Clase 110 — Batch Normalization, Layer Normalization

Parte: 2 — Deep Learning · Fuente: Géron, cap. 11 § Batch Normalization. Duración estimada: 75 min.

Clase 110 — Batch Normalization, Layer Normalization

Parte: 2 — Deep Learning · Fuente: Géron, cap. 11 § Batch Normalization. Duración estimada: 75 min.

Objetivo

Entender BatchNorm (Ioffe & Szegedy 2015) — la técnica que destrabó el entrenamiento de redes muy profundas estandarizando las activaciones en cada capa — y su variante LayerNorm (Ba, Kiros & Hinton 2016) — usada en Transformers y RNN porque no depende del batch. Saber dónde poner BN en la arquitectura, qué problemas tiene (batch chico, distribución entre train/inference) y cuándo preferir LN.

Resultados de aprendizaje

Al finalizar, el estudiante podrá:

- Aplicar BatchNormalization() antes o después de la activación (debate clásico — moderno: antes suele ser mejor para ReLU, después para GELU).
- Explicar qué hace BN en train (normaliza con stats del batch) vs inference (usa moving averages acumulados).
- Aplicar LayerNormalization() en RNN y Transformers; saber por qué allí BN falla.
- Reconocer las 3 variantes: BN, LN, GroupNorm (Wu & He 2018, para batch chico en visión).
- Diagnosticar el problema de "train-test mismatch" cuando el batch en inference es muy distinto al de train.

Temas

- BN: $y = \gamma \cdot (x - \mu_{batch})/\sigma_{batch} + \beta$. γ, β trainables.
- Beneficios: convergencia más rápida, regularización mild, permite LR más altos.
- BN en train vs inference: moving avg de μ, σ acumulados.
- LN: normaliza sobre los features de una sola muestra (no sobre el batch).
- GroupNorm: agrupa canales, normaliza dentro de cada grupo. Para batch chico (segmentación, detección).
- ¿Antes o después de la activación? Géron y la práctica moderna: antes funciona mejor con ReLU, después con GELU/Swish.

Definiciones y características

- BatchNorm: normaliza cada feature usando la media y varianza del batch actual durante training.
- γ (scale), β (shift): parámetros learnables que permiten al modelo deshacer la normalización si necesita.
- Moving averages: durante training, mantiene EMAs de μ, σ. En inference usa esos valores fijos.
- LayerNorm: normaliza sobre features de una sola muestra, independiente del batch. Default en NLP/Transformers.
- GroupNorm: agrupa N canales y normaliza dentro de cada grupo. Funciona con batch_size=1.
- InstanceNorm: como BN pero por sample individual (estilo style transfer).

Dataset / recursos

- Fashion-MNIST + un modelo profundo ([512, 256, 128, 64, 10]).
- Librerías: tensorflow, keras.

Ejercicios

1. BN vs sin BN: entrenar el mismo MLP con y sin BN. Comparar curvas de val_loss y tiempo hasta llegar a accuracy 0.85.
2. ¿Antes o después de la activación?: probar las dos variantes (Dense → BN → ReLU vs Dense → ReLU → BN). Comparar.
3. Inference mode: entrenar con BN, cambiar a training=False, predecir un batch y comparar con training=True. Las predicciones cambian (correcto).
4. Batch chico: forzar batch_size=4 y entrenar con BN. Observar inestabilidad. Cambiar a LayerNormalization y verificar que se estabiliza.
5. LayerNorm en RNN: aplicar keras.layers.LSTM con recurrent_activation y un LayerNormalization previo. Útil como anticipo de Transformers.

Homework verificable

Sobre MLP [512, 256, 128, 64, 10] en Fashion-MNIST:

1. Entrenar 3 versiones: sin norm, con BN, con LN.
2. Reportar val_accuracy + épocas hasta val_loss < 0.4.
3. Entrenar BN con batch_size=128 y luego inferir batch por batch de tamaño 1. Verificar que el accuracy no se rompe (gracias a moving averages).

Criterio de aceptación: BN debe acelerar la convergencia ≥ 30 % (menos épocas para mismo loss); LN debe ser estable también pero típicamente algo peor en MLP feedforward.

Errores comunes

Síntoma / mensaje	Causa y cómo arreglar
BN en batch_size=1 → entrenamiento inestab	μ del batch es la muestra misma → σ=0. Fix
Métricas en val muy distintas de train par	Las stats de BN en inference son distintas
Olvidar pasar training=True/False en custo	Por default es False → BN usa moving avg i
BN en una RNN	Las estadísticas cambian con la longitud d
Aplicar BN a la última capa antes de softm	Generalmente innecesario y a veces dañino

Preguntas frecuentes

¿BN reemplaza dropout?

Parcialmente. BN tiene efecto regularizador mild (el ruido del batch). En CNNs/MLPs profundos suele alcanzar; en Transformers se usa LN + dropout juntos.

¿LN o BN en Transformers?

LN siempre. BN falla porque (a) batch chico es común, (b) longitudes variables rompen las estadísticas.

¿Por qué LN no es default también en CNN?

Empíricamente BN gana en visión (los canales se benefician de stats compartidos). LN gana en NLP donde el orden temporal importa más.

¿fused=True qué hace?

BatchNormalization(fused=True) usa una implementación CUDA optimizada que fusiona ops. Default auto lo elige cuando puede. Velocidad ~ 20 % más rápido.

¿Cuánto cuesta BN en cómputo?

Casi gratis en GPU (ops elementwise + pocas reducciones). En CPU es notable pero negligible en práctica.

Referencias

- Géron, cap. 11 — Batch Normalization.
- Ioffe & Szegedy (2015), Batch Normalization, ICML.
- Ba, Kiros & Hinton (2016), Layer Normalization.
- Wu & He (2018), Group Normalization, ECCV.
- Santurkar et al. (2018), How Does Batch Normalization Help Optimization?, NeurIPS — explicación moderna.

Siguiente clase

Clase 111 — Gradient clipping

Apéndice: notebook (primer bloque)

Primera celda ejecutable del notebook de la clase.

```
# Imports y configuración inicial
```

Archivos complementarios

- notebook.ipynb