Bab 11: Training Deep Neural Networks

Tujuan Bab

Memahami tantangan dalam melatih jaringan neural yang dalam (*deep neural networks*), serta teknik dan praktik terbaik untuk mengatasinya agar pelatihan stabil, efisien, dan memberikan hasil yang baik.

Konsep Utama

1. Tantangan Melatih Deep Neural Networks

- Vanishing gradients: Gradien sangat kecil di lapisan awal, membuat pembelajaran lambat.
- Exploding gradients: Gradien membesar secara eksponensial, menyebabkan ketidakstabilan.
- Overfitting: Model kompleks cenderung menghafal data pelatihan.
- Training lambat: Jaringan dalam memerlukan banyak iterasi dan data besar.

2. Feature Scaling

Sangat penting untuk menormalkan input (misal, menggunakan *StandardScaler* dari Scikit-Learn) agar pelatihan lebih cepat dan stabil.

3. Berat Awal (Weight Initialization)

Pemilihan inisialisasi bobot berpengaruh besar:

- He initialization (untuk ReLU dan turunannya)
- Glorot/Xavier initialization (untuk sigmoid/tanh)

Contoh (Keras):

```
keras.layers.Dense(10, activation="relu", kernel_initializer="he_normal")
```

4. Aktivasi

- ReLU (Rectified Linear Unit) → default untuk hidden layer
- Leaky ReLU → versi modifikasi ReLU agar tetap memiliki gradien saat negatif
- ELU → perbaikan ReLU, mendukung nilai negatif yang lebih halus
- selu (Scaled ELU) → digunakan dengan self-normalizing networks

5. Batch Normalization

Menstabilkan dan mempercepat pelatihan dengan menormalkan output layer per mini-batch.

Contoh (Keras):

```
keras.layers.BatchNormalization()
```

6. Gradient Clipping

Membatasi nilai maksimum gradien untuk menghindari *exploding gradients*.

Contoh:

```
optimizer = keras.optimizers.SGD(clipvalue=1.0)
```

7. Early Stopping

Menghentikan pelatihan jika model mulai overfitting pada data validasi.

8. Optimizer yang Lebih Baik

- Adam: Umumnya terbaik untuk jaringan dalam
- Nadam: Gabungan Adam dan Nesterov momentum
- RMSProp: Alternatif populer untuk pelatihan stabil

9. Regularisasi

- Dropout: Menonaktifkan neuron secara acak selama pelatihan
- keras.layers.Dropout(rate=0.5)
- L1/L2 regularization: Menambahkan penalti terhadap bobot besar

10. Callback Keras

Gunakan callback untuk:

- Menyimpan model terbaik (ModelCheckpoint)
- Menghentikan pelatihan dini (EarlyStopping)
- Menyimpan log (TensorBoard)

Proyek / Notebook Praktik

Isi:

- Eksperimen dengan berbagai teknik seperti weight initialization, batch normalization, dan dropout
- Visualisasi learning curve
- Perbandingan performa optimizer: SGD vs Adam
- Implementasi EarlyStopping dan ModelCheckpoint
- Evaluasi dropout untuk mencegah overfitting

Inti Pelajaran

Teknik	Tujuan
Feature Scaling	Membuat pelatihan lebih stabil
He/Glorot Initialization	Menstabilkan propagasi awal sinyal
ReLU, Leaky ReLU, ELU	Mempercepat konvergensi, hindari saturasi
Batch Normalization	Menstabilkan aktivasi antar layer

Dropout	Mencegah overfitting
Adam, Nadam	Optimizer modern yang efisien
Gradient Clipping	Menghindari exploding gradients
Early Stopping	Mencegah overfitting, hemat waktu

Kesimpulan

Bab ini mempersiapkan landasan untuk membangun dan melatih deep learning model yang stabil dan efisien. Fokus utama adalah pada penerapan teknik yang membantu pelatihan model besar yang sebelumnya sulit dikonvergenkan. Teknik ini akan sangat berguna saat membangun arsitektur lanjutan seperti CNN dan RNN pada bab berikutnya.

Jika diperlukan, pembahasan dapat dilanjutkan ke Bab 12: Custom Models and Training with TensorFlow.