Clase 125 — TFRecord

Parte: 2 — Deep Learning · Fuente: Géron, cap. 13 § The TFRecord Format. Duración estimada: 60 min.

Clase 125 — TFRecord

> *Parte: 2 — Deep Learning · Fuente: Géron, cap. 13 § The TFRecord Format. Duración estimada: 60 min.*

Objetivo

Aprender el formato TFRecord — el formato binario nativo de TF, optimizado para datasets grandes (cientos de GB) que no caben en RAM. Saber escribirlo (tf.io.TFRecordWriter), parsearlo (tf.io.parse_single_example), y por qué es estándar en TPU/Vertex AI para training a escala.

Resultados de aprendizaje

Al finalizar, el estudiante podrá:

- Serializar un tf.train.Example con Features ↔ Feature (BytesList, Int64List, FloatList).
- Escribir TFRecord shards: with tf.io.TFRecordWriter('file.tfrecord') as w: w.write(serialized).
- Leer con tf.data.TFRecordDataset('file.tfrecord').map(parse_fn).
- Splitear datasets grandes en múltiples shards (*.tfrecord-00000-of-00010) para paralelizar reads.
- Reconocer cuándo TFRecord vale la pena vs alternativas modernas (Parquet, WebDataset).

Temas

- tf.train.Example: estructura protobuf con Features (dict de strings a Feature).
- Feature types: BytesList, Int64List, FloatList.
- Serialize → escribir → leer → parse.
- Sharding: data.tfrecord-NNNNN-of-MMMMM.
- tf.data.experimental.bucket_by_sequence_length para batches eficientes por longitud (NLP).

Definiciones y características

- TFRecord: archivo binario de records seriales protobuf. Eficiente, splitteable, compresible.
- tf.train.Example: proto que envuelve un dict de features.
- SerializeToString(): convierte el Example en bytes para escribir.
- parse_single_example(serialized, feature_spec): parsea con un schema declarado.
- Sharding: dividir un dataset en N archivos → reads paralelos vía interleave.

Dataset / recursos

- Fashion-MNIST exportado a TFRecord.
- Librerías: tensorflow.

Ejercicios

1. Escribir: convertir Fashion-MNIST (60 000 imágenes) a 10 shards TFRecord. Cada Example tiene image (bytes) y label (int).

2. Leer y parsear: ds = tf.data.TFRecordDataset(glob.glob('shards/*.tfrecord')).map(parse_fn). Iterar y verificar shapes.
3. Compresión: escribir con options=tf.io.TFRecordOptions(compression_type='GZIP'). Comparar tamaño.
4. Reads paralelos: ds = ds.interleave(lambda f: tf.data.TFRecordDataset(f), num_parallel_calls=AUTOTUNE). Medir speedup vs lectura serial.
5. Schema: usar tf.io.FixedLenFeature (tamaño fijo) vs VarLenFeature (variable, returns SparseTensor).

Homework verificable

Pipeline completo de Fashion-MNIST con TFRecord:

1. Escribir 10 shards.
2. Leer con interleave + map + batch + prefetch.
3. Entrenar un modelo y comparar tiempo vs cargar desde NumPy.

Criterio de aceptación: el pipeline TFRecord debe escalar mejor cuando los datos no caben en RAM (simular limitando RAM si es necesario); para Fashion-MNIST en RAM, los tiempos deben ser similares.

Errores comunes

Síntoma / mensaje	Causa y cómo arreglar
parse_single_example falla con "Key not fo	Schema no coincide con lo escrito. Fix: im
Tamaño TFRecord enorme	No comprimiste. Fix: compression_type='GZI
tf.io.parse_tensor vs parse_single_example	Para tensores serializados con tf.io.seria
Reads serializados secuenciales en un únic	El throughput está limitado por 1 disco. F
Sin drop_remainder los últimos batches tie	Algunos modelos rompen. Fix: batch(N, drop

Preguntas frecuentes

¿TFRecord o Parquet?

TFRecord para TF/JAX. Parquet para data science general (Polars, Spark, DuckDB). PyTorch tiene WebDataset que es similar a TFRecord pero más portable (tarballs).

¿Cuándo TFRecord es necesario?

Cuando los datos no caben en RAM (>10 GB) y querés training rápido. Para datasets chicos (< 1 GB), arrays NumPy alcanzan.

¿Cuántos shards?

Regla práctica: 100-1000 archivos, cada uno 100 MB - 1 GB. Para TPU training, ≥ 8 shards por nodo.

¿Comprimir o no?

Comprimir si I/O es el bottleneck. Si CPU es el cuello (decode lento), no comprimir. Probá ambos.

¿Hugging Face datasets?

Sí, formato moderno (basado en Arrow/Parquet) con caching automático y API más amigable. Para LLMs, default industrial. Saber TFRecord es útil legacy pero datasets lo está reemplazando.

Referencias

- Géron, cap. 13 — The TFRecord Format.
- TF — TFRecord and tf.train.Example.
- WebDataset — alternativa PyTorch.
- Hugging Face datasets — alternativa moderna multi-framework.

Siguiente clase

Clase 126 — Keras preprocessing layers

Apéndice: notebook (primer bloque)

Primera celda ejecutable del notebook de la clase.

```
# Imports y configuración inicial
```

Archivos complementarios

- notebook.ipynb