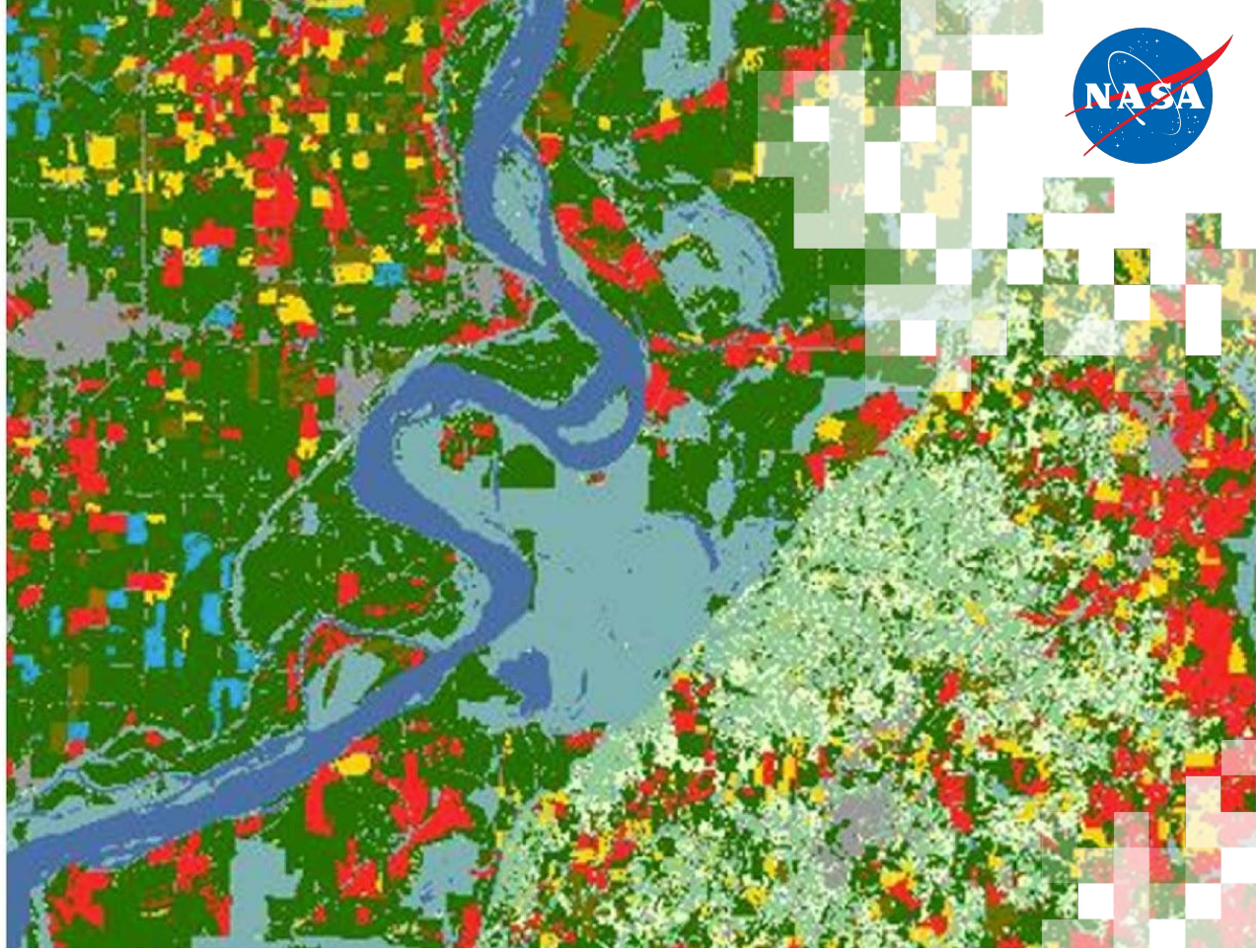
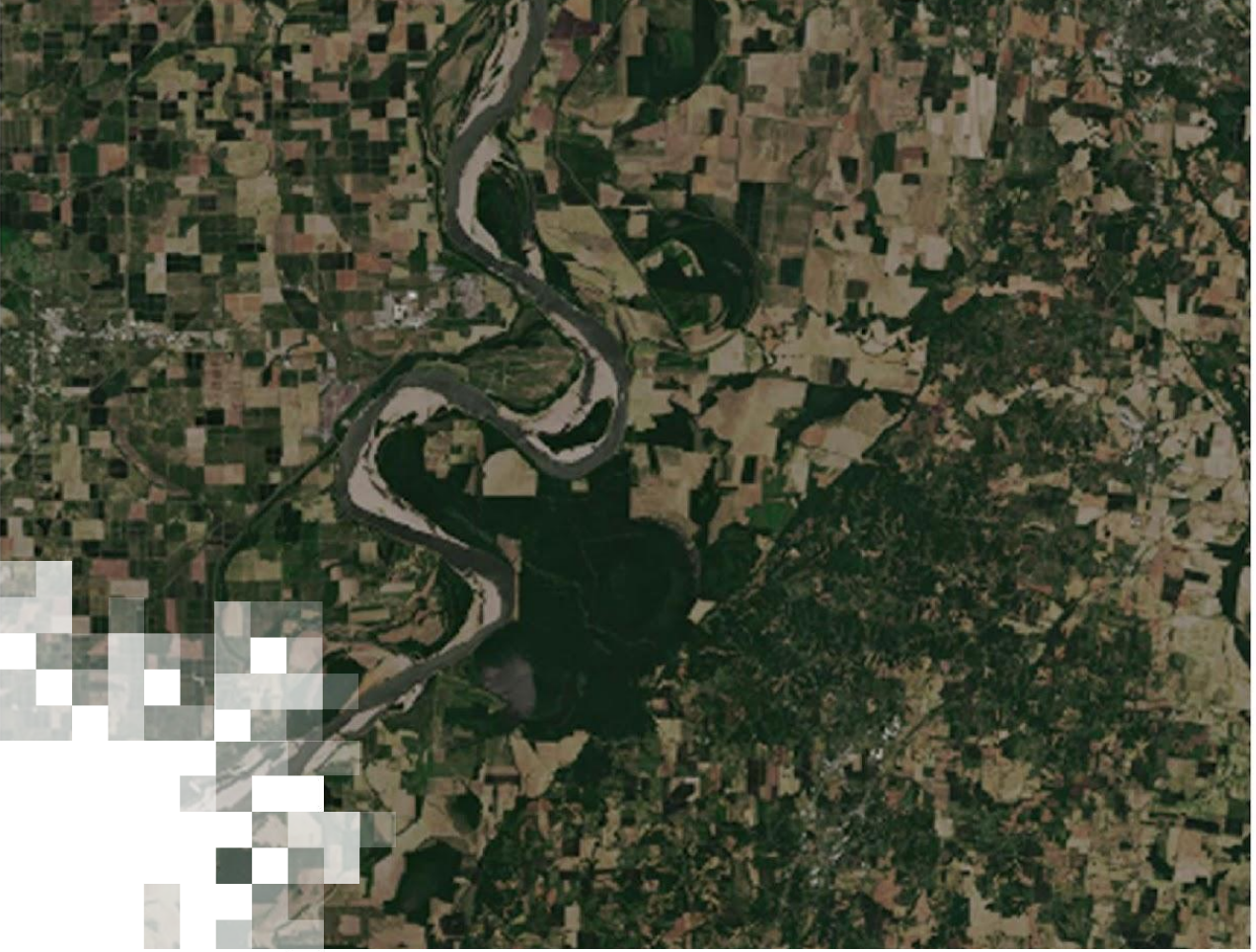


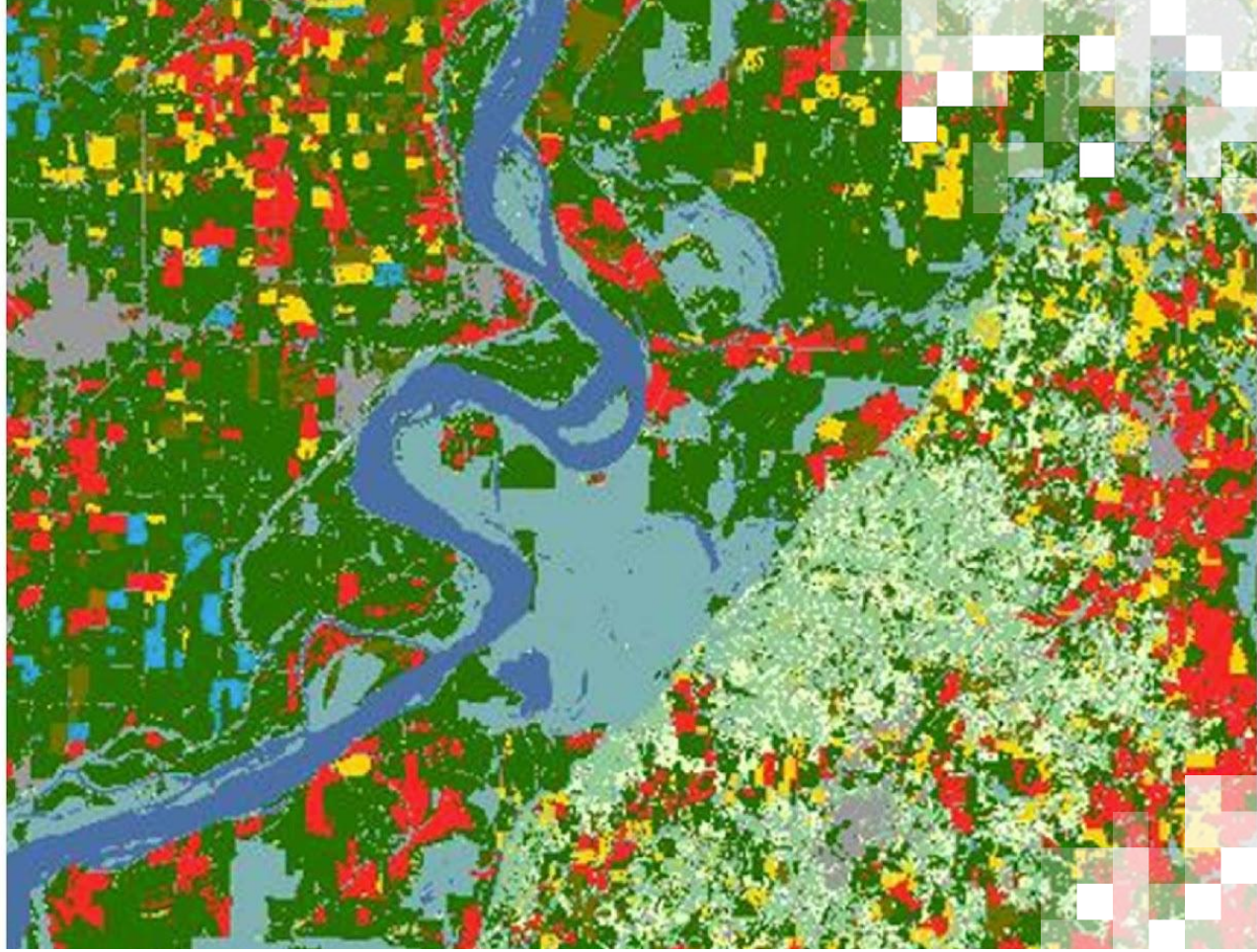
Aplicaciones de Aprendizaje Automático a Gran Escala usando Teledetección para la Formulación de Soluciones Agrícolas

3ra Parte: Entrenamiento y Prueba de Modelos de Aprendizaje Automático para Series Temporales de Imágenes Espaciadas Irregularmente

John Just (Deere y Cia., Universidad Estatal de Iowa), Erik Sorensen (Deere y Cia.)

19 de marzo de 2024





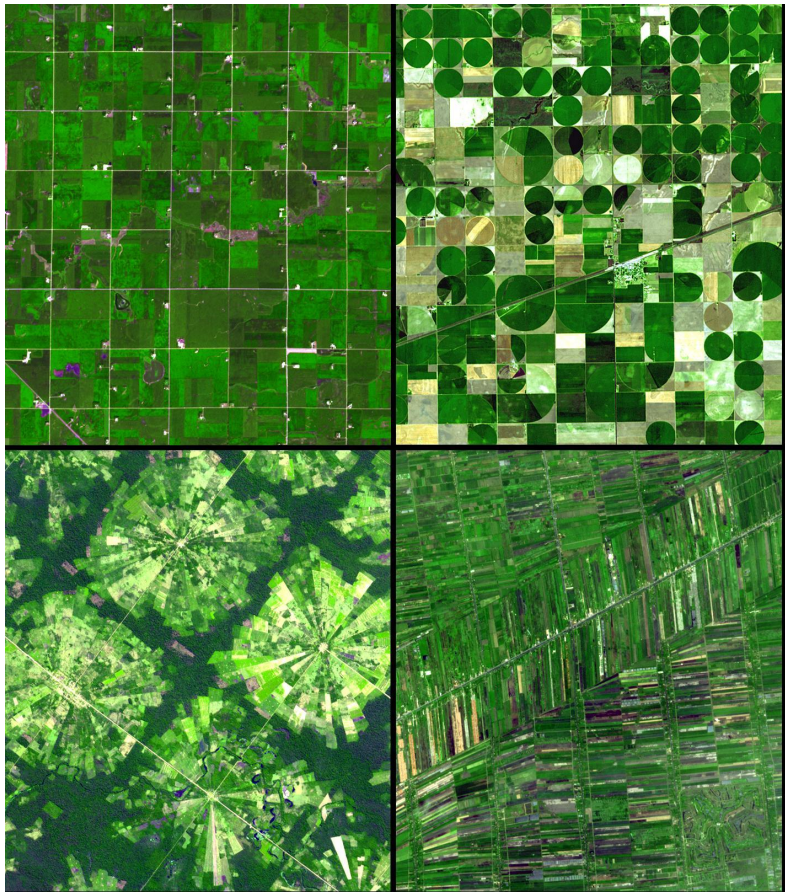
Aplicaciones de Aprendizaje Automático a Gran Escala usando Teledetección para la Formulación de Soluciones Agrícolas

Resumen General

Motivación para Esta Capacitación

- Los mapas oportunos y precisos de los cultivos de la temporada, a escala local y regional, son cruciales para la toma de decisiones y la gestión agrícola. Las series temporales espaciadas irregularmente son comunes en las imágenes satelitales ópticas. El entrenamiento robusto de modelos con datos de teledetección a menudo requiere datos muy grandes, pero el procesamiento y el entrenamiento son complejos.

- La capa Cropland Data Layer* (CDL, USDA–NASS) solo proporciona estimaciones de los tipos de cultivos que se dan a conocer al público unos meses después del final de la temporada de crecimiento, y no su secuencia o cronología (por ejemplo, para cultivos dobles)



Montaje de imágenes mostrando diferencias en la geometría y el tamaño de los campos agrícolas en diferentes partes del mundo. Fuente de la imagen: NASA (Instrumento: Terra – ASTER)

*Capa de datos de tierras de cultivo, en inglés

Objetivos de Aprendizaje para Esta Capacitación

Al final de esta serie, las/los participantes habrán desarrollado la capacidad para:

- Utilizar las técnicas recomendadas para descargar y procesar datos de teledetección de Sentinel-2 y la capa Cropland Data Layer (CDL) a gran escala (> 5 GB) con herramientas en la nube (Amazon Web Services [AWS] Simple Storage Service [S3], Databricks, Spark/Pyspark, Parquet)
  Producir gráficos interactivos de mapas, tablas, series temporales etc. para la investigación y verificación de datos y modelos.
  Filtrar datos de los dominios medidos (imágenes satelitales) y el objetivo (CDL) para cumplir con los objetivos de modelación basado en factores de calidad, clasificación de tierras, superposición de áreas de interés (AOI, por sus siglas en inglés) y ubicación geográfica.
  Crear canalizaciones de entrenamiento en TensorFlow para entrenar algoritmos de aprendizaje automático en conjuntos de datos geoespaciales/de teledetección a gran escala para el monitoreo agrícola
  Utilizar técnicas de muestreo aleatorio para crear solidez en un algoritmo predictivo y, al mismo tiempo, evitar la fuga de información en las divisiones de entrenamiento/validación/prueba

Prerrequisitos

- Fundamentos de la Percepción Remota (Teledetección)
- Clasificación de Cultivos con Series Temporales, 2da Parte
- Inscribirse y acceder a Databricks Community Edition

Esquema de la Capacitación

1ra Parte

Preparación de Datos para Imágenes y Etiquetas para la Modelación con Aprendizaje Automático a Gran Escala

5 de marzo de 2024

2da Parte

Cargadores de Datos para Entrenar Modelos de Aprendizaje Automático sobre Series Temporales de Imágenes Espaciadas Irregularmente

12 de marzo de 2024

3ra Parte

Entrenamiento y Prueba de Modelos de Aprendizaje Automático para Series Temporales de Imágenes Espaciadas Irregularmente

19 de marzo de 2024

Tarea

Abre el 19 de marzo – **Fecha Límite: 1ro de abril** – Publicada en la Página Web de la Capacitación

Se otorgará un certificado de finalización de curso a quienes asistan a todas las sesiones en vivo y completen la tarea asignada antes de la fecha estipulada.

Cómo Hacer Preguntas

- Por favor escriba sus preguntas en la casilla denominada “Questions” y las responderemos al final de este webinar.
- No dude en escribir sus preguntas mientras vayamos avanzando. Intentaremos responder todas las preguntas durante la sesión para preguntas y respuestas después del webinar.
- Las demás preguntas las responderemos en el documento de preguntas y respuestas, el cual será publicado en la página web de la capacitación aproximadamente una semana después de esta.

3ra Parte – Formadores

John Just

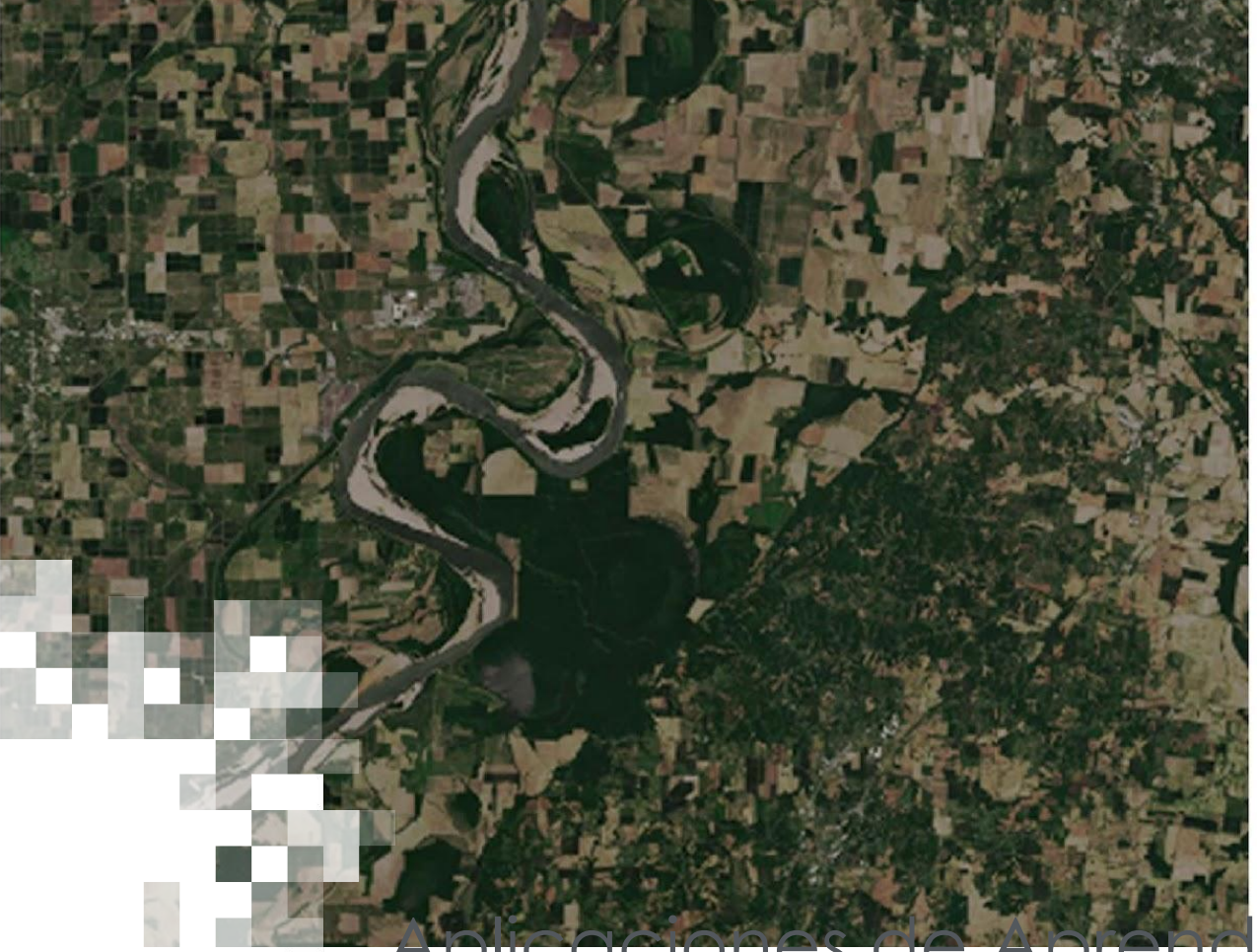
Científico Informático Principal

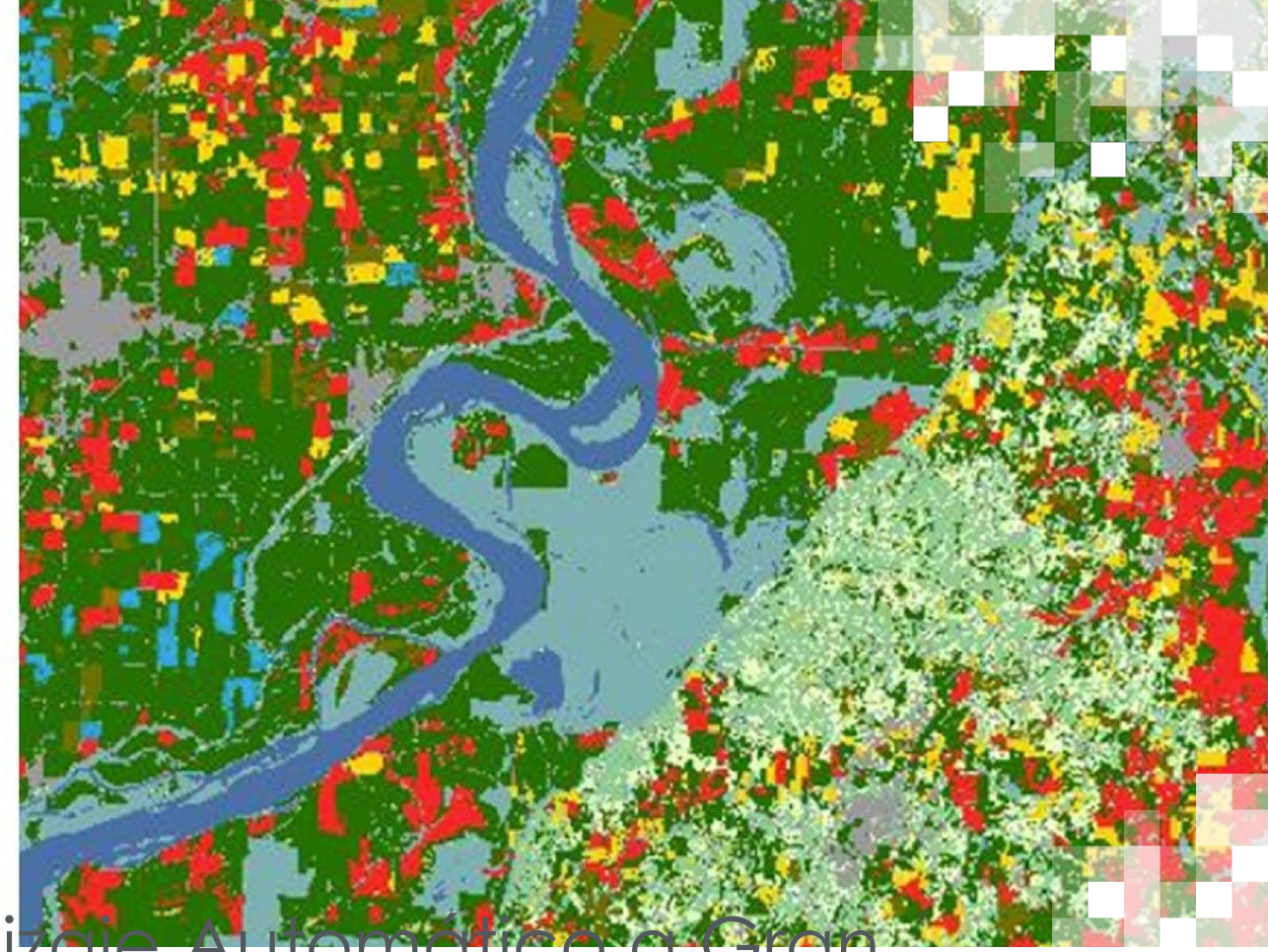
John Deere

Erik Sorensen

Científico Informático Sénior

John Deere





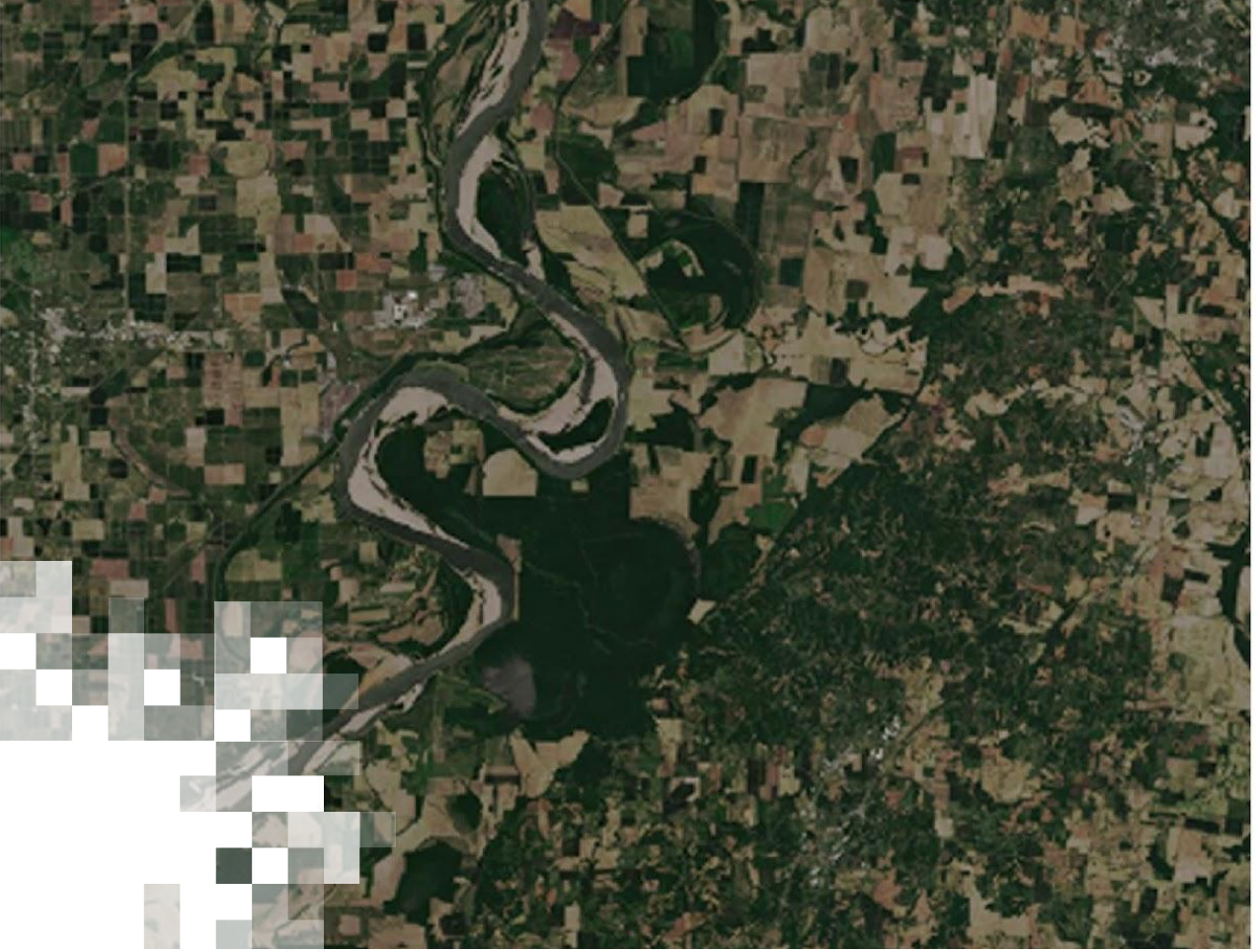
Aplicaciones de Aprendizaje Automático a Gran Escala usando Teledetección para la Formulación de Soluciones Agrícolas

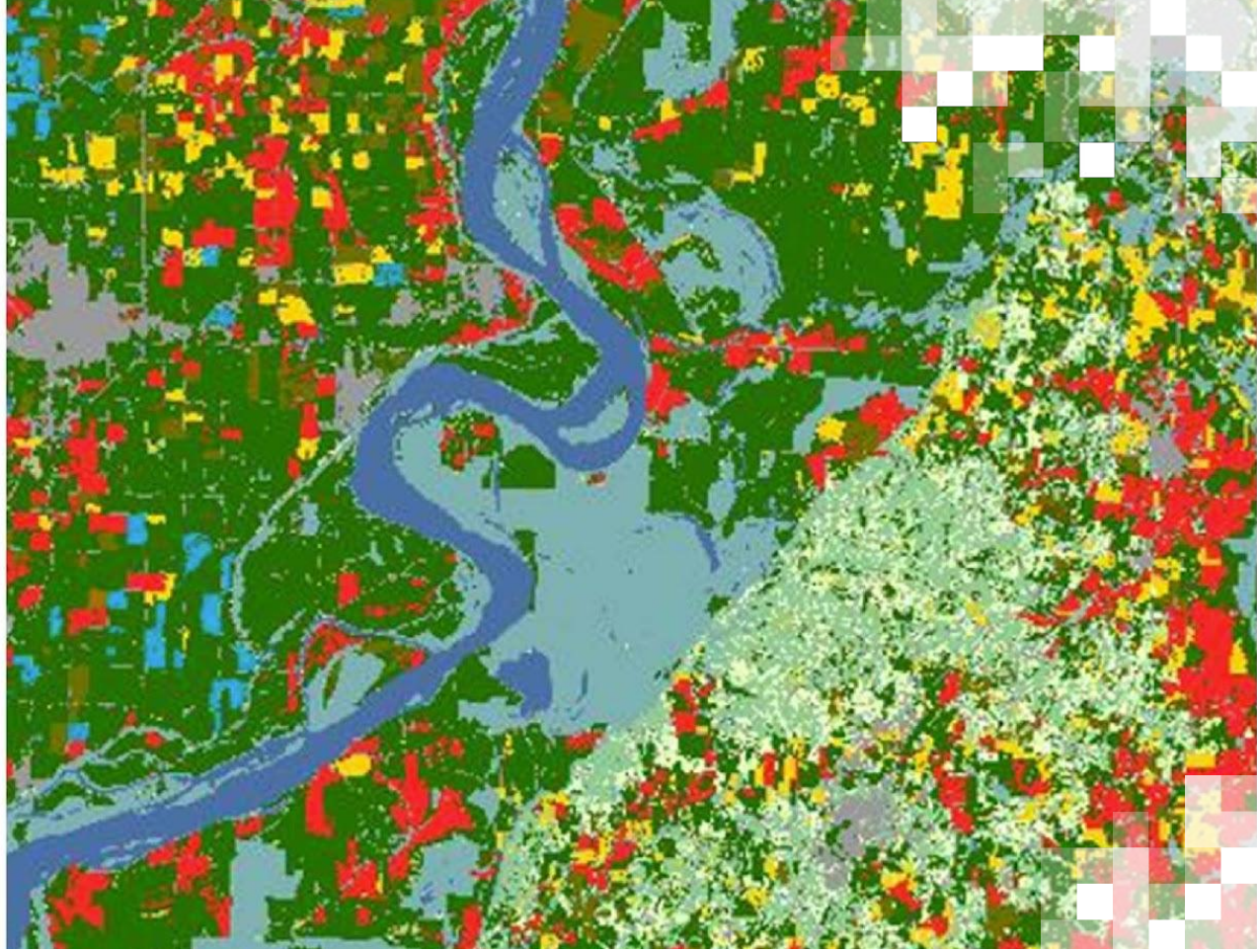
3ra Parte: Entrenamiento y Prueba de Modelos de Aprendizaje Automático para Series Temporales de Imágenes Espaciadas Irregularmente

3ra Parte - Objetivos

Al final de la 3ra Parte, las/los participantes podrán usar Python en Databricks Community Edition (o cualquier otro entorno de Python) para hacer lo siguiente:

- Configurar y entrenar un modelo de una red neuronal convolucional unidimensional (1-D convolutional neural network o CNN) que aprende a detectar tipos de cultivos a partir de una imagen satelital.
- Supervisar el rendimiento del modelo durante el entrenamiento y cómo elegir los ajustes de hiperparámetros adecuados.
  Visualizar el resultado del modelo de varias maneras para validar el rendimiento después del entrenamiento.



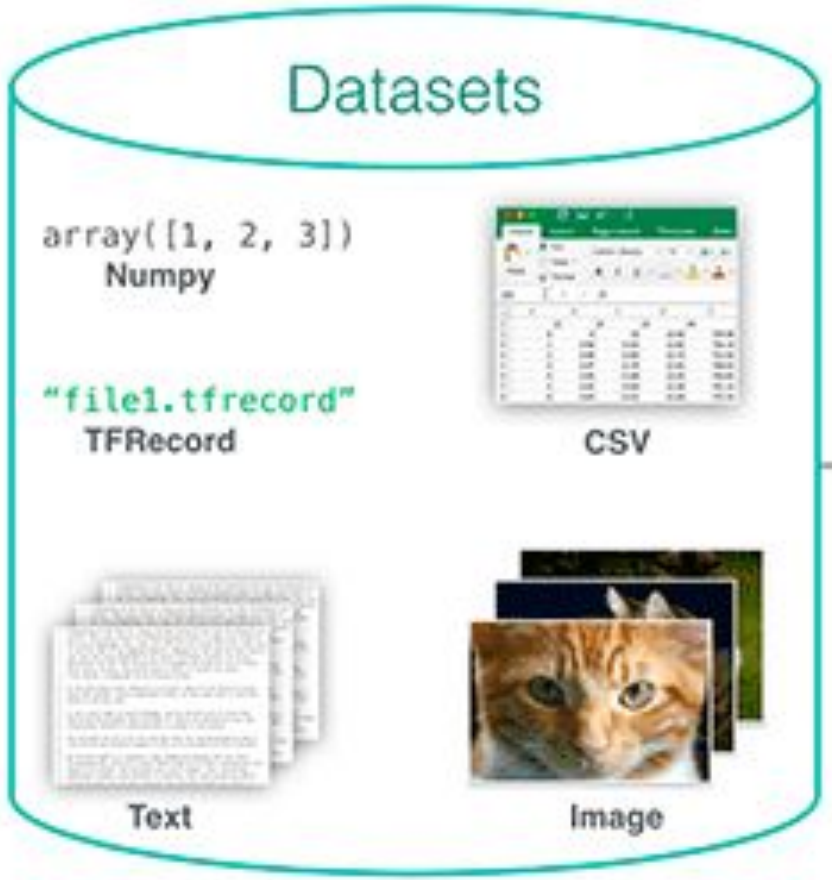


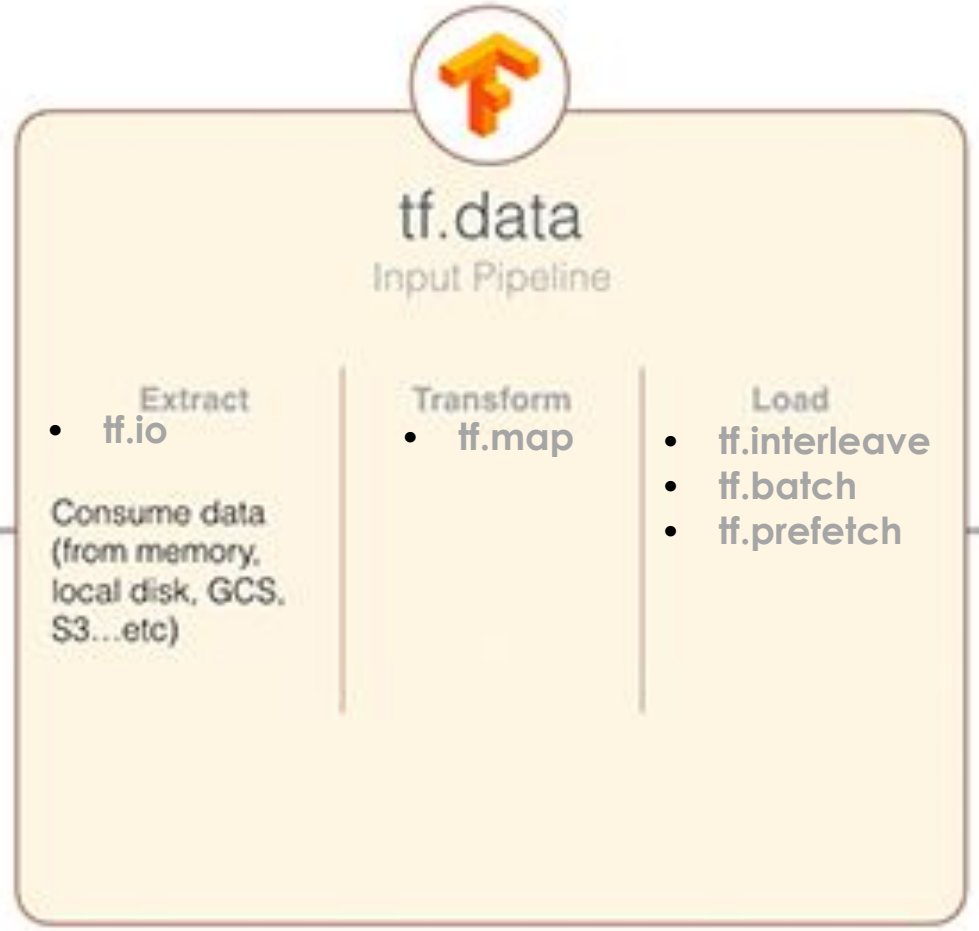
3ra Parte, Sección 1:

Construir una Red Neuronal Convolucional Unidimensional (temporal) con Keras

Flujo de Datos

Nos encontramos aquí





1ra Parte (construir un conjunto de datos)

2da Parte (construir un cargador de datos)

3ra Parte (entrenamiento del modelo e inferencia)

Módulos de Keras

Models API

- The Sequential class
- Model training APIs
- Saving & serialization

Losses

- Probabilistic losses
- Regression losses

Layers API

- Layer activations
- Layer weight initializers
- Convolution layers
- Pooling layers
- Normalization layers
- Regularization layers
- Reshaping layers
- Activation layers

Callbacks API

- ModelCheckpoint
- BackupAndRestore
- TensorBoard
- EarlyStopping
- LearningRateScheduler
- LambdaCallback

Optimizers

- SGD
- RMSprop
- Adam

Metrics

- Accuracy metrics
- Probabilistic metrics
- Regression metrics
- Classification metrics based on True/False positives & negatives
- Image segmentation metrics

Modelo para un 1D CNN* Simple de Keras

- model = Sequential()
- model.add(Conv1D(filters=3, kernel_size=5, input_shape=(20, 12)))
- model.add(MaxPooling1D(pool_size=5))
- model.add(Flatten())
- model.add(Dense(10, activation='relu'))
- model.add(Dense(num_classes, activation='softmax'))

model.summary()

Precaución: Keras hará algunas suposiciones sobre las capas de su modelo. Por ejemplo, aquí se asumió que uno no quiere ningún relleno.

Layer (type)	Output Shape	Param #
conv1d_1 (Conv1D)	(None, 16, 3)	183
max_pooling1d_1 (MaxPooling1D)	(None, 3, 3)	0
flatten_1 (Flatten)	(None, 9)	0
dense_1 (Dense)	(None, 10)	100
dense_2 (Dense)	(None, 6)	66

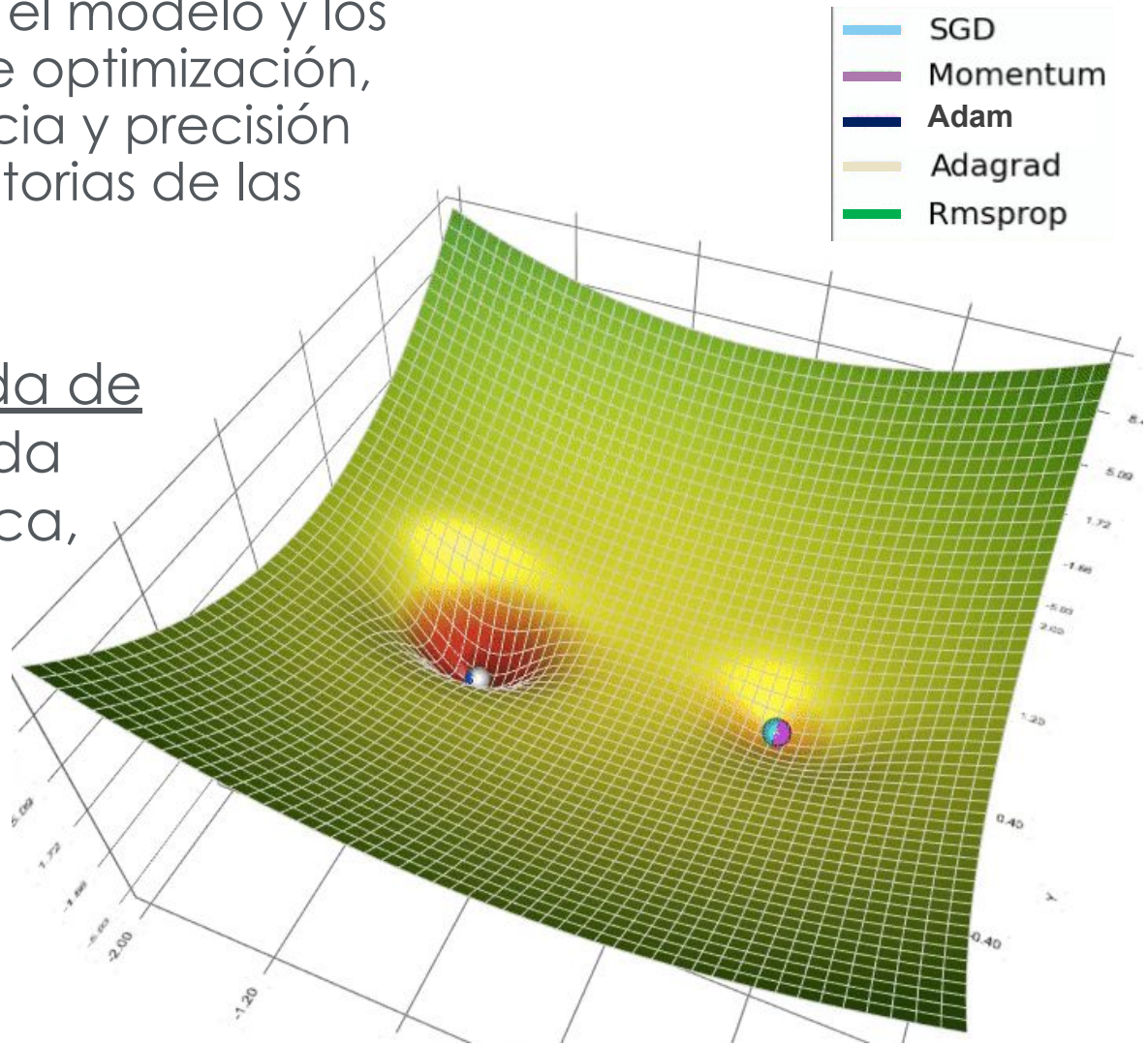
Total params: 349
Trainable params: 349
Non-trainable params: 0

*modelo de una red neuronal convolucional unidimensional, por sus siglas en inglés

Pérdidas y Optimizadores

En términos matemáticos, la función de pérdida, el modelo y los datos determinan la estructura de la variedad de optimización, mientras que el optimizador determina la eficiencia y precisión con la que se encuentra el punto óptimo (trayectorias de las canicas).

Para esta demostración utilizamos una pérdida de clasificación multiclase probabilística conocida como pérdida de entropía cruzada categórica, y el optimizador Adam.



Credit: Medium

Entrenamiento y Prueba

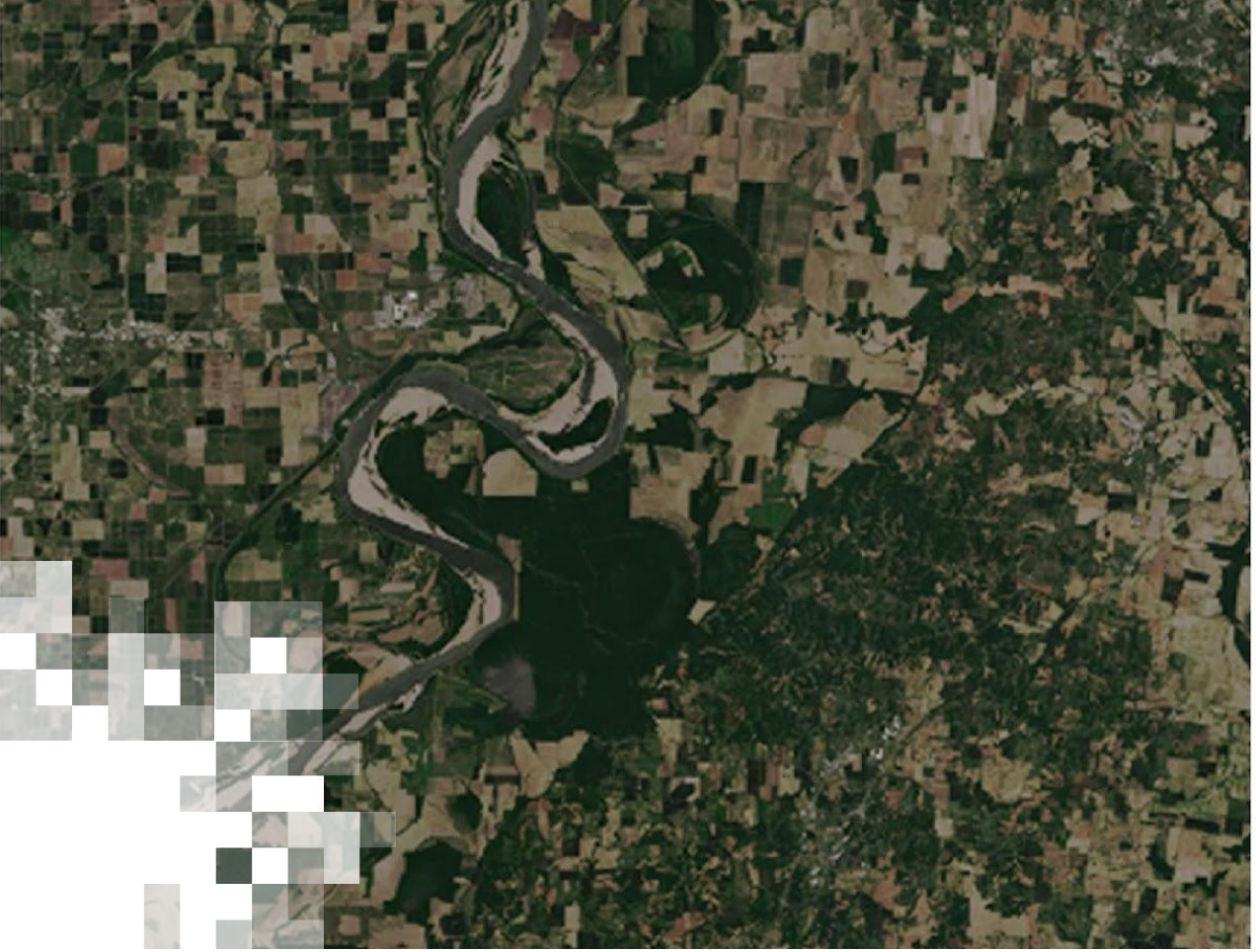
El entrenamiento es bastante fácil con Keras, puesto que gran parte de la funcionalidad que necesitamos ya está incorporada y simplemente la llamamos "model.fit". Sin embargo, tenemos que especificar algunas cosas:

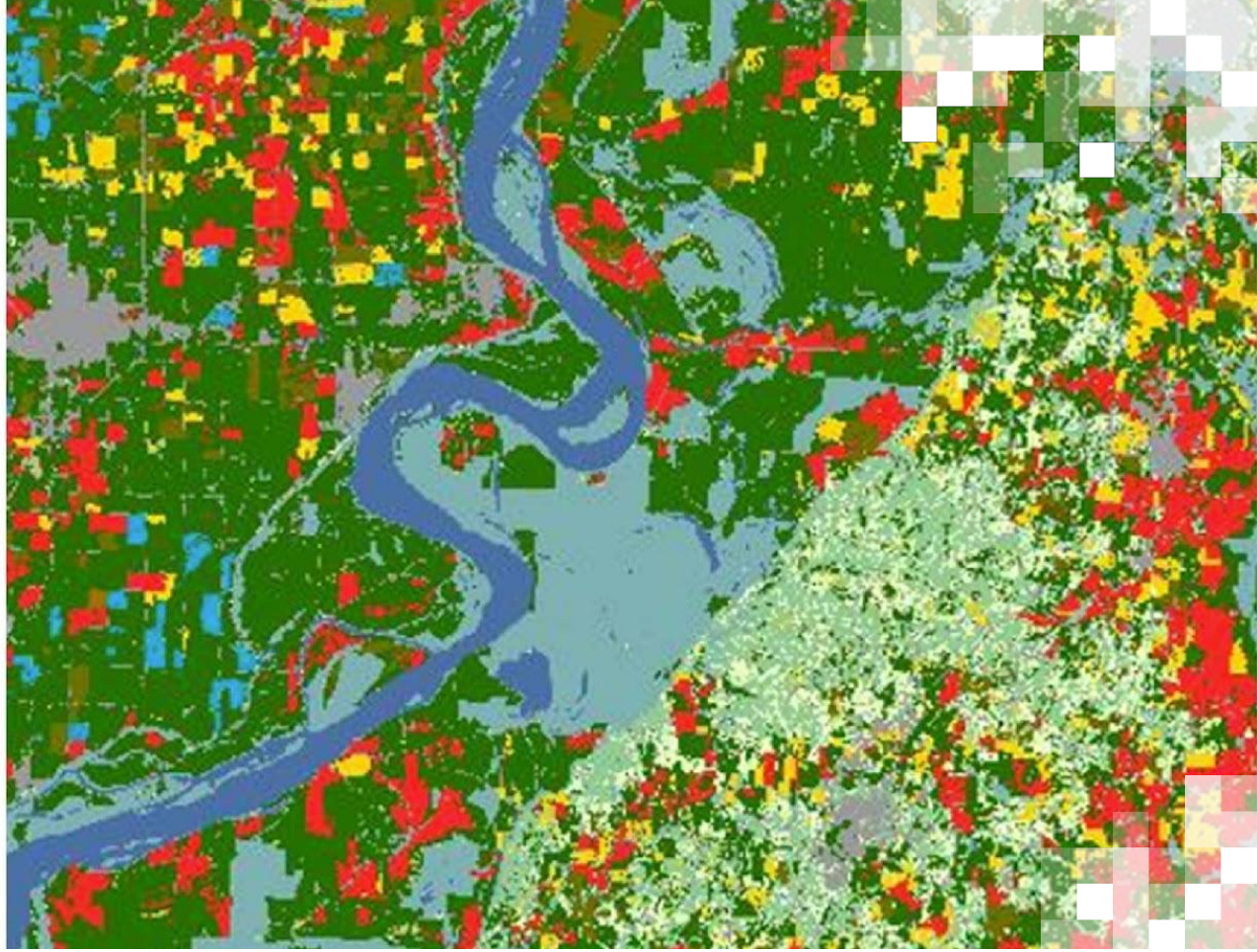
- Los conjuntos de datos de entrenamiento y validación (lo que construimos en la "2da Parte")
  Función de pérdida, optimizador y métricas
  Callbacks(en este caso, TensorBoard y EarlyStopping)

```
#Callbacks
early_stopping = tf.keras.callbacks.EarlyStopping(monitor='val_loss', patience=ES_PATIENCE, mode='min')
tb_callback = tf.keras.callbacks.TensorBoard('/tmp/tensorboard/', update_freq=1)

#Assign loss, optimizer, and metrics to the model
model.compile(loss=tf.keras.losses.CategoricalCrossentropy(),
              optimizer=tf.keras.optimizers.Adam(),
                  metrics=[tf.keras.metrics.CategoricalAccuracy()])

#Run training/optimization routine to fit model to data
history = model.fit(train_ds,
                    epochs=MAX_EPOCHS,
                          validation_data=val_ds,
                    callbacks=[early_stopping, tb_callback],
                          verbose=1)
```





3ra Parte, Sección 2:
TensorBoard

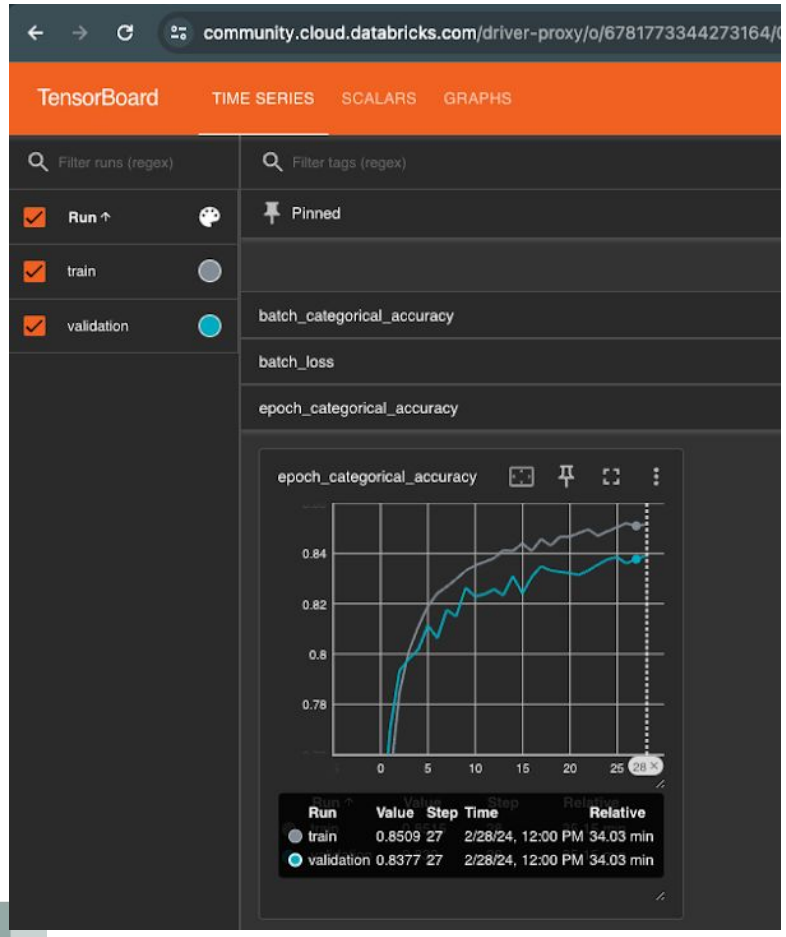
TensorBoard – Resumen General

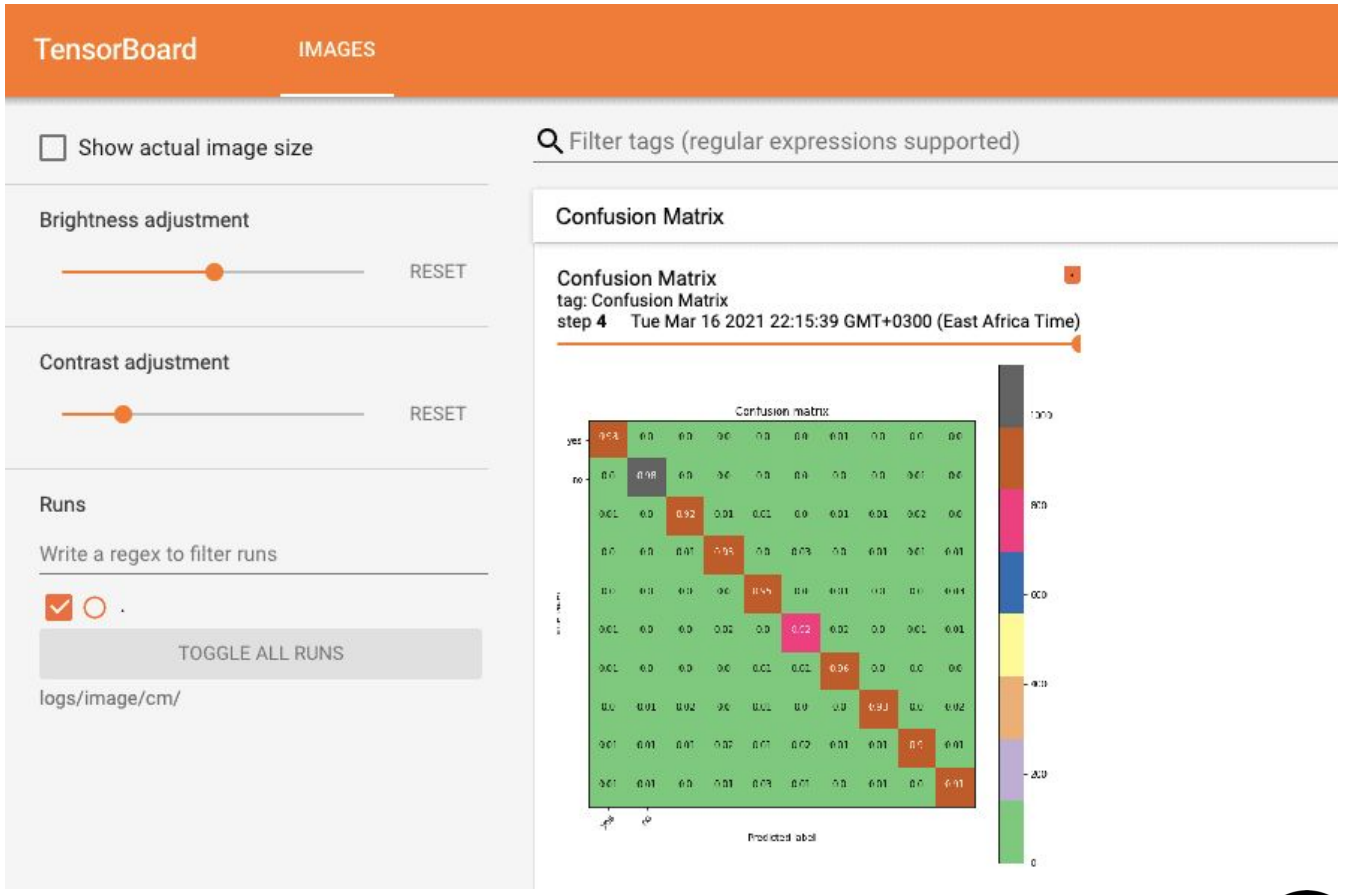
- TensorBoard es una herramienta que forma parte de TensorFlow y se utiliza para proporcionar las mediciones y visualizaciones necesarias durante el flujo de trabajo de aprendizaje automático.
  Es una biblioteca única, e incluso otras herramientas de ML* de la competencia, como PyTorch, usan TensorBoard.
  Algunas características incluyen:
  - Seguimiento y visualización de métricas como pérdida y precisión
    Visualización del gráfico del modelo (operaciones y capas)
    Visualización de histogramas de ponderaciones, sesgos u otros tensores a medida que cambian con el tiempo
    Proyección de incrustaciones en un espacio dimensional inferior
    Visualización de imágenes, texto y datos de audio
    Generación de perfiles de programas de TensorFlow

*ML- siglas de "machine learning", "aprendizaje automático" en inglés

TensorBoard: Visualizaciones del Rendimiento de los Modelos

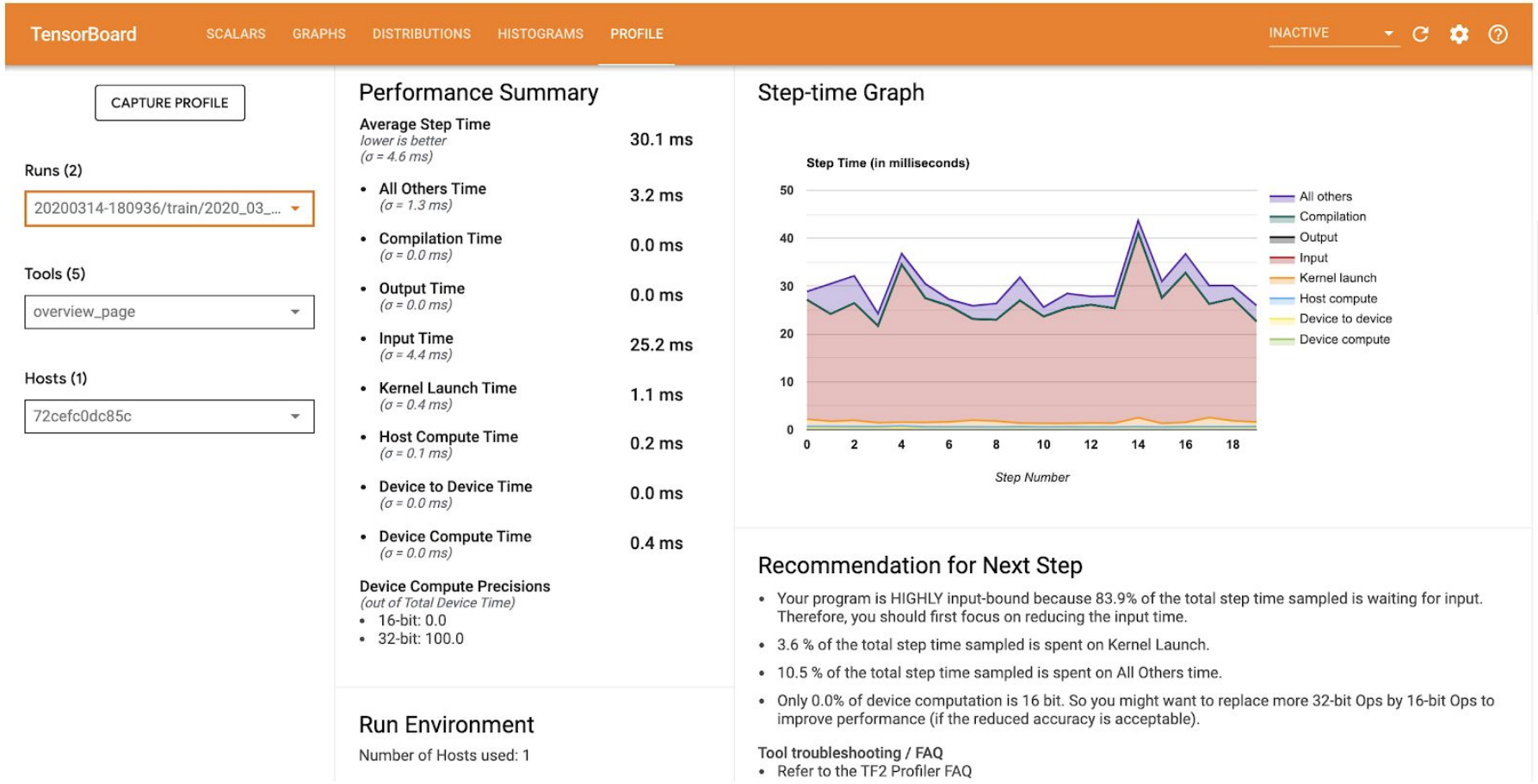
- Las series temporales y los gráficos escalares son las visualizaciones más típicas que se utilizan para supervisar el proceso de entrenamiento (las métricas de pérdida y precisión se muestran aquí). También se pueden ver otras visualizaciones personalizadas, por ejemplo, imágenes.



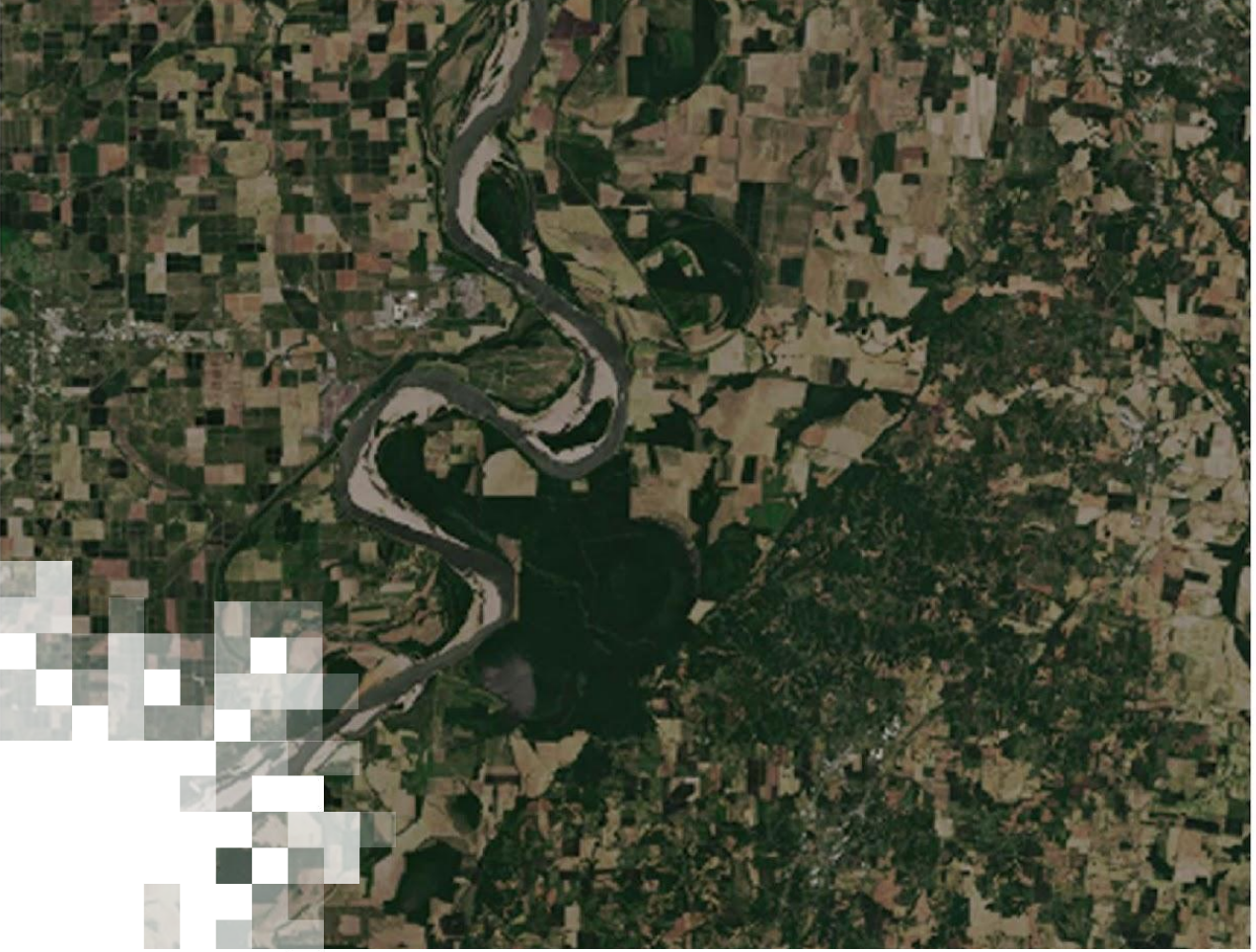


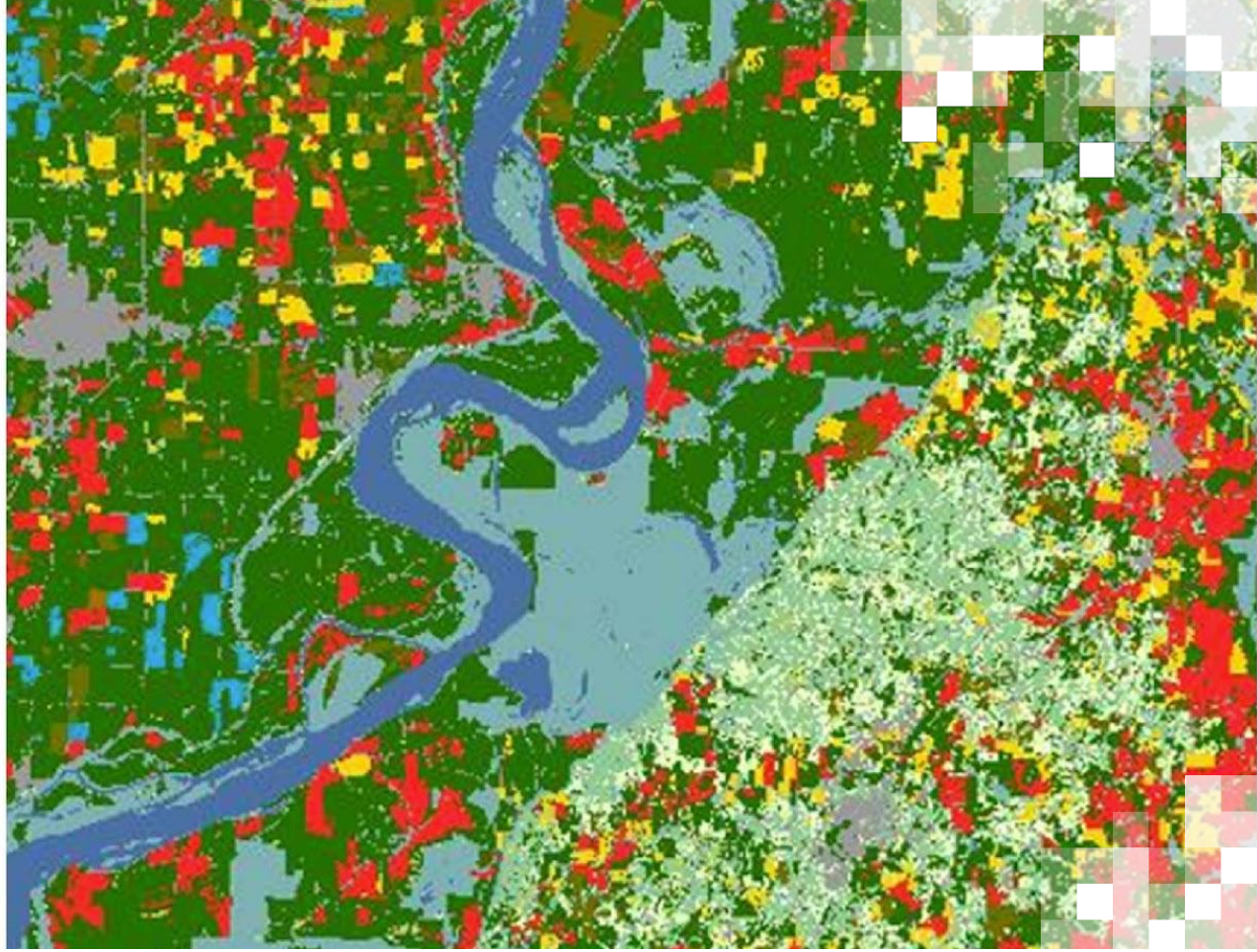
TensorBoard: El Perfilador TF Profiler

El perfilador TF Profiler le ayuda a comprender el consumo de recursos de hardware (tiempo y memoria) de las diversas operaciones de TensorFlow (operaciones) en su modelo y a resolver los embotellamientos de rendimiento para, en última instancia, hacer que el proceso se ejecute más rápidamente.



Nota- es posible que esto no funcione en Databricks debido a la naturaleza del entorno de ejecución de la base de datos.





3ra Parte, Sección 3:

Demostración del Procedimiento con Databricks (Ejecución del Código)

Resumen General de Databricks Community Edition

Enlace para inscribirse para Databricks Community Edition

- Codificación de estilo de Jupyter Notebook
  Databricks Community Edition permite hasta 10 GB de almacenamiento persistente en el "FileStore."
  - Puede almacenar archivos genéricos, tablas y código.
    Los cuadernos se almacenan en el área de "workspace".
- Puede poner en marcha instancias pequeñas con 2 CPU, 15 GB de RAM, 130 GB de almacenamiento local, Spark habilitado para usar desde el primer momento.
  Todo lo almacenado en el equipo local se pierde cuando se cierra la instancia.
  La ejecución del código del cuaderno durante más de ~60 minutos provocará que se apague el nodo. Sin embargo, mientras siga interactuando con el cuaderno (escribiendo y ejecutando código manualmente), normalmente permanecerá activo durante más tiempo.

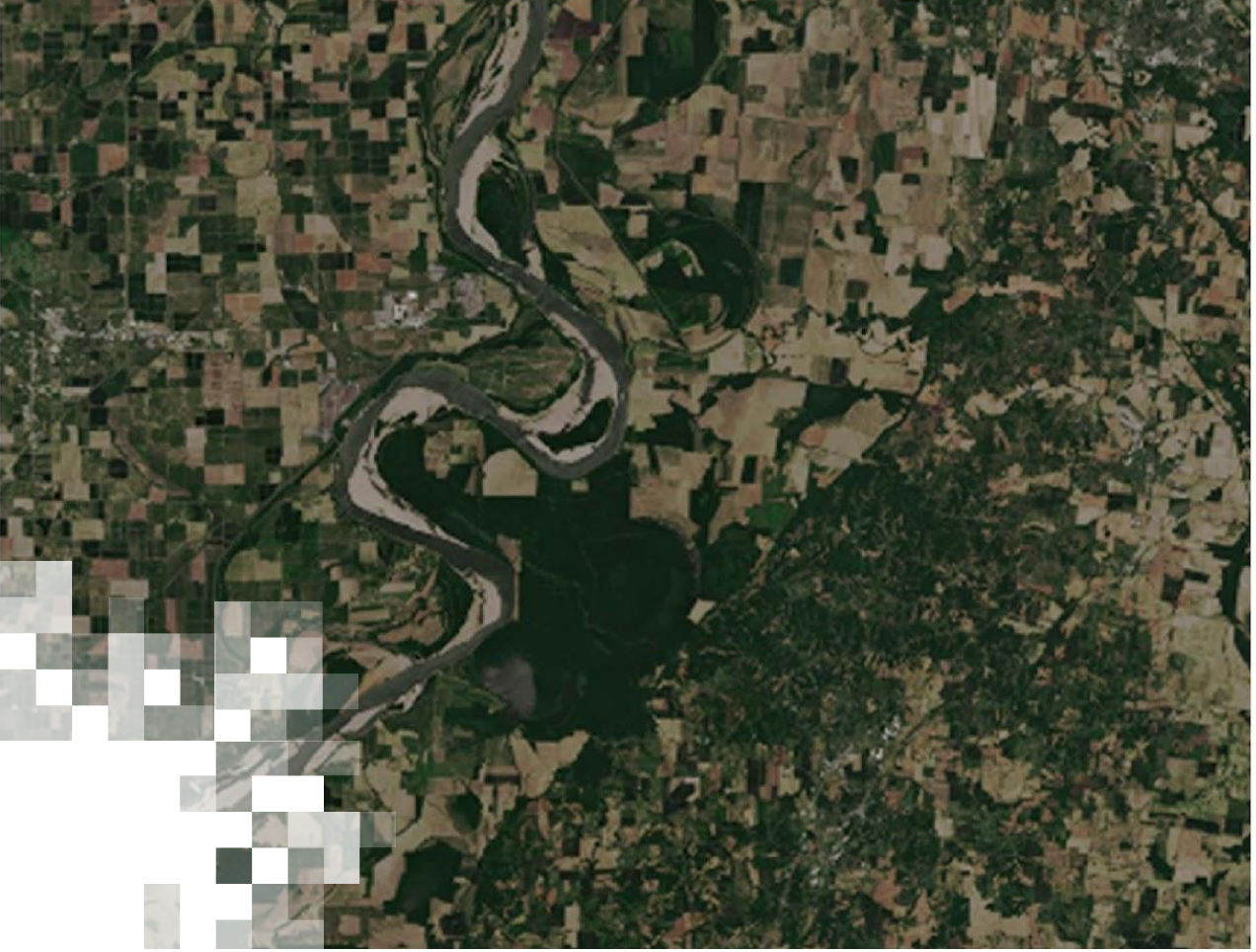
Demostración – Datos y Apuntes

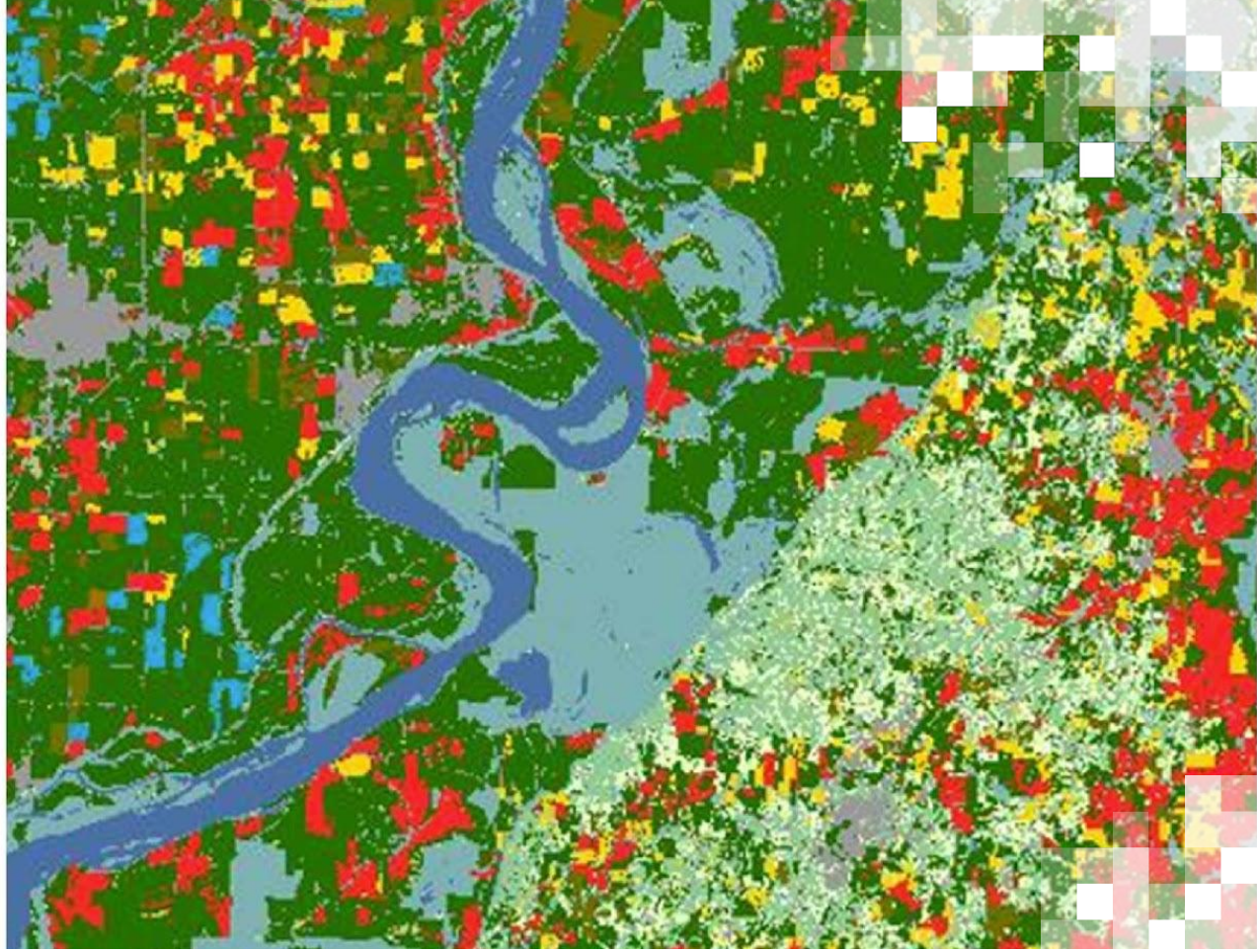
Materiales Disponibles para esta Demostración:

- Un script para el entrenamiento y la evaluación del modelo para la 3ra Parte. Los scripts de las partes anteriores también se pueden obtener en el sitio web (con otros materiales de la capacitación).
  Para obtener los datos para el entrenamiento y el modelo tal como existiría después de ejecutar los scripts en esta demostración de entrenamiento, descargue los archivos zip que se encuentran con los otros materiales de la capacitación.
  Nota: El modelo y el proceso de entrenamiento no se han ajustado exhaustivamente a través de búsquedas de hiperparámetros.

Cómo descargar los archivos resultantes del FileStore de su cuenta de Databricks:

- Esto no es intuitivo. Para descargarlo, debe navegar a la ruta de **SU** archivo utilizando el siguiente formato.
  - https://community.cloud.databricks.com/files/path/to/folder/filename.extension
    - 'path/to-folder' es la ruta de directorio en la que se encuentra el archivo en el FileStore.





Resumen de la Capacitación

Resumen de la 1ra y 2da Parte

- Las API nos permiten automatizar y escalar canalizaciones de procesamiento de datos muy grandes en preparación para el análisis y la creación de modelos.
  Una forma cómoda para modelar datos de imágenes de series temporales consiste en almacenarlos en formato de tabla Parquet.
  El almacenamiento de datos en formato Parquet y el uso de Spark/Databricks para consultar o manipular los datos permite una investigación y transformación rápidas.
  La creación de una cola personalizada y altamente eficiente para cargar información a través de conjuntos de datos de TensorFlow mediante archivos Parquet permite la extracción eficiente de subconjuntos de datos.
  Mediante el uso de map, shuffle, batch y prefetch, se puede optimizar el rendimiento de conjuntos de datos de TensorFlow con paralelización.

Resumen de la 3ra Parte

- **Keras** es un módulo dentro de TensorFlow que permite crear canalizaciones de entrenamiento rápidas de modelos de redes neuronales.
- TensorBoard Permite supervisar lo que hace la canalización del modelo durante el entrenamiento, incluso las **curvas de pérdida,** los **gráficos de métricas de rendimiento** y las **imágenes personalizadas**.
- Cuando se trabaja con modelos de datos satelitales, es importante visualizar los resultados tanto en el **espacio** como en el tiempo, ya que los resultados pueden cambiar drásticamente según la época del año en que se realicen las predicciones.
- La **aleatoriedad** es importante en el cargador de datos de **entrenamiento**, pero debe **eliminarse** del cargador de datos de **prueba** (para/al analizar los resultados).
- La definición de etiquetas tiene un gran impacto en los resultados del modelo. Por ejemplo, una clase escasamente representada en el conjunto de entrenamiento probablemente dará lugar a un rendimiento deficiente para esa clase en el momento de la prueba (p. ej., "Cultivated").
- ¡Podemos entrenar con éxito un modelo para predecir el tipo de cultivo en tiempo real durante la temporada utilizando este proceso!

Tarea y Certificados

- **Tarea:**
  - Habrá una tarea asignada
  - Abre el 19 de marzo de 2024
  - Acceso desde la página web de la capacitación
  - Debe enviar sus respuestas vía Formularios de Google
  - **Fecha límite: 1ro de abril de 2024**

- **Certificado de Finalización de Curso:**
  - Asista a las tres sesiones en vivo (la asistencia se registra automáticamente)
  - Complete la tarea dentro del plazo estipulado
  - Recibirá un certificado por correo electrónico aproximadamente dos meses después de la conclusión del curso.

Datos de Contacto

Formadores:

- John Just (John Deere)
  - JustJohnP@JohnDeere.com
- Erik Sorensen
  - SorensenErik@JohnDeere.com
- Sean McCartney
  - Sean.McCartney@nasa.gov

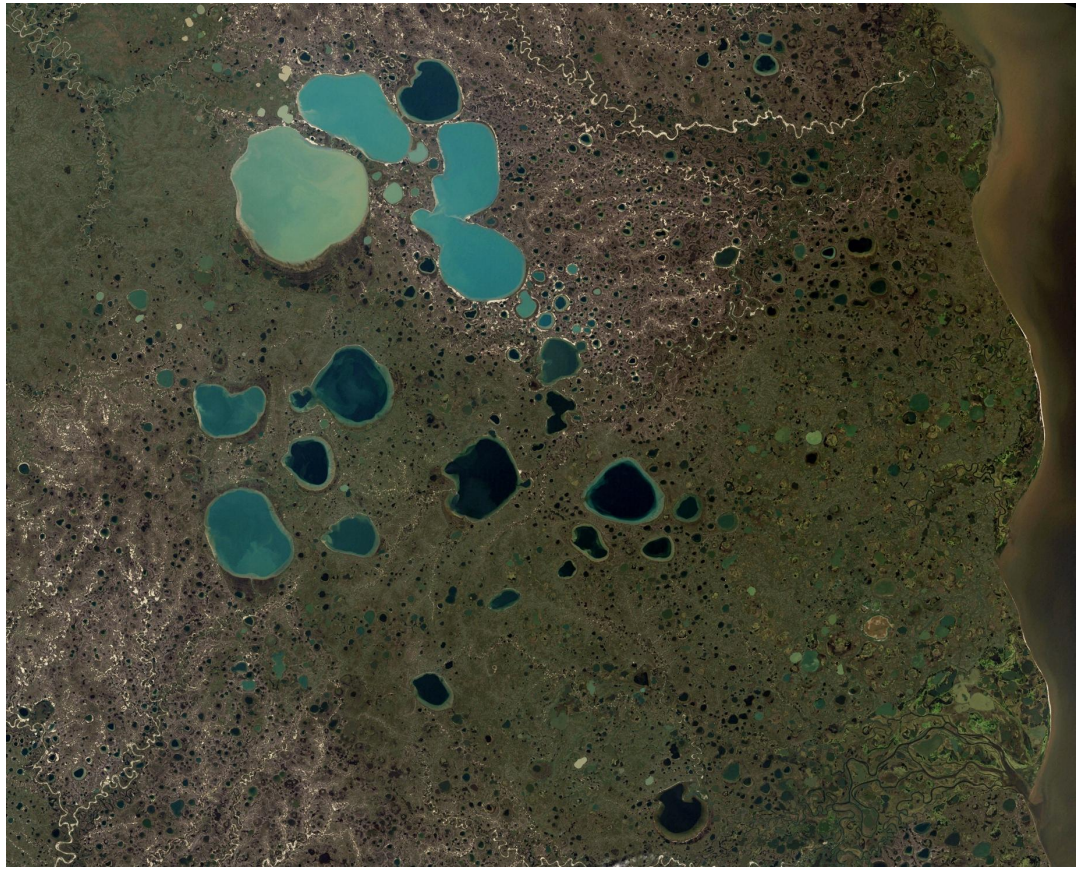
- Página web de ARSET
- ¡Síganos en X (antiguamente Twitter)!
  - @NASAARSET
- ARSET YouTube

Visite nuestros Programas Hermanos:

- DEVELOP
- SERVIR

¿Preguntas?

- Por favor escriba sus preguntas en la casilla denominada “Questions”. Las responderemos en el orden en que fueron recibidas.
- Publicaremos las preguntas y respuestas a la página web de la capacitación después de la conclusión del webinar.



https://earthobservatory.nasa.gov/images/6034/pothole-lakes-in-siberia


APPLIED REMOTE SENSING TRAINING PROGRAM
TRAIN - EMPOWER - ADVANCE

¡Gracias!