Bab 19: Training and Deploying TensorFlow Models at Scale

Tujuan Bab

Menyediakan panduan tentang bagaimana melatih model TensorFlow secara efisien pada skala besar (dengan distribusi atau akselerasi hardware) serta bagaimana menyiapkan model untuk deployment ke produksi.

Konsep Utama

1. Training pada Skala Besar

Akselerasi Perangkat Keras

- GPU: Cocok untuk komputasi paralel seperti pelatihan deep learning
- TPU (Tensor Processing Unit): Didesain khusus oleh Google untuk TensorFlow

Multi-GPU dan Multi-Device Training

Menggunakan tf.distribute.Strategy:

```
strategy = tf.distribute.MirroredStrategy()

with strategy.scope():
   model = build_model()
```

Cloud Training

- Gunakan layanan seperti Google Cloud AI Platform atau Vertex AI
- Menyimpan data dan model di cloud storage
- Gunakan Docker atau notebook untuk eksperimen

2. SavedModel Format

Standar format penyimpanan TensorFlow model:

```
model.save("path/to/saved_model")
```

Termasuk:

- Arsitektur
- Bobot
- Optimizer
- Konfigurasi pelatihan

3. Serving dan Deployment

TensorFlow Serving

Digunakan untuk menyajikan model via REST API atau gRPC:

- Bisa dijalankan dalam Docker container
- Mendukung versi model secara dinamis

```
docker run -p 8501:8501 \
  --mount type=bind,source=/models/model_dir,target=/models/my_model \
  -e MODEL_NAME=my_model -t tensorflow/serving
```

-TensorFlow Lite

Digunakan untuk deployment ke perangkat mobile dan embedded systems.

-TensorFlow.js

Untuk deployment model ke browser atau Node.js.

4. TFRecord dan Data Pipelines

Format TFRecord

Efisien untuk menyimpan dataset besar dalam format binary:

```
feature = {
```

```
  'image': tf.train.Feature(bytes_list=tf.train.BytesList(value=[image_bytes])),
  ...
}
```

tf.data API

Untuk membangun pipeline input data:

```
dataset = tf.data.TFRecordDataset(filenames)
dataset = dataset.map(parse_example).batch(32).prefetch(1)
```

5. Monitoring dengan TensorBoard

TensorBoard dapat digunakan untuk:

- Visualisasi loss, accuracy
- Lacak distribusi bobot
- Lihat model graph
- Debugging bottleneck I/O

```
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir="./logs")
```

Proyek Colab

Isi Notebook:

1. Latihan dengan strategi distribusi

- Menggunakan tf.distribute.MirroredStrategy untuk menjalankan pelatihan model di multi-GPU
- Uji kompatibilitas model custom dengan strategi distribusi

2. Simpan dan muat model SavedModel

- Demonstrasi model.save() dan keras.models.load_model()

3. TFRecord & tf.data

- Buat TFRecord dari data Fashion MNIST
- Gunakan tf.data.TFRecordDataset untuk membacanya kembali

4. TensorFlow Serving

- Simulasi export model untuk disajikan lewat TensorFlow Serving (dijelaskan secara konseptual)

Inti Pelajaran

Konsep	Penjelasan
SavedModel	Format standar menyimpan model lengkap
MirroredStrategy	Distribusi pelatihan ke multi-GPU
TFRecord	Format data efisien untuk penyimpanan besar
tf.data	API untuk preprocessing dan pipeline data
TensorFlow Serving	Menyajikan model via API untuk produksi
TensorFlow Lite / JS	Deployment ke mobile & web
TensorBoard	Monitoring dan debugging pelatihan

Kesimpulan

Bab ini menutup buku dengan fokus praktis dan industri: bagaimana model machine learning disiapkan untuk pelatihan berskala besar, disimpan secara efisien, dan di-deploy ke sistem produksi. Aspek seperti efisiensi data input, format penyimpanan, dan serving menjadi kunci dalam membuat model siap pakai dalam aplikasi nyata.