una historia que se remonta a principios de la década de 1960. Sin embargo, la IA estuvo estancada durante mucho tiempo, y esta vez a menudo se la llama "invierno de la IA". Los conceptos de IA estaban plagados de incredulidad y cálculos matemáticos extremos para los pocos datos disponibles. Quién

podría culpar a estos investigadores? La mayoría de los desarrolladores de software de hoy dependen de enviar aplicaciones sin escribir álgebra lineal y cálculo habilitados para GPU desde cero, y construir su propia IA no debería ser la excepción. Afortunadamente, gracias a algunas contribuciones de código abierto del equipo de Google Brain, tenemos opciones.

Hay muchas palabras de moda que aparecen cuando comienzas el aprendizaje automático. Se pueden mencionar TensorFlow, TensorFlow Lite y TensorFlow.js, y no está claro para la mayoría de los recién llegados qué significan estos términos o por qué hay tres de ellos. Por ahora, ignoremos el término tensor, ya que escuchó la palabra en el Capítulo 1 y realmente la entenderá en los capítulos siguientes. En su lugar, concentrémonos en definir TensorFlow.js para poder usarlo.

TensorFlow, sin ningún ".js" o "Lite" adicional, fue el primer marco de aprendizaje automático público de Google; el equipo de Google Brain lo lanzó a fines de 2015.1 Este marco se centró en resolver de manera efectiva problemas de aprendizaje automático para Google en la nube con Python. No pasó mucho tiempo antes de que Google se diera cuenta de que impulsar este popular marco de trabajo para IoT y dispositivos móviles que tienen una potencia informática limitada traería beneficios, y eso requería una adaptación de TensorFlow, que se conoce como TensorFlow Lite. Esta exitosa adaptación allanó el camino para llevar los ideales de TensorFlow a otros idiomas.

Probablemente puedas adivinar lo que sucedió después. A principios de 2018, Google anunció una importación de JavaScript respaldada por Google del marco de aprendizaje automático TensorFlow para JavaScript, llamado TensorFlow.js. Este nuevo esfuerzo potenció la practicidad de TensorFlow de una manera completamente nueva. Daniel Smilkov, Nikhil Thorat y Shanqing Cai formaron parte de un equipo que lanzó TensorFlow.js. En la cumbre de desarrolladores de TensorFlow , Smilkov y Thorat entrenan a un modelo para controlar un juego PAC-MAN usando visión por computadora y una cámara web en el navegador.

Fue en ese momento cuando las cadenas "solo para Python" se eliminaron de las opciones de los marcos de trabajo de IA populares y las redes neuronales pudieron atravesar efectivamente el dominio de JavaScript. Si puede ejecutar JavaScript, puede ejecutar AI que funciona con TensorFlow.js ML.

Las tres implementaciones están vivas hoy y crecen con su propósito específico. Al expandir TensorFlow a una implementación de JavaScript, ahora podemos implementar AI/ML con servidores de nodo e incluso el navegador del cliente. En el documento "TensorFlow.js: Machine Learning for the Web and Beyond" (Daniel Smilkov et al., 2019), afirman: "TensorFlow.js ha empoderado a un nuevo conjunto de desarrolladores de la extensa comunidad de JavaScript para construir e implementar modelos de aprendizaje automático y habilitó nuevas clases de computación en el dispositivo". TensorFlow.js puede aprovechar una gran

1 TensorFlow no alcanzó el estado 1.0.0 hasta el 11 de febrero de 2017.

plataforma de dispositivos sin dejar de acceder a la GPU e incluso a Web Assembly. Con Javaÿ Script, nuestro aprendizaje automático puede aventurarse al horizonte y regresar.

También vale la pena señalar que en varias pruebas comparativas, Node superó a Python 3 con una carga de CPU más baja,2 por lo que, si bien Python ha sido el lenguaje adoptado por la mayoría de las IA, JavaScript sirve como plataforma de lenguaje principal para productos y servicios.

Pero no hay necesidad de eliminar o promocionar ningún idioma. Los modelos de TensorFlow se basan en gráficos acíclicos dirigidos (DAG), que son gráficos independientes del lenguaje que son el resultado del entrenamiento. Estos gráficos pueden ser entrenados por un lenguaje y luego convertidos y consumidos por un lenguaje de programación completamente diferente. El objetivo de este libro es brindarle las herramientas que necesitará para aprovechar al máximo el uso de JavaScript y TensorFlow.js.

Aprovechando TensorFlow.js

Para muchas personas, "aprender" a veces puede significar comenzar con los fundamentos, lo que significa comenzar con las matemáticas. Para esas personas, un marco como TensorFlow y una rama pragmática de un marco como TensorFlow.js es un mal comienzo.
En este libro, construiremos proyectos y abordaremos los fundamentos del marco de TensorFlow.js, y dedicaremos poco tiempo, si es que dedicamos alguno, a la magia matemática subyacente.

Marcos como TensorFlow y TensorFlow.js nos ayudan a evitar los detalles del álgebra lineal involucrada. Estás libre de términos como propagación hacia adelante y propagación hacia atrás, así como sus cálculos y cálculo. En cambio, nos centraremos en términos de la industria como inferencia y entrenamiento de modelos.

Si bien TensorFlow.js puede acceder a API de capa inferior (como tfjs-core) para realizar una optimización fundamental en problemas clásicos, esos momentos se dejan a los académicos y usuarios avanzados que tienen una base sólida, independientemente del marco en el que se encuentren. mano. Este libro está destinado a mostrar el poder de TensorFlow.js, y utilizar el trabajo duro y la optimización del marco es la forma en que lo haremos. Dejamos a TensorFlow.js el trabajo de configurar y optimizar nuestro código para que funcione con la amplia variedad de restricciones de dispositivos y API de WebGL.

Incluso podríamos llevar las cosas un poco demasiado lejos y aplicar el aprendizaje automático a algoritmos que podría codificar fácilmente a mano, pero ahí es generalmente donde la mayoría de las personas realmente captan los conceptos con claridad. Resolver problemas simples que comprende con el aprendizaje automático lo ayuda a extrapolar los pasos, la lógica y las ventajas y desventajas de resolver problemas avanzados que nunca podría codificar a mano.

2 Impulso 2x con Node over Python caso de estudio: https://oreil.ly/4Jrbu

En el otro lado de la moneda, algunos fundamentos de las neuronas, las funciones de activación y la inicialización del modelo no se pueden ignorar y pueden requerir alguna explicación. El objetivo de este libro es brindarle un equilibrio saludable entre teoría y practicidad.

Como habrás adivinado, la variedad de plataformas para TensorFlow.js significa que no hay una configuración prescrita única. Podremos ejecutar TensorFlow.js en un cliente o servidor para este libro. Sin embargo, nuestra opción interactiva más tácita es aprovechar al máximo el navegador. Por esa razón, realizaremos la mayor parte de los ejemplos en el navegador. Por supuesto, seguiremos cubriendo los aspectos clave del alojamiento de una solución de servidor de nodo cuando corresponda. Cada una de estas dos herramientas tiene sus inconvenientes y beneficios subyacentes, que mencionaremos a medida que nos adentremos en el poder de TensorFlow.js.

Preparemos TensorFlow.js

Al igual que con cualquier herramienta popular, es posible que observe que hay varios sabores para el paquete TensorFlow.js, así como varias ubicaciones donde puede acceder al código. La mayor parte de este libro se centrará en las versiones más disponibles y "listas para ejecutar" de Tensorÿ Flow.js, lo que significa el cliente del navegador. Las compilaciones optimizadas del marco se realizan para el lado del servidor. Estas compilaciones hablan con la misma API central de C++ subyacente que Python, pero a través de Node.js, lo que le permite aprovechar todo el rendimiento de la tarjeta gráfica o la CPU de su servidor. Los modelos de IA de TensorFlow.js se ejecutan en una variedad de ubicaciones y utilizan una variedad de optimizaciones para cada entorno (consulte la Figura 2-1).

Figura 2-1. Opciones para TensorFlow.js

El conocimiento que aprenderá en este libro se puede aplicar a la mayoría de las plataformas. Para su comodidad, cubriremos el proceso de configuración para las plataformas más comunes. Si no se siente cómodo configurando sus entornos desde cero, simplemente puede acceder a la

proyectos preconfigurados creados para usted en el código fuente asociado con este libro, ubicado en https://github.com/GantMan/learn-t s.

Cómo configurar TensorFlow.js en el navegador

Pasemos a la forma más rápida, versátil y sencilla de ejecutar TensorFlow.js.
Para que TensorFlow.js se ejecute en su navegador, en realidad es bastante fácil. Voy a suponer que está familiarizado con los conceptos básicos de JavaScript y que ha importado bibliotecas de JavaScript en código existente anteriormente. TensorFlow.js admite una amplia variedad de formas de incluirse, por lo que los desarrolladores de cualquier experiencia pueden acceder a él. Si está familiarizado con la inclusión de dependencias de JavaScript, estará familiarizado con estas prácticas comunes. Podemos importar TensorFlow.js a una página de dos maneras:

- Usar NPM •
- Incluir una etiqueta de script

Uso de NPM

Una de las formas más populares de administrar las dependencias de su sitio web es usar un administrador de paquetes. Si está acostumbrado a crear proyectos con NPM o Yarn, puede acceder al código a través del registro de NPM en https://oreil.ly/R2lB8. Simplemente instale la dependencia en la línea de comando:

```
# Importar con npm $
npm y @tensorflow/tfjs

# O hilo $
hilo añadir @tensorflow/tfjs
```

Una vez que haya importado el paquete tfjs , puede importar este código en su Javaÿ
Proyecto de script con el siguiente código de importación ES6 JavaScript:

```
importar * como tf desde '@tensorflow/tfjs';
```

Incluir una etiqueta de

secuencia de comandos Si un sitio web no utiliza un administrador de paquetes, simplemente puede agregar una etiqueta de secuencia de comandos al documento HTML. Esta es la segunda forma en que puede incluir TensorFlow.js en su proyecto. Puede descargar y alojar TensorFlow.js localmente o utilizar una red de entrega de contenido (CDN). Apuntaremos la etiqueta del script a una fuente de script alojada en CDN:

```
<script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@2.7.0/dist/tf.min.js"> </script>
```

Además del almacenamiento en caché en los sitios web, las CDN son extremadamente rápidas porque utilizan ubicaciones de borde para garantizar una entrega rápida en todo el mundo.

Configuración con el nodo TensorFlow.js

El paquete TensorFlow.js que usamos para el navegador funciona bien con Node.js, y esta es una buena solución si planea experimentar solo temporalmente con Node.js. Una buena regla es usar el simple /tfjs sobre la importación de /tfjs-node si no está interesado en hospedar un proyecto en vivo para otros o capacitarse en grandes cantidades de datos.

Si su objetivo es ir más allá de la experimentación y convertirse en Node.js eficaz con Tensor-Flow.js, debería dedicar algún tiempo a mejorar su configuración de Node.js con algunos de estos paquetes alternativos. Hay dos mejores distribuciones de TensorFlow.js que están diseñadas específicamente para Node y speed. Son tfjs node y tfjs-node-gpu. Tenga en cuenta que cada máquina de desarrollador es única y sus instalaciones y experiencias pueden
variar.

Para Node.js, probablemente elija entre @tensorflow/tfjs-node o @tensorflow/tfjs-node-gpu. Puede utilizar el último paquete impulsado por GPU si su computadora está configurada con una GPU NVIDIA y correctamente configurada con el software CUDA. Compute Unified Device Architecture (CUDA) permite el acceso directo acelerado por GPU a través de una plataforma informática paralela para hardware NVIDIA.

Si bien el paquete GPU es absolutamente el más rápido de las opciones de TensorFlow.js, también es el menos probable de estar listo y configurado para la mayoría de las máquinas, debido a sus limitaciones de hardware y software. Por ahora, nuestros ejemplos funcionarán en la instalación de tfjs-node y le dejarán a usted la configuración opcional de CUDA.

```
# Importar con npm $
npm i @tensorflow/tfjs-node

# o hilo
$ hilo añadir @tensorflow/tfjs-node
```

A menudo, si su computadora no ha sido configurada para desarrollar bibliotecas C++ avanzadas, es posible que tenga que hacer un poco de instalación para que su máquina esté lista. Este agujero de conejo solo es necesario si está buscando trabajar activamente con tfjs-node o tfjs-node-gpu.

Install Tips

On Windows si tiene problemas con node-gyp, esto a veces puede significar que necesitará descargar Visual Studio y verificar la carga de trabajo "Desarrollo de escritorio en C ++". Consulte la Figura 2-2 para ver una captura de pantalla del proceso de instalación.

Figura 2-2. Instalar C++ para que node- gyp ejecute tfjs-node

En una Mac, es posible que deba instalar Xcode y ejecutar xcode-select --install. Revise el GitHub de node-gyp para obtener documentación específico para su máquina.

Una vez que tenga la carga de trabajo de C++ y pueda instalar el paquete, es posible que deba reconstruir tfjs-node en su máquina con el siguiente comando: **npm rebuild @tensorflow/tfjs-node --build-from-source.**

Si desea utilizar tfns-node-gpu, deberá configurar CUDA para que funcione con su tarjeta gráfica. Esa instalación puede requerir bastante tiempo e instrucciones.
Debe consultar la documentación CUDA más reciente para obtener esas instrucciones.

Si su instalación de NPM fue exitosa, ¡felicidades! Está listo para importar desde este paquete. Si tiene Node configurado para manejar ES6, puede importar con lo siguiente:

```
importar * como tf desde '@tensorflow/tfjs-node';
```

Si no ha configurado su paquete Node.js para manejar las importaciones de ES6, aún puede acceder al código con un requerimiento clásico:

```
const tf = require('@tensorflow/tfjs-node');
```

Verificando que TensorFlow.js esté funcionando

Todos los métodos anteriores harán que una variable tf esté disponible en su código JavaScript, lo que le da acceso a TensorFlow.js. Para asegurarnos de que nuestra importación funcionó correctamente, registremos la versión de la biblioteca TensorFlow.js importada.

Agregue este código a su JavaScript, y si ve una versión impresa en la consola, ¡su importación está lista!

```
consola.log(tf.version.tfjs);
```

Cuando se ejecuta la página, podemos hacer clic derecho en la página e inspeccionar para acceder a los registros de la consola de JavaScript. Allí encontraremos el resultado de nuestro comando de registro, "3.0.0" o cualquier versión de TensorFlow.js que hayas importado. Para el ejemplo de Node.js, el valor simplemente se imprimirá directamente en la consola.

> Antes de acceder a las funciones de la variable tf (biblioteca TensorFlow.js), normalmente deberá asegurarse de que TensorFlow.js haya cargado correctamente un backend y esté listo. El código antes mencionado pasa por alto esta verificación, pero siempre es prudente ejecutar su código inicial a la espera de la promesa de tf.ready().

Descargue y ejecute estos ejemplos Como se mencionó en el Capítulo 1, tiene acceso al código de este libro. Para asegurarse de que no tiene que configurar estos proyectos desde cero en cada ejemplo, asegúrese de tener el código fuente de cada proyecto, incluido el código simple que se muestra anteriormente.

Descargue el proyecto de la forma que prefiera desde el repositorio del libro: https://github.com/GantMan/learn-t s .

Navegue hasta el directorio del Capítulo 2 y asegúrese de que puede ejecutar el código en su máquina.

Ejecutando el ejemplo

simple En el capítulo 2/simple/simplest-example estamos evitando NPM y simplemente extrayendo nuestro código de un CDN. Con la forma en que este código está estructurado actualmente, ¡ni siquiera tenemos que alojar el sitio! Simplemente podemos abrir index.html en cualquier navegador moderno, ¡y funcionará!

En algún momento, tendremos que alojar estos ejemplos simples porque accederemos a activos adicionales que requieren URI completos. Podemos hacer esto muy fácilmente usando un pequeño servidor web para alojar los archivos. El servidor web más pequeño que conozco se llama Servidor web para Chrome y tiene un divertido "200 OK!" dibujado a mano. logo. En cinco minutos, podemos hacer que nuestros archivos se sirvan correctamente en un servidor local.

Puede encontrar Web Server for Chrome en Chrome Web Store como una extensión. En este libro a veces llamaremos a este complemento “200 OK!” Cuando dirija el servidor web al archivo index.html, automáticamente entregará el archivo por usted, y todos los archivos adyacentes serán accesibles con sus URL asociadas, como se requerirá en lecciones posteriores. La interfaz de la aplicación debería parecerse a la Figura 2-3.

Figura 2-3. Servidor web para Chrome 200 ¡OK! diálogo

Si desea examinar otras opciones o desea un enlace al complemento de Chrome mencionado, eche un vistazo a chapter2/extra/hosting-options.md para encontrar el que funcione para usted.

Y, por supuesto, si encuentra una opción fantástica que no está en la lista, contribuya con una solicitud de extracción.

Una vez que encuentre un servidor que ejecute el ejemplo simple de una manera que disfrute, puede usar ese servicio para todas las opciones simples en el futuro.

Ejecución del ejemplo web de NPM

Si está más familiarizado con NPM, el ejemplo básico de NPM para este proyecto utiliza Parcela. Parcel es el paquete de aplicaciones más rápido sin configuración. También incluye Hot Module Reloading para obtener actualizaciones en tiempo real y un excelente registro de errores.

Para ejecutar el código, vaya a chapter2/web/web-example y realice una instalación de NPM (npm i). Una vez hecho esto, hay una secuencia de comandos en el paquete.json que inicia todo.
Simplemente puede ejecutar el script de inicio:

```
$ npm inicio de ejecución
```

¡Eso es todo! Usaremos este método para ejecutar todo el código basado en NPM en el libro.

Ejecutar el ejemplo de Node.js El

ejemplo de Node.js es tan fácil de ejecutar como el ejemplo de Parcel NPM. Si bien Node.js generalmente no tiene opiniones, los ejemplos de Node.js en este libro incluirán algunas dependencias de desarrollo con opiniones para que podamos hacer que nuestro código de ejemplo de Node.js se alinee con los ejemplos del navegador. El código a lo largo de este libro aprovechará al máximo ECMAÿScript. Hacemos esto con algo de transpilación, observación de archivos y magia de nodos.

Para preparar este ejemplo, vaya a chapter2/node-example y realice una instalación de NPM (npm i). Si tiene algún problema, es posible que deba ejecutar npm i -g ts-node nodemon node-gyp para asegurarse de tener las bibliotecas necesarias para hacer realidad toda nuestra magia. Una vez que sus paquetes de nodos estén correctamente en su lugar, puede iniciar el proyecto en cualquier momento ejecutando el script de inicio:

```
$ npm inicio de ejecución
```

El código se transpila a través de TypeScript y nodemon fácil de recargar. Si todo funcionó correctamente, verá la versión instalada de TensorFlow.js impresa directamente en la consola/terminal donde ejecutó el servidor.

Usemos algo de TensorFlow.js real

Ahora que tenemos TensorFlow.js, ¡utilicémoslo para hacer algo épico! Bien, eso es una gran simplificación: si fuera tan fácil, el libro estaría terminado. Todavía hay una montaña de cosas que aprender, pero eso no nos impide tomar una góndola para obtener una vista de alto nivel.

TensorFlow.js tiene muchos códigos y modelos preescritos que podemos utilizar. Estas bibliotecas escritas previamente nos ayudan a obtener los beneficios de utilizar TensorFlow.js sin comprender por completo los conceptos subyacentes.

Si bien hay muchos modelos impulsados por la comunidad que funcionan bastante bien, la lista oficial mantenida de modelos de TensorFlow.js se encuentra en TensorFlow GitHub bajo un repositorio llamado tfjs-models. Para mayor estabilidad, los usaremos tan a menudo como podamos en este libro. Puede examinar los enlaces aquí: https://github.com/tensorflow/t s-models.

Para esta incursión en la ejecución de modelos reales de TensorFlow.js, escojamos algo con una entrada y salida relativamente simples. Usaremos el clasificador de toxicidad de TensorFlow.js para verificar si la entrada de texto es insultante o no.

El clasificador de toxicidad de

Google proporciona algunos modelos "listos para usar" de diversa complejidad. Un modelo beneficioso se llama modelo de toxicidad, que es quizás uno de los modelos más sencillos y útiles para principiantes.

Como toda programación, un modelo requerirá una entrada específica y proporcionará una salida específica. Para empezar, echemos un vistazo a para qué sirven en este modelo. La toxicidad detecta contenido tóxico como amenazas, insultos, palabrotas y odio generalizado. Dado que no son necesariamente mutuamente excluyentes, es importante que cada una de estas violaciones tenga su propia probabilidad.

El modelo de toxicidad intenta identificar la probabilidad de que una entrada dada sea verdadera o falsa para las siguientes características:

- Ataque de identidad
- Insulto
- Obsceno
- Toxicidad severa •

Sexualmente explícito

- Amenaza
- Toxicidad

Cuando le da al modelo una cadena, devuelve una matriz de siete objetos para identificar la predicción de porcentaje de probabilidad para cada infracción específica. Los porcentajes son resentido como dos valores Float32 entre cero y uno.

Si una oración seguramente no es una violación, las probabilidades darán la mayor parte del valor al índice cero en la matriz Float32 .

Por ejemplo, [0.7630404233932495, 0.2369595468044281] lee que la predicción para esta violación en particular es 76% no una violación y 24% probablemente una violación.

Esto puede ser bastante "Espera, ¿¡qué!?" momento para la mayoría de los desarrolladores. Es un poco extraño obtener probabilidades donde estamos acostumbrados a verdadero y falso, ¿no? Pero de manera intuitiva, siempre hemos entendido que el lenguaje tiene muchas áreas grises. ¡La ciencia exacta de los insultos a menudo depende de la persona e incluso del día!

Por esta razón, el modelo tiene una característica adicional que le permite pasar un umbral que identificará cuándo una infracción en particular supera el límite asignado. Cuando se detecta un insulto más allá del umbral, el indicador de coincidencia se establece en verdadero. Esta es una pequeña bonificación agradable para ayudarlo a mapear rápidamente los resultados de infracciones significativas. Elegir un umbral válido depende de sus necesidades y de la situación. Puede disparar desde la cadera, pero si necesita orientación, las estadísticas tienen todo tipo de herramientas que puede revisar. Lea los gráficos de características operativas del receptor (ROC) para trazar y elegir un umbral óptimo para sus necesidades.

Para activar el modelo de Toxicidad, tendremos que escribir algo insultante. El siguiente ejemplo utiliza un insulto basado en la apariencia. El insulto evita el uso de blasfemias pero sigue siendo ofensivo. Esto no está dirigido a nadie en particular y pretende ilustrar las capacidades de la IA para comprender e identificar comentarios tóxicos.

Es importante elegir un insulto que sea fácil de reconocer para los humanos, pero difícil para una computadora. La detección de sarcasmo es difícil en forma de texto y ha sido un problema importante en informática. Para probar seriamente este modelo, el insulto debe evitar una redacción incendiaria común y flagrante. Ejecutar el modelo de Toxicidad en un insulto particularmente astuto con el umbral establecido en 0.5 produce la matriz que se muestra en el Ejemplo 2-1.

La entrada de insulto: "¡Parece una mujer de las cavernas, solo que mucho menos inteligente!"

Ejemplo 2-1. El informe de toxicidad completo en la frase de entrada

```
[{
        "etiqueta":"identidad_ataque",
        "resultados":[{ "probabilidades":
                { "0":0.9935033917427063,
                    "1":0.006496586836874485

                }, "coincidencia":falso
            }]
    },
     { "etiqueta":"insulto",
        "resultados":
                [{ "probabilidades":
                    { "0":0.5021483898162842,
```

{ "etiqueta":"obsceno",
"resultados":
[{ "probabilidades":
{

```
                    "0":0.9993441700935364,
                    "1":0.0006558519671671093 },
                "coincidencia":falso
            }]
    },
     { "etiqueta":"toxicidad_grave",
        "resultados":[{ "probabilidades":
                { "0":0.9999980926513672,
                    "1":0.000001861434952843410 4

                }, "coincidencia":falso
            }]
    },
     { "etiqueta":"sexual_explícito",
        "resultados":[{ "probabilidades":
                { "0":0.9997043013572693,
                    "1":0.00029564235592260957 },
                    "coincidencia":falso

            }]
    },
     { "etiqueta":"amenaza",
        "resultados":
                [{ "probabilidades":
                    { "0":0.9989342093467712,
                    "1":0.0010658185929059982
                }, "coincidencia":falso
            }]
    },
     { "etiqueta":"toxicidad",
        "resultados":
                [{ "probabilidades":
                    { "0":0.4567308723926544,
                    "1":0.543269157409668 },
                "coincidencia":verdadero
            }]
}]
```


Como puede ver en el Ejemplo 2-1, nos colamos por debajo del radar de "insulto" por un cabello (50,2 % falso), pero nos afectó el indicador de toxicidad, lo que resultó en "coincidencia": verdadero. Esto es bastante impresionante, porque no tengo ningún lenguaje explícitamente ofensivo en la oración. Como programador, no sería sencillo escribir un algoritmo para capturar e identificar este insulto tóxico, pero la IA fue entrenada para identificar los patrones complejos del lenguaje tóxico después de estudiar montones de insultos etiquetados para que no tengamos que hacerlo nosotros.

El ejemplo anterior usa una sola oración en una matriz como entrada. Si incluye varias oraciones como entrada, su índice de oraciones se corresponderá directamente con su índice de resultados para cada categoría.

Pero no confíe en mi palabra; ahora es tu turno de ejecutar el código. Puede agregar el modelo a su sitio web con NPM a través de esto:

```
$ npm install @tensorflow-models/toxicity
```

y luego importar la biblioteca:

```
importar * como toxicidad de "@tensorflow-models/toxicity";
```

O puede agregar la secuencia de comandos directamente desde una CDN.3 El pedido importa con las etiquetas de secuencias de comandos, así que asegúrese de colocar su etiqueta en la página antes de intentar usar el modelo:

```
<script src="https://cdn.jsdelivr.net/npm/@tensorflow-models/toxicity@1.2.2"> </script>
```

Cualquiera de los ejemplos anteriores proporcionará resultados en una variable de toxicidad lista para usar. Usaremos el método de carga de esta variable para cargar la promesa del modelo ML. Y a partir de ese modelo, podemos utilizar el método de clasificación en una serie de oraciones.

Aquí hay un ejemplo de cómo cargar el modelo y ejecutar la clasificación en tres oraciones. Este ejemplo exacto se puede encontrar en tres formas diferentes en las secciones asociadas del código del capítulo en GitHub.

```
// confianza de predicción positiva mínima // Si esto
no se pasa, el valor predeterminado es 0,85 umbral
const = 0,5;

// Carga el modelo
toxicidad.load(umbral).then((modelo) => { const frases
  = [ "¡Eres un cabeza de caca!", "Me gustan las
    tortugas", "¡Cállate!" ];


  // Pide al modelo que clasifique las entradas
  model.classify(sentences).then((predictions) => {
    // resultados semi-bonitos-impresos
    console.log(JSON.stringify(predictions, null, 2)); }); });
```

El modelo se carga en el navegador con un umbral.

3 Tenga en cuenta que esta versión está bloqueada en 1.2.2.

Se le pide al modelo cargado que clasifique las entradas.

El objeto se imprime muy bien utilizando la notación de objetos de JavaScript.

> Si ejecuta este código en un navegador, deberá ver la consola para ver el resultado. Puede navegar a la consola desde la inspección de la página o, en general, puede presionar Control+Shift+J en Windows o Command+Option+J en Mac. Si está ejecutando esto desde la línea de comando con npm start, debería ver el resultado inmediatamente en la consola.

Los resultados de varias oraciones se agrupan por categoría de toxicidad. Por lo que el código anterior intenta identificar cada oración dependiendo de cada categoría. Por ejemplo, el resultado de "insulto" de lo anterior debería ser similar al Ejemplo 2-2.

Ejemplo 2-2. Resultados de la sección de insultos

```
...
{
   "etiqueta": "insulto",
   "resultados": [ {

        "probabilidades":
          { "0": 0.05905626341700554,
          "1": 0.9409437775611877 },
        "coincidencia": verdadero },
        { "probabilidades": { "0":
      0.9987999200820923, "1":
      0.001200090744532 : falso },
        "coincidencia": falso } , {




        "probabilidades":
          { "0": 0.029087694361805916,
          "1": 0.9709123373031616 },
        "coincidencia": verdadero

      }

] },
...
```

Ta-daaaaa! El código funciona muy bien. Cada índice de resultados corresponde al índice de la oración de entrada y diagnostica adecuadamente los dos insultos entre las tres oraciones.

Felicitaciones por ejecutar su primer modelo TensorFlow.js. Ahora que domina la IA, analicemos los pasos y los conceptos subyacentes de esta biblioteca.

Cargando el modelo

Cuando llamamos toxicidad.cargar, puede pensar que el modelo se está cargando en la memoria, pero solo tiene la mitad de la razón. La mayoría de estas bibliotecas no se envían con el modelo entrenado en el código base de JavaScript. Lea esa frase de nuevo. Esto puede parecer un poco alarmante para nuestros desarrolladores de NPM, pero tiene mucho sentido para nuestros usuarios de CDN. El método de carga activa una llamada de red para descargar el modelo que usa la biblioteca. En algunos casos, el modelo que se carga está optimizado para el entorno y el dispositivo donde se encuentra el JavaScript. Revise los registros de red ilustrados en la Figura 2-4.

Figura 2-4. Solicitudes de descarga de red

> Si bien el paquete Toxicity NPM se puede minimizar y comprimir a solo 2,4 KB, hay una carga útil adicional de varios megabytes en la red para el archivo del modelo real cuando se usa la biblioteca.

El método de carga para esta biblioteca de toxicidad toma un umbral que se aplicará a todas las clasificaciones posteriores y luego activa una llamada de red para descargar el archivo del modelo real. Cuando ese modelo se descarga por completo, la biblioteca carga el modelo en la memoria optimizada para tensores para su uso.

Es importante evaluar cada biblioteca adecuadamente. Revisemos algunas preguntas comunes que la gente hace cuando aprende un poco más sobre esto.

P: ¿Todos los modelos de TensorFlow.js realizan esta descarga?

Respuesta: No existe una regla que tenga que tener TensorFlow.js para importar el archivo modelo desde una URL. De hecho, puede sacar el archivo de una multitud de lugares. En el mundo de TensorFlow.js de nivel inferior, los archivos de modelo se colocan donde son más útiles. Sin embargo, es bastante común que las bibliotecas mantenidas por Google extraigan sus modelos de un único destino alojado para acceder a ellos cuando sea necesario.

P: ¿Dónde está alojado?

Respuesta: Debido a que es difícil garantizar actualizaciones y resolver la necesidad de un hospedaje de modelos efectivo, Google ha establecido un servicio de hospedaje de modelos para modelos comunitarios populares llamado TensorFlow Hub (TFHub). Puedes revisar todos los modelos en https:// t ub.dev.

Cuando nuestro código llama a .load, está realizando una solicitud a TensorFlow Hub para el modelo.
Si ayuda, puede pensar en esto como NPM para modelos ML.

P: ¿Por qué los modelos están en "fragmentos"?

Respuesta: Cuando un modelo tiene más de 4 MB, se divide en fragmentos de 4 MB para alentar al dispositivo a paralelizar las descargas y almacenar en caché el modelo localmente. Muchos cachés de navegadores móviles superan los 4 MB por archivo para una sesión determinada.

P: ¿Todos los modelos son de este tamaño?

Respuesta: No. Cada modelo tiene una variedad de tamaños, recortes y optimizaciones. He creado modelos de 48 KB y modelos de 20 MB. La toxicidad en el momento de escribir este artículo es de 28,06 MB,4 pero los modelos grandes a menudo se pueden optimizar significativamente.
Veremos cómo puede cambiar el tamaño de los modelos en capítulos posteriores. Es importante conocer y evaluar el costo de cada modelo y su efecto en el plan de internet del usuario.

P: ¿Puedo alojar el modelo yo mismo?

Respuesta: En muchos casos, puede hacerlo. La mayoría de las bibliotecas que no son de Google incluso lo alentarán a asumir su propio alojamiento de modelos y pueden depender de él por completo. Sin embargo, otros, como este, no te dejan opción. Este modelo de toxicidad solo se carga desde TensorFlow Hub. Afortunadamente, el software de código abierto significa que podemos solucionar esto modificando una bifurcación de la biblioteca de JavaScript.

4 La información del modelo de toxicidad está disponible en https://oreil.ly/Eejyi.

Clasificación

Lo siguiente que hizo nuestro código de toxicidad fue ejecutar un método de clasificación . Este es el momento en el que nuestras oraciones de entrada pasaron por el modelo y obtuvimos los resultados.
Si bien parecía tan simple como cualquier otra función de JavaScript, esta biblioteca en realidad ocultaba algunos procesos fundamentales que eran necesarios.

Todos los datos que entran y salen de un modelo se convierten en un tensor. Cubriremos los tensores con mayor detalle en el Capítulo 3, pero es importante tener en cuenta que esta conversión es esencial para la IA. Todas las cadenas de entrada se convierten, se realizan los cálculos y los resultados que surgen son tensores que se reconvierten en primitivas de JavaScript normales.

Es bueno que esta biblioteca haya manejado esto por nosotros. Cuando haya terminado con este libro, podrá envolver modelos de aprendizaje automático con la misma destreza. Podrá mantener a sus usuarios en una feliz ignorancia de las complejidades de las conversiones de datos que ocurren detrás de escena.

En el próximo capítulo, saltaremos a esa conversión. Comprenderá por completo la transición de datos a tensores y todos los superpoderes de manipulación de datos que conlleva.

Inténtalo tú mismo

Ahora que ha implementado un modelo, lo más probable es que pueda implementar los otros modelos proporcionados por Google. La mayoría de las páginas de GitHub de los otros modelos de Google tienen documentos LÉAME que explican cómo implementar cada biblioteca. Muchas de las implementaciones son similares a lo que vimos con Toxicidad.

Tómese un momento para explorar los modelos existentes y dejar volar su imaginación. Puede comenzar a trabajar con estas bibliotecas inmediatamente. Saber que existen estos modelos también será útil a medida que avance en este libro. No solo comprenderá mejor de lo que son capaces estas bibliotecas, sino que también querrá combinar e incluso mejorar estas bibliotecas existentes para sus necesidades.

En el próximo capítulo, comenzaremos a profundizar en los detalles de lo que ocultan estas bibliotecas bien envueltas para que podamos liberar sus habilidades de TensorFlow.js sin límite.

Revisión del capítulo

Configuramos su computadora para TensorFlow.js a través de algunas opciones de práctica común. Nos aseguramos de que nuestra máquina esté lista para ejecutar TensorFlow.js, e incluso desmontamos y ejecutamos un modelo empaquetado para determinar la toxicidad del texto.

Desafío del capítulo: ¡Alerta de camión!

Tómese el tiempo para probar el modelo MobileNet, que tiene la capacidad de mirar imágenes e intentar clasificar el artefacto predominante. A este modelo se le puede pasar cualquier elemento <img>, <video> o <canvas> y devuelve una matriz de las predicciones más probables de lo que ve en ese gráfico en particular.

El modelo MobileNet ha sido entrenado para clasificar 1.000 elementos posibles desde muros de piedra, hasta camiones de basura, hasta incluso un gato egipcio. La gente ha usado esta biblioteca para detectar una amplia variedad de cosas divertidas. Una vez vi un código que conectaba una cámara web a MobileNet para detectar llamas.

Para este desafío del capítulo, tiene la tarea de crear un sitio web que pueda detectar camiones. Dada una imagen de entrada, busca identificar si es un camión o no. Cuando detecte un camión a partir de una foto, emita una alerta ("¡CAMIÓN DETECTADO!"). De forma predeterminada, el paquete MobileNet devuelve las tres detecciones principales. Si alguno de esos tres ve un camión en la foto, su alerta debería notificar al usuario como en la Figura 2-5.

Figura 2-5. detector de camiones funcionando

Puede encontrar la respuesta a este desafío en el Apéndice B.

Preguntas de revisión

Revisemos las lecciones que hemos aprendido del código que ha escrito en este capítulo.
Tómese un momento para responder las siguientes preguntas:

1. ¿Se puede ejecutar TensorFlow normal en el navegador?
2. ¿TensorFlow.js tiene acceso a la GPU?
3. ¿Es necesario tener instalado CUDA para ejecutar TensorFlow.js?
4. Si no especifico una versión en un CDN, ¿qué sucede?
5. ¿Cómo identifica las infracciones el clasificador de toxicidad?
6. ¿Cuándo pasamos un umbral de toxicidad?
7. ¿El código de toxicidad contiene todos los archivos necesarios?
8. ¿Tenemos que hacer algún trabajo de tensor para usar esta biblioteca de toxicidad?

Las soluciones a estos ejercicios están disponibles en el Apéndice A.

Hasta la próxima…

En el Capítulo 3 , finalmente profundizará en la estructura más fundamental del aprendizaje automático, el tensor. Convertirá datos en tensores y volverá con un ejemplo del mundo real. Comprenderá cómo los tensores abandonan el mundo de los bucles iterativos y se convierten en el concepto de lotes optimizados de empoderamiento matemático.

3

ores

Hemos mencionado la palabra tensor varias veces y reside como la palabra predominante en TensorFlow.js, por lo que es hora de que sepamos cuáles son estas estructuras. Este capítulo crítico le brindará experiencia práctica con el concepto fundamental de administrar y acelerar

datos, que está en el corazón de enseñar máquinas con datos.

Lo haremos:

- Explicar el concepto y la terminología de los tensores
- Crear, leer y destruir tensores • Practicar conceptos de datos estructurados • Dar el salto a utilizar tensores para construir algo útil

Tómese su tiempo con este capítulo si es nuevo en tensores. Sentirse cómodo con este aspecto de los datos lo ayudará a sentirse cómodo con el aprendizaje automático en general.

¿Por qué tensores?

Vivimos en un mundo lleno de datos y, en el fondo, todos sabemos que termina en 1 y 0. Para muchos de nosotros, esto sucede mágicamente. Toma una foto con su teléfono y se crea un archivo binario complejo. Luego, desliza hacia arriba y hacia abajo, y nuestro archivo binario cambia de JPG a PNG en un instante. Miles de bytes desconocidos se generan y destruyen en microsegundos a medida que los archivos cambian de tamaño, se reformatean y, para usted,

niños, filtro. Ya no puedes ser mimado. A medida que te aventuras a tocar, sentir y alimentar datos, tienes que decir adiós a la felicidad ignorante.

Para citar la película Blade de 1998:

> Será mejor que te despiertes. El mundo en el que vives no es más que un aderezo recubierto de azúcar. Hay otro mundo debajo de él".

OK, es así pero no tan intenso. Para entrenar una IA, deberá asegurarse de que sus datos sean uniformes, y deberá comprenderlos y verlos. No estás entrenando a tu IA para que haga la tarea de decodificar archivos PNG y JPG de manera uniforme; lo estás entrenando en las versiones descodificadas e imitadas de lo que realmente hay en una foto.

Esto significa que las imágenes, la música, las estadísticas y cualquier otra cosa que esté usando en sus modelos de TensorÿFlow.js necesitan un formato de datos uniforme y optimizado. Idealmente, nuestros datos se convertirían en contenedores numéricos que escalan rápidamente y funcionan directamente con optimizaciones de cálculo en GPU o Web Assembly. Necesita algo limpio y directo para que nuestros datos informativos entren y salgan. Estos contenedores deben ser sin opinión para que puedan contener cualquier cosa. ¡Bienvenidos a los tensores!

> Comprender el uso y las propiedades de los tensores es un ejercicio continuo incluso para el experto más experto en TensorFlow.js. Si bien este capítulo sirve como una excelente introducción, no debe sentirse rezagado por tener dificultades con el uso de tensores. Este capítulo puede servir como referencia a medida que avanza.

Hola tensores

Los tensores son colecciones de datos en un tipo estructurado. No es nada nuevo que un marco convierta todo en números, pero podría ser un concepto nuevo darse cuenta de que depende de usted elegir cómo se forman los datos en última instancia.

Como se mencionó en el Capítulo 1, todos los datos deben destilarse en números para que las máquinas los entiendan. Los tensores son el formato preferido de información e incluso tienen pequeñas abstracciones para tipos no numéricos. Son como señales eléctricas del mundo físico al cerebro de nuestra IA. Si bien no hay una especificación de cómo se deben estructurar sus datos, debe ser constante para mantener sus señales organizadas para que nuestro cerebro pueda ver el mismo patrón una y otra vez. Las personas generalmente organizan sus datos en grupos, como arreglos y arreglos multidimensionales.

Pero, ¿qué es un tensor? Un tensor, como se define matemáticamente, es simplemente un conjunto estructurado de valores de cualquier dimensión. En última instancia, esto se resuelve en una agrupación optimizada de datos como números que están listos para el cálculo. Eso significa que, matemáticamente hablando, una matriz JavaScript tradicional es un tensor, una matriz 2D es un tensor y una matriz 512D es un tensor. Los tensores TensorFlow.js son la encarnación de estas estructuras matemáticas que contienen las señales aceleradas que alimentan datos dentro y fuera de un modelo de aprendizaje automático.

Si está familiarizado con las matrices multidimensionales en JavaScript, debería sentirse como en casa con la sintaxis de los tensores. A medida que agrega una nueva dimensión a cada matriz, a menudo se dice que está aumentando el rango de un tensor.

Repaso de las dimensiones de la

matriz Una matriz unidimensional (1D) es su conjunto plano estándar de datos. Probablemente trabaje con estos todo el tiempo en su JavaScript. Estas matrices de rango uno son ideales para capturar conjuntos de secuencias y datos relacionados.

```
[1, 2, 3, 4]
```

Una matriz bidimensional (2D) o de rango dos captura cuadrículas de datos relacionados. Por ejemplo, una matriz 2D puede almacenar las coordenadas X e Y de un gráfico.

```
[
  [2, 3],
  [5, 6],
  [8, 9]
]
```

Una matriz tridimensional (3D) sería de rango tres. El rango tres es el último rango comúnmente visual para nosotros, meros humanos. Es difícil visualizar más allá de las tres dimensiones.

```
[
  [
    [1, 2, 3], [4,
    5, 6] ], [


    [6, 5, 4], [3,
    2, 1]
  ]
]
```

En pocas palabras, una matriz 2D es una matriz de matrices, y una matriz 3D es una matriz de matrices de matrices. Entiendes la idea. Las matrices anidadas le permiten dar formato a los datos para que la información correlacionada pueda correlacionarse entre sí y puedan surgir patrones. Después de tres dimensiones, no puede graficar fácilmente los datos, pero puede utilizarlos.

Creación de tensores

Independientemente de cómo haya importado TensorFlow.js, el código de este libro supone que ha consolidado la biblioteca en una variable llamada tf, que se usará para representar TensorFlow.js en todos los ejemplos.

Para crear su primer tensor, mantendremos las cosas simples y lo construirá con una matriz 1D de Javaÿ Script (Ejemplo 3-1). La sintaxis y la estructura de la matriz se transfieren a los tensores.

Ejemplo 3-1. Creando tus primeros tensores

```
// creando nuestro primer tensor
const dataArray = [8, 6, 7, 5, 3, 0, 9] const first
= tf.tensor(dataArray)

// hace lo mismo const
first_again = tf.tensor1d(dataArray)
```

> tf.tensor crea un tensor 1D si se le pasa una matriz 1D. Crearía un tensor 2D si pasara una matriz 2D.
>
> tf.tensor1d crea un tensor 1D si se pasa una matriz 1D. Se produciría un error si pasara una matriz 2D.

Este código crea una estructura de datos de tensor 1D de siete números en la memoria. Ahora esos siete números están listos para manipulación, operaciones aceleradas o simplemente entrada. Sin embargo, estoy seguro de que notó que proporcionamos dos formas de realizar la misma acción.

El segundo método proporciona un nivel adicional de verificación de tiempo de ejecución ya que ha definido la dimensionalidad esperada. Determinar la dimensionalidad deseada es útil cuando busca asegurar la cantidad de dimensiones en los datos con los que está trabajando.
Existen métodos para verificar hasta seis dimensiones con tf.tensor6d.

En este libro, trabajaremos principalmente con el tf.tensor genérico, pero si se encuentra inmerso en un proyecto complejo, no olvide que puede ahorrarse el dolor de cabeza de recibir dimensiones inesperadas identificando explícitamente la dimensionalidad deseada. de un tensor.

Como nota adicional, mientras que los tensores del ejemplo 3-1 eran una matriz de números naturales, el tipo de datos predeterminado para almacenar números es Float32. Los números de coma flotante (es decir, números con decimales, por ejemplo, 2,71828) son bastante dinámicos e impresionantes. Por lo general, pueden manejar la mayoría de los números que necesitará y estar listos para los valores intermedios. A diferencia de las matrices de JavaScript, el tipo de datos de un tensor debe ser homogéneo (todos del mismo tipo). Estos tipos pueden ser solo Float32, Int32, bool, complex64 o string, sin mezclarlos.

Si desea hacer cumplir que su tensor se crea como un tipo particular, no dude en utilizar el tercer parámetro de la función tf.tensor , que define explícitamente la estructura de tipos del tensor.

```
// creando un tensor 'float32' (predeterminado) const
first = tf.tensor([1.1, 2.2, 3.3], null, 'float32')

// un tensor 'int32' const
first_again = tf.tensor([1, 2, 3], null, 'int32')

// tipo inferido para constante
booleana the_truth = tf.tensor ([true, false, false])

// Adivina qué hace esto
const adivinar = tf.tensor([verdadero, falso, falso], nulo, 'int32')

// ¿Qué pasa con esto?
const adivinar_de nuevo = tf.tensor ([1, 3.141592654, falso])
```

Este tensor se crea como un tensor Float32 . El tercer parámetro era redundante en este caso.

El tensor resultante es Int32, y sin el tercer parámetro, habría sido un Float32.

El tensor resultante es un tensor booleano.

El tensor resultante es un tensor Int32, con los valores booleanos convertidos en 0 para falso y 1 para verdadero. Entonces, la variable adivinar contiene los datos [1, 0, 0].

Puede pensar que esta matriz salvaje generará un error, pero cada uno de los valores de entrada se convierte a su Float32 correspondiente con los datos de tensor resultantes [1, 3.1415927, 0].

¿Cómo puedes identificar el tipo de tensor que se creó? Al igual que cualquier matriz de JavaScript, los tensores están equipados con métodos para explicar sus propiedades. Las propiedades útiles incluyen longitud (tamaño), dimensionalidad (rango) y tipo de datos (dtype).

Apliquemos lo aprendido:

```
const segundo = tf.tensor1d([8, 6, 7, 5, 3, 0, 9])

// ¡Vaya!
prueba
   { constante no = tf.tensor1d ([[1],[2]])
} catch (e)
   { console.log("Ese es un Ghost Rider negativo")
}
```

```
console.log("Clasificación:",
segundo.rango) console.log ("Tamaño:",
segundo.tamaño) console.log(" Tipo de datos :", segundo.tipo)
```

Esto crea un tensor exitoso. Debe conocer el tipo de datos, la dimensión y el tamaño.

Dado que está utilizando tensor1d para crear un tensor de rango dos, esto arrojará y hará que se ejecute catch y registre un mensaje.

La matriz simple es de rango uno, por lo que imprimirá 1.

El tamaño es la longitud de la matriz e imprimirá 7.

El tipo de datos del tensor de una matriz de números imprimirá float32.

¡Felicitaciones por crear sus primeros tensores! Es seguro decir que ser un maestro de los tensores es el núcleo de la domesticación de datos para TensorFlow.js. Estos cubos estructurados de valores son la base para introducir y sacar datos del aprendizaje automático.

Tensores para ejercicios de datos

Digamos que quieres hacer una IA para jugar tres en raya (tres en raya para mis amigos al otro lado del estanque). Como siempre con los datos, es hora de tomar un café o té y pensar en la forma correcta de convertir datos del mundo real en datos de tensor.

Puede almacenar imágenes de juegos, cadenas de tutoriales o simplemente las X y Os del juego. Las imágenes y los tutoriales serían bastante impresionantes, pero por ahora, consideremos la idea de almacenar el estado de un tablero de juego. Solo hay nueve casillas posibles para jugar, por lo que una matriz simple de nueve valores debería representar cualquier estado dado del tablero.

¿Deberían leerse los valores de izquierda a derecha y de arriba hacia abajo? Rara vez importa mientras seas consistente. Todas las codificaciones están compuestas. Sin embargo, tenga en cuenta que un tensor se resuelve en números. Esto significa que si bien puede almacenar las cadenas "X" y "O", de todos modos tendrían que convertirse en números. Almacenemos nuestras X y Os asignándolas a algún tipo de valor numérico que tenga sentido. ¿Eso significa que solo asignas uno de ellos a 0 y el otro a 42? Estoy seguro de que puedes encontrar una estrategia que refleje adecuadamente el estado del juego.

Evaluemos el estado de un juego activo para un ejercicio. Tómese un momento para revisar la cuadrícula de un partido en curso, como se muestra en la Figura 3-1. ¿Cómo podría convertirse esto en tensores y números?

Figura 3-1. Un juego con datos

Quizás el tablero que se muestra aquí podría leerse y representarse como un tensor unidimensional. Puede leer los valores de izquierda a derecha, de arriba a abajo. En cuanto a los números, elijamos -1, 0 y 1 para representar los tres valores posibles para cualquier cuadrado. La Tabla 3-1 muestra la búsqueda para cada valor posible.

Tabla 3-1. Tabla de valor a número

	1
X	-1
O	0
Vacío	

Esto crearía un tensor así: [1, 0, 0, 0, -1, 0, 1, 0, 0]. O bien, crearía un tensor 2D, así: [[1, 0, 0],[0, -1, 0],[1, 0, 0]].

Ahora que tiene un objetivo, escribamos un código para convertir el tablero en un tensor. Incluso exploraremos los parámetros adicionales de la creación de tensores.

```
// Este código crea un tensor 1D `Float32` const a
 = tf.tensor([1, 0, 0, 0, -1, 0, 1, 0, 0])

// Este código crea un tensor 2D `Float32` const b
 = tf.tensor([[1, 0, 0],[0, -1, 0],[1, 0, 0]])

// Esto hace lo mismo que el anterior pero con una matriz // de
 entrada 1D que se convierte en un tensor 2D `Float32` const c =
 tf.tensor([1, 0, 0, 0, -1, 0, 1, 0, 0], [3, 3])
```

```
// Este código convierte la matriz de entrada 1D en un tensor 2D Int32
const d = tf.tensor([1, 0, 0, 0, -1, 0, 1, 0, 0], [3, 3], 'int32 ')
```

El segundo parámetro de un tensor puede identificar la forma deseada de los datos de entrada. Aquí, convierte la matriz 1D en un tensor 2D especificando que desea que los datos tengan una estructura de rango dos como 3 x 3.

El tercer parámetro del tensor identifica el tipo de datos que le gustaría usar sobre el tipo de datos inferido. Dado que está almacenando números redondos, puede especificar el tipo int32. Sin embargo, el rango del tipo predeterminado float32 es enorme y puede manejar cómodamente nuestros números.

Los tensores son para siempre... un poco

Identificamos cómo especificar el tamaño y el tipo de datos de un tensor cuando se crea y, por supuesto, puede editar el código, pero una vez que se crea un tensor, está atascado con él. Supongamos que una biblioteca que no administramos nos entrega un tensor de Float32s, y realmente necesitábamos que el tipo fuera Int32s; ¿Qué puedes hacer al respecto? La noticia agridulce es que los tensores son inmutables. Si bien no puede modificar este tensor, puede crear fácilmente un nuevo tensor con el tipo y los datos correctos. Para esto, puede usar asType.

Por ejemplo, convierta un tensor así:

```
constante no = tf.tensor ([4], null, 'float32') constante
sí = nope.asType('int32')
```

La variable yep es un nuevo tensor Int32 con el valor 4. El tensor nope todavía existe sin cambios.

Es importante tener en cuenta que estas conversiones son rápidas y sucias. Si tiene el valor 3,9999 y lo convierte a Int32, se convierte en 3. No tiene lógica llevar el valor al Int32 más cercano. Simplemente elimina la parte decimal como Math.floor de JavaScript. Los tensores booleanos cambian a 0 y 1, y los tensores de cadena tendrán un error total.
Si está convirtiendo el tipo de datos de un tensor, asegúrese de estar listo para los resultados.

Cuando está creando tensores para representar datos, depende de usted decidir cómo está formateando los datos de entrada y cuál debería ser la estructura de tensor resultante. A medida que comprende los conceptos de aprendizaje automático, siempre está perfeccionando su intuición sobre qué tipo de datos funcionan mejor.

Volveremos a este problema del tres en raya más adelante en este libro.

Tensores de gira

Vamos a profundizar en los tensores a medida que avanza el libro, por lo que es esencial tomarse un momento y discutir por qué son tan importantes. Sin comprender la magnitud de los cálculos que estamos aprovechando, es difícil comprender los beneficios de dejar la seguridad de las variables y el motor de JavaScript familiar para las matemáticas antiguas.

Los tensores brindan velocidad

Ahora que sabe que puede crear tensores y representar datos como tensores, ¿cuál es el beneficio de realizar esta conversión? Hemos mencionado que los cálculos con tensores están optimizados por el marco TensorFlow.js. Cuando convierte matrices de números de JavaScript en tensores, puede realizar operaciones de matriz a velocidades vertiginosas, pero ¿qué significa eso realmente?

Las computadoras son excelentes para hacer un solo cálculo, y existen beneficios al realizar agrupaciones masivas de cálculos. Los tensores están diseñados para una inmensa cantidad de cálculos en paralelo. Si alguna vez ha realizado cálculos matriciales y vectoriales a mano, puede comenzar a apreciar los beneficios de los cálculos acelerados.

Los tensores proporcionan acceso directo

Sin el aprendizaje automático, aún puede usar tensores para crear gráficos 3D, sistemas de recomendación de contenido y hermosos sistemas de funciones iteradas (IFS) como el triángulo de Sier-piÿski ilustrado en la Figura 3-2.

Figura 3-2. Ejemplo IFS: el triángulo de Sierpiÿski

Hay muchas bibliotecas para imágenes, sonido, modelos 3D, video y más.
Todos ellos tienen una cosa en común. A pesar de todos los formatos que existen, las bibliotecas te dan datos en un formato universal. Los tensores son como ese formato de datos sin procesar y desenrollado, y con ese acceso puede construir, leer o predecir cualquier cosa que desee.

Incluso puede usar estas estructuras avanzadas para modificar los datos de la imagen (comenzará a hacerlo en el Capítulo 4). Comenzarás a divertirte más con las funciones de tensor después de que te hayas graduado de los conceptos básicos.

Datos de lote de tensores

En el ámbito de los datos, es posible que se encuentre recorriendo montañas de datos y preocupándose de que los editores de texto se bloqueen. Los tensores están optimizados para el procesamiento por lotes a altas velocidades. El pequeño proyecto al final de este capítulo tiene solo cuatro usuarios para simplificar las cosas, pero cualquier entorno de producción debe estar preparado para manejar cientos de miles.

La mayoría de los beneficios de los tensores se reconocerán cuando solicite a modelos capacitados que realicen los cálculos para predecir operaciones similares a las humanas en milisegundos. Comenzará a ver ejemplos de esto ya en el Capítulo 5. Hemos identificado que los tensores son estructuras impresionantes que brindan mucha aceleración y poder matemático a JavaScript, por lo que tiene sentido que use comúnmente este beneficioso estructura en lotes.

Tensores en la memoria

La velocidad del tensor viene con un costo general. Por lo general, cuando terminamos con una variable en JavaScript, la memoria se elimina limpiamente cuando se completan todas las referencias a esa variable. Esto se llama detección y recolección automática de basura (AGDC), y sucede sin que la mayoría de los desarrolladores de JavaScript comprendan o se preocupen por cómo funciona.
Sin embargo, sus tensores no reciben el mismo tipo de cuidado automático. Persisten mucho después de que se haya recopilado la variable que los utiliza.

Desasignación de tensores

Debido a que los tensores sobreviven a la recolección de elementos no utilizados, se comportan de manera diferente al JavaScript estándar y deben contabilizarse y desasignarse manualmente. Incluso si una variable se recolecta como basura en JavaScript, el tensor asociado queda huérfano en la memoria.
Puede acceder al conteo y tamaño actuales usando tf.memory(). Esta función devuelve un objeto con un informe de tensores activos.

El código del ejemplo 3-2 ilustra la memoria de tensor no recopilada.

Ejemplo 3-2. Tensores dejados en la memoria

```
/* Comprueba el número de tensores en la
 memoria * y el tamaño de la huella.
 * Ambos registros deben ser cero. */
console.log(tf.memory().numTensors)
console.log(tf.memory().numBytes)
```