Chapter 13: Summary

Diedrick Darrell Darmadi - 1103223031

Bab 13 membahas bagaimana menangani data menggunakan TensorFlow, khususnya dengan API **tf.data**, yang dirancang untuk memproses data dalam skala besar secara efisien. Bab ini menekankan bahwa dalam proyek machine learning nyata, kualitas dan alur data sering lebih menentukan kesuksesan model dibanding kompleksitas arsitekturnya. Oleh karena itu, memahami cara memuat, membersihkan, mentransformasi, dan mengalirkan data secara efisien merupakan keterampilan penting sebelum melakukan pelatihan model.

Pembahasan dimulai dengan pengenalan objek **tf.data.Dataset**, yaitu struktur yang memungkinkan kita membuat pipeline data yang bersifat *lazy*, efisien, dan dapat berjalan paralel di CPU maupun GPU. Dataset dapat dibuat dari beragam sumber, seperti list Python, file CSV, folder gambar, atau data biner seperti TFRecord. Dataset ini kemudian dapat dimodifikasi melalui operasi seperti map(), batch(), shuffle(), dan prefetch(). Operasi-operasi ini sangat penting untuk meningkatkan performa pipeline—misalnya, shuffle() membantu pelatihan menjadi lebih stabil, sedangkan prefetch() memungkinkan CPU dan GPU bekerja secara paralel sehingga waktu idle dapat diminimalkan.

Bab ini kemudian menjelaskan bagaimana membangun **pipeline preprocessing** yang lengkap. Dengan API map(), kita dapat menambahkan transformasi seperti normalisasi data, konversi tipe, augmentasi gambar, parsing data mentah, atau operasi pembersihan lainnya. TensorFlow menjalankan operasi ini secara *graph-based*, sehingga mampu mengoptimalkan pipeline dan menjalankannya dengan performa tinggi. Selain itu, pembahasan juga mencakup penggunaan cache() untuk menyimpan hasil preprocessing sehingga tidak perlu diproses ulang setiap epoch, serta interleave() untuk membaca banyak file secara paralel, meningkatkan throughput data secara signifikan.

Bagian berikutnya fokus pada **TFRecord**, format data biner yang direkomendasikan TensorFlow untuk dataset berskala besar. Format ini memungkinkan penyimpanan dan pembacaan data yang jauh lebih cepat dibanding format teks seperti CSV. Bab ini mengajarkan cara membuat file TFRecord, mengisi data dalam bentuk *protocol buffer*, serta mem-parsing kembali data tersebut menggunakan tf.io.parse_single_example(). Teknik ini sangat penting saat menangani dataset besar seperti ImageNet, atau ketika bekerja di cloud environment seperti Google Cloud Storage.

Selain itu, bab ini juga membahas tentang **serialization dan tf.Example**, yaitu cara terstandardisasi TensorFlow untuk menyimpan data dengan tipe variabel seperti string, float, dan integer. Setelah memahami serialization, pembaca dapat membuat pipeline data yang sepenuhnya otomatis, mulai dari membaca dataset mentah hingga memberikan batch yang siap dilatih ke model deep learning.

Pada bagian akhir, bab ini membahas teknik **performance optimization** dalam pipeline data. Teknik seperti menambahkan parameter num_parallel_calls=tf.data.AUTOTUNE, menerapkan prefetch(), menggunakan cache(), dan membagi dataset menjadi shard untuk dibaca paralel sangat penting dalam mempercepat proses pelatihan, terutama pada GPU dan TPU. Bab ini menekankan bahwa pipeline data yang buruk akan membuat GPU menganggur, menyebabkan pelatihan menjadi lambat meskipun model sangat efisien.

Secara keseluruhan, Bab 13 memberikan wawasan mendalam tentang bagaimana memuat, membersihkan, mengonversi, dan menyiapkan data dalam skala besar menggunakan TensorFlow. Pembaca diajarkan membuat pipeline yang tidak hanya benar secara fungsional tetapi juga optimal dari sisi performa. Bab ini menjadi landasan penting sebelum mempelajari model-model deep learning

yang lebih kompleks, karena tanpa pipeline data yang efisien, performa pelatihan akan terhambat dan hasil model tidak akan maksimal.