CSE: **Faculty of Computer Science and Engineering**
Thuyloi University

HUẤN LUYỆN TỐI ƯU

Overfitting

TS. Nguyễn Thị Kim Ngân

Nội dung

1. Phương pháp hạn chế suy giảm đạo hàm
2. Phương pháp tinh chỉnh quá trình huấn luyện
3. Phương pháp huấn luyện chuyển tiếp

Nội dung

1. **Phương pháp hạn chế suy giảm đạo hàm**
2. Phương pháp tinh chỉnh quá trình huấn luyện
3. Phương pháp huấn luyện chuyển tiếp

Tiêu biến gradient

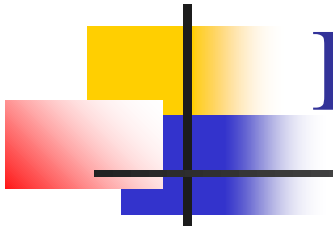
- Cho mạng nơ-ron sâu với L tầng, đầu vào x và đầu ra o. Mỗi tầng *l* được định nghĩa bởi một phép biến đổi f_l với tham số là trọng số W_l. Mạng nơ-ron này có thể được biểu diễn như sau:

$$\mathbf{h}^{l+1} = f_l(\mathbf{h}^l) \text{ và vì vậy } \mathbf{o} = f_L \circ \ldots, \circ f_1(\mathbf{x}).$$

- Nếu tất cả giá trị kích hoạt và đầu vào là vector, ta có thể viết lại gradient của o theo một tập tham số W_l bất kỳ như sau:

$$\partial_{\mathbf{W}_l} \mathbf{o} = \underbrace{\partial_{\mathbf{h}^{L-1}} \mathbf{h}^L}_{:=\mathbf{M}_L} \cdot \ldots \cdot \underbrace{\partial_{\mathbf{h}^l} \mathbf{h}^{l+1}}_{:=\mathbf{M}_l} \underbrace{\partial_{\mathbf{W}_l} \mathbf{h}^l}_{:=\mathbf{v}_l}.$$

- => Nếu nhân với một giá trị đạo hàm quá nhỏ thì việc học trở nên bất khả thi khi các tham số hầu như không thay đổi ở mỗi bước cập nhật (tiêu biến gradient).



Khởi tạo trọng số

Khởi tạo trọng số tốt có thể hạn chế Tiêu biến gradient.

- Khởi tạo mặc định
  Net.initialize(init.Normal(sigma=0.01))

- Khởi tạo Xavier

Chuẩn hóa theo batch (Batch Normalization)

- Khi huấn luyện các mạng học sâu, các giá trị kích hoạt ở các tầng trung gian có thể nhận các giá trị với mức độ biến thiên lớn- dọc theo các tầng từ đầu vào đến đầu ra, qua các nút ở cùng một tầng, và theo thời gian do việc cập nhật giá trị tham số
- Sự thay đổi trong phân phối của những giá trị kích hoạt có thể cản trở sự hội tụ của mạng. Dễ thấy rằng nếu một tầng có các giá trị kích hoạt lớn gấp 100 lần so với các tầng khác, thì cần phải có các điều chỉnh bổ trợ trong tốc độ học

Chuẩn hóa theo batch (Batch Normalization)

Chuẩn hoá theo batch được áp dụng cho từng tầng riêng lẻ (hoặc có thể cho tất cả các tầng). Việc chuẩn hóa được thực hiện theo từng vòng lặp, tại từng tầng:

- Tại mỗi tầng: tính giá trị kích hoạt như thường lệ
- Chuẩn hóa những giá trị kích hoạt của mỗi nút:

$$\mathrm{BN}(\mathbf{x}) = \gamma \odot \frac{\mathbf{x} - \hat{\mu}}{\hat{\sigma}} + \beta$$

Trong đó, $\hat{\mu}$ là giá trị trung bình và $\hat{\sigma}$ là độ lệch chuẩn của các mẫu trong minibatch, γ là hệ số tỷ lệ, β là độ lệch

Chuẩn hóa theo batch (Batch Normalization)

```
net = nn.Sequential()
net.add(nn.Conv2D(6, kernel_size=5),
        BatchNorm(6, num_dims=4),
        nn.Activation('sigmoid'),
        nn.MaxPool2D(pool_size=2, strides=2),
        nn.Conv2D(16, kernel_size=5),
        BatchNorm(16, num_dims=4),
        nn.Activation('sigmoid'),
        nn.MaxPool2D(pool_size=2, strides=2),
        nn.Dense(120),
        BatchNorm(120, num_dims=2),
        nn.Activation('sigmoid'),
        nn.Dense(84),
        BatchNorm(84, num_dims=2),
        nn.Activation('sigmoid'),
        nn.Dense(10))
```

Nội dung

1. Phương pháp hạn chế suy giảm đạo hàm
2. Phương pháp tinh chỉnh quá trình huấn luyện
3. Phương pháp huấn luyện chuyển tiếp

Kỹ thuật chuẩn một, chuẩn hai

Thêm vào hàm mất mát một số hạng nữa:

- Số hạng này thường dùng để đánh giá độ phức tạp của mô hình
- Số hạng này càng lớn, thì mô hình càng phức tạp

Hàm mất mát mới này thường được gọi là **regularized loss function**:

$$f_{reg}(w)=f(w) + \lambda R(w)$$

- w: tham số trong mô hình
- f(w) là hàm mất mát (loss function) phụ thuộc vào training set và w
- R(w) là số hạng regularization chỉ phụ thuộc vào w
- λ: tham số regularization là số vô hướng, thường là một số dương nhỏ để đảm bảo nghiệm của bài toán tối ưu $f_{reg}(w)$ không quá xa nghiệm của bài toán tối ưu f(w)

Kỹ thuật chuẩn một, chuẩn hai

Tối thiểu *regularized loss function* đồng nghĩa với việc tối thiểu cả *loss function* và số hạng *regularization*.

$$f_{reg}(w) = f(w) + \lambda R(w)$$

Tối thiểu — Tối thiểu

LASSO regularization

Norm 1 là tổng các trị tuyệt đối của tất cả các phần tử. Đã chứng minh được rằng tối thiểu norm 1 sẽ dẫn tới nghiệm có nhiều phần tử bằng 0

Khi chọn

$$R(\mathbf{W}) = \|\mathbf{w}\|_1 = \sum_{i=0}^{d} |w_i|$$

Nghiệm w tìm được của bài toán tối thiểu $f_{reg}(w)$ có xu hướng rất nhiều phần tử bằng không. Hàm LASSO regression

$$\mathbf{f_{reg}}(\mathbf{w}) = \mathbf{f}(\mathbf{w}) + \lambda\|\mathbf{w}\|_1$$

- Các thành phần khác không của w tương ứng với các đặc trưng quan trọng trong dự đoán đầu ra
- Các thành phần bằng không của w tương ứng với các đặc trưng được coi là ít quan trọng

=> ***LASSO regression giúp lựa chọn đặc trưng hữu ích cho mô hình***

Ridge regularization

Khi chọn R(w) là Norm 2

$$\mathbf{R(w)} = \|\mathbf{w}\|_2^2$$

Ta có hàm Ridge regularization

$$\mathbf{f_{reg}(w)} = \mathbf{f(w)} + \lambda\|\mathbf{w}\|_2^2$$

Hàm $\mathbf{R(w)} = \|\mathbf{w}\|_2^2$ khiến các hệ số trong w không quá lớn, giúp tránh việc đầu ra phụ thuộc quá nhiều vào một đặc trưng nào đó

Ridge regularization với Hồi quy tuyến tính

Hàm mất mát của Hồi quy tuyến tính:

$$l(\mathbf{w}, b) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b - y^{(i)}\right)^2$$

Hồi quy được chuẩn hóa L2:

$$l(\mathbf{w}, b) + \frac{\lambda}{2}\|\mathbf{w}\|^2.$$

Cập nhật gradient ngẫu nhiên:

$$\mathbf{w} \leftarrow (1 - \eta\lambda)\,\mathbf{w} - \frac{\eta}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} \mathbf{x}^{(i)}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b - y^{(i)}\right)$$

Kỹ thuật loại bỏ một số nơ ron (Drop out)

- Kỹ thuât Drop out là loại bỏ một số nơ-ron trong quá trình huấn luyện. Tại mỗi vòng lặp huấn luyện, phương pháp dropout sẽ đặt giá trị của một lượng nhất định (thường là 50%) các nút trong mỗi tầng về không, trước khi tính toán các tầng kế tiếp

Kỹ thuật loại bỏ một số nơ ron (Drop out)

```
# dropout in the input layer with weight constraint
def create_model():
 # create model
 model = Sequential()
 model.add(Dropout(0.2, input_shape=(60,)))
 model.add(Dense(60, activation='relu', kernel_constraint=MaxNorm(3)))
 model.add(Dense(30, activation='relu', kernel_constraint=MaxNorm(3)))
 model.add(Dense(1, activation='sigmoid'))
 # Compile model
 sgd = SGD(learning_rate=0.1, momentum=0.9)
 model.compile(loss='binary_crossentropy', optimizer=sgd, metrics=['accuracy'])
 return model
```

Kỹ thuật loại bỏ một số nơ ron (Drop out)

```
# dropout in hidden layers with weight constraint
def create_model():
 # create model
 model = Sequential()
 model.add(Dense(60, input_shape=(60,), activation='relu', kernel_constraint=MaxNorm(3)))
 model.add(Dropout(0.2))
 model.add(Dense(30, activation='relu', kernel_constraint=MaxNorm(3)))
 model.add(Dropout(0.2))
 model.add(Dense(1, activation='sigmoid'))
 # Compile model
 sgd = SGD(learning_rate=0.1, momentum=0.9)
 model.compile(loss='binary_crossentropy', optimizer=sgd, metrics=['accuracy'])
 return model
```

Nội dung

1. Phương pháp hạn chế suy giảm đạo hàm
2. Phương pháp tinh chỉnh quá trình huấn luyện
3. **Phương pháp huấn luyện chuyển tiếp**

Huấn luyện chuyển tiếp (Transfer Learning)

- Quá trình áp dụng tri thức đã được học từ một mô hình trước (pretrained-model) sang bài toán hiện tại được gọi là transfer learning
- Transfer learning sẽ tận dụng lại các đặc trưng được học từ những pretrained-model