Bài Giảng: Các Kỹ Thuật Regularization trong Deep Learning

Biên soạn: Tran Que

Ngày 19 tháng 10 năm 2025

Mục lục

1 Tổng quan

Trong học sâu (*Deep Learning*), **regularization** là tập hợp các kỹ thuật giúp mô hình *giảm overfitting* và *tăng khả năng tổng quát hoá* (generalization). Overfitting xảy ra khi mô hình học quá kỹ dữ liệu huấn luyện, làm giảm hiệu suất trên dữ liệu kiểm tra.

Regularization hoạt động bằng cách:

- Giới hạn độ phức tạp của mô hình.
- Thêm nhiễu có kiểm soát hoặc ràng buộc lên trọng số.
- Tăng đa dạng dữ liệu hoặc tính bất biến.

—

2 Regularization dựa trên hàm mất mát (Loss-based)

2.1 L1 Regularization (Lasso)

Thêm ràng buộc chuẩn L1 vào hàm mất mát:

$$L' = L + \lambda \sum_i |w_i|$$

Tác dụng: ép nhiều trọng số về 0, giúp chọn lọc đặc trưng.

Ví dụ TensorFlow:

```
from tensorflow.keras import layers, models, regularizers

model = models.Sequential([
    layers.Dense(64, activation='relu',
                 kernel_regularizer=regularizers.l1(0.001)),
    layers.Dense(1, activation='sigmoid')
])
model.compile(optimizer='adam', loss='binary_crossentropy')
model.summary()
```

—

2.2 L2 Regularization (Ridge)

Thêm ràng buộc chuẩn L2:

$$L' = L + \lambda \sum_i w_i^2$$

Tác dụng: giảm độ lớn trọng số, ổn định mô hình.

Ví dụ TensorFlow:

```
model = models.Sequential([
    layers.Dense(64, activation='relu',
                 kernel_regularizer=regularizers.l2(0.001)),
    layers.Dense(1, activation='sigmoid')
])
```

—

2.3 Elastic Net

Kết hợp L1 và L2:

$$L' = L + \lambda_1 \|w\|_1 + \lambda_2 \|w\|_2^2$$

Ví dụ TensorFlow:

```
model = models.Sequential([
    layers.Dense(64, activation='relu',
                 kernel_regularizer=regularizers.l1_l2(l1=1e-4, l2=1e-3)),
    layers.Dense(1, activation='sigmoid')
])
```

—

3 Regularization trên mạng Neural

3.1 Dropout

Ngẫu nhiên tắt một số neuron trong quá trình huấn luyện:

$$h_i' = \begin{cases} 0, & \text{với xác suất } p, \\ \dfrac{h_i}{1-p}, & \text{ngược lại.} \end{cases}$$

Ví dụ TensorFlow:

```
model = models.Sequential([
    layers.Dense(128, activation='relu'),
    layers.Dropout(0.5),
    layers.Dense(64, activation='relu'),
    layers.Dropout(0.3),
    layers.Dense(10, activation='softmax')
])
```

—

3.2 Batch Normalization

Chuẩn hoá đầu ra của từng lớp giúp ổn định gradient:

$$\hat{x} = \frac{x - \mu}{\sqrt{\sigma^2 + \epsilon}}$$

Ví dụ TensorFlow:

```
model = models.Sequential([
    layers.Dense(128, activation='relu'),
    layers.BatchNormalization(),
    layers.Dense(64, activation='relu'),
    layers.BatchNormalization(),
    layers.Dense(10, activation='softmax')
])
```

—

4 Regularization ở mức dữ liệu (Data-level)

4.1 Data Augmentation

Mở rộng dữ liệu huấn luyện bằng các phép biến đổi ngẫu nhiên. Ví dụ dưới đây minh họa cách áp dụng tăng cường dữ liệu ảnh trong TensorFlow:

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import tensorflow as tf

datagen = ImageDataGenerator(
    rotation_range=20,
    horizontal_flip=True,
    zoom_range=0.2
)

# Ap dung cho anh don le (vi du tao du lieu ao)
img = tf.random.uniform((1, 128, 128, 3))
aug_iter = datagen.flow(img, batch_size=1)
aug_img = next(aug_iter)[0]
```

—

4.2 Noise Injection

Thêm nhiễu vào dữ liệu đầu vào hoặc trọng số để tăng khả năng khái quát.

Ví dụ TensorFlow:

```
from tensorflow.keras import layers, models

model = models.Sequential([
    layers.Dense(64, activation='relu'),
    layers.GaussianNoise(0.1),
    layers.Dense(32, activation='relu'),
    layers.Dense(1)
])
```

—

5 Regularization cấu trúc (Structural)

5.1 Early Stopping

Dừng huấn luyện khi lỗi trên tập kiểm tra bắt đầu tăng.

Ví dụ TensorFlow:

```
from tensorflow.keras.callbacks import EarlyStopping

early_stop = EarlyStopping(
    monitor='val_loss', patience=5, restore_best_weights=True
)

model.fit(x_train, y_train, validation_data=(x_val, y_val),
          epochs=100, callbacks=[early_stop])
```

—

6 Regularization xác suất (Bayesian / Dropout)

6.1 Bayesian Dropout (Monte Carlo Dropout)

Duy trì Dropout trong giai đoạn dự đoán để ước lượng bất định:

```
import numpy as np

# Du doan nhieu lan voi dropout bat
preds = [model(x_test, training=True) for _ in range(20)]
mean_pred = np.mean(preds, axis=0)
uncertainty = np.var(preds, axis=0)
```

—

7 Regularization hình học (Manifold)

Ràng buộc các biểu diễn gần nhau trong không gian ẩn:

$$L' = L + \lambda \|f(x_i) - f(x_j)\|^2$$

Ý tưởng TensorFlow:

```
import tensorflow as tf

def manifold_loss(z1, z2):
    # z1, z2 la hai vector bieu dien gan nhau
    return tf.reduce_mean(tf.square(z1 - z2))
```

—

8 Bảng tổng hợp kỹ thuật

Bảng 1: Tóm tắt các kỹ thuật Regularization trong Deep Learning

Nhóm kỹ thuật	**Ví dụ**	**Tác dụng chính**
Loss-based	L1, L2, Elastic Net	Giảm trọng số, chọn lọc đặc trưng
Network-level	Dropout, BatchNorm	Giảm phụ thuộc neuron, ổn định huấn luyện
Data-level	Data Augmentation, Noise	Tăng đa dạng dữ liệu, khái quát hóa
Structural	Early Stopping	Ngăn học quá mức
Bayesian	Monte Carlo Dropout	Định lượng bất định
Geometry-based	Manifold loss	Giữ cấu trúc dữ liệu ẩn

—

9 Kết luận

Regularization đóng vai trò cốt lõi trong học sâu. Việc lựa chọn và kết hợp hợp lý các kỹ thuật giúp mô hình học hiệu quả, tránh overfitting và phản ánh đúng bản chất dữ liệu. Đây là nền tảng cho các hướng nghiên cứu hiện đại như *trust modeling*, *graph learning* và *Bayesian deep learning*.

—

HƯỚNG DẪN KIỂM TRA 1

(Kiểm tra 2 tiết – Thứ Ba cho lớp CQ và Thứ Năm cho lớp CLC)

- Sinh viên tạo thư mục để lưu data và code: `C:\DATA\`
- Copy sẵn 5 ảnh của 5 loại hoa vào thư mục `DATA`.
- Nghiên cứu lại các Bài tập **4, 5, 6**.
- Ôn lại các kỹ thuật **Data Augmentation** cho ảnh và văn bản (text).
- Nghiên cứu các kỹ thuật **Regularization** (tham khảo phần bài giảng này và tài liệu từ ChatGPT).

Gợi ý: Sinh viên nên thực hành trước các ví dụ TensorFlow trong bài để hiểu rõ cách kết hợp Dropout, L2 Regularization, Data Augmentation và EarlyStopping trong một pipeline huấn luyện.