Notas de Aula - Semana 5

Técnicas de Regularização em Redes Neurais

Curso de Redes Neurais

Eduardo Adame

10 de setembro de 2025

Sumário

1 Introdução

O treinamento bem-sucedido de redes neurais requer não apenas algoritmos de otimização eficientes, mas também técnicas para prevenir **overfitting** - quando o modelo memoriza os dados de treinamento em vez de aprender padrões generalizáveis. Este documento explora as principais técnicas de regularização, com foco especial em dropout, uma das inovações mais importantes em deep learning.

A regularização é fundamental porque redes neurais modernas frequentemente têm milhões de parâmetros, tornando-as propensas a memorizar dados de treinamento. Sem regularização adequada, mesmo com algoritmos de otimização sofisticados, os modelos falham em generalizar para dados não vistos.

2 O Problema do Overfitting

2.1 Caracterização do Overfitting

Definição 2.1: Overfitting

Overfitting ocorre quando um modelo se ajusta excessivamente aos dados de treinamento, capturando ruído e particularidades específicas em vez de padrões gerais. Formalmente, seja J_{train} o erro de treinamento e J_{test} o erro de teste. Dizemos que há overfitting quando:

$$J_{test} - J_{train} > \epsilon$$

para algum limiar $\epsilon > 0$ significativo.

Teorema 2.1: Trade-off Viés-Variância

O erro de generalização de um modelo pode ser decomposto em:

$$\mathbb{E}[(y - \hat{f}(x))^2] = \text{Bias}^2[\hat{f}(x)] + \text{Var}[\hat{f}(x)] + \sigma^2$$

onde:

- Bias: erro devido a suposições simplificadoras do modelo
- Variância: erro devido à sensibilidade a flutuações nos dados
- σ^2: ruído irredutível nos dados

2.2 Diagnóstico de Overfitting

Observação 2.1: Sinais de Overfitting

Indicadores práticos de overfitting incluem:

1. **Divergência de curvas**: Erro de treinamento diminui enquanto erro de validação aumenta
2. **Pesos grandes**: $\|W\|$ cresce descontroladamente
3. **Sensibilidade a perturbações**: Pequenas mudanças na entrada causam grandes mudanças na saída
4. **Memorização**: Modelo consegue "decorar"labels aleatórios

3 Regularização L2 e L1

3.1 Penalização de Norma

Definição 3.1: Regularização por Penalização

A regularização por penalização adiciona um termo à função de custo que penaliza a complexidade do modelo:

$$J_{reg}(\theta) = J_{original}(\theta) + \lambda \cdot R(\theta)$$

onde $\lambda > 0$ é o hiperparâmetro de regularização e $R(\theta)$ é o termo de regularização.

3.2 Regularização L2 (Ridge)

Definição 3.2: Regularização L2

A regularização L2 penaliza a norma euclidiana ao quadrado dos pesos:

$$R_{L2}(W) = \frac{1}{2}\sum_{i,j} W_{ij}^2 = \frac{1}{2}\|W\|_2^2$$

A função de custo completa fica:

$$J_{L2} = J_{original} + \frac{\lambda}{2}\sum_{l=1}^{L} \|W^{(l)}\|_2^2$$

onde L é o número de camadas.

Teorema 3.1: Efeito da Regularização L2

A atualização de pesos com regularização L2 pode ser escrita como:

$$W^{(t+1)} = W^{(t)} - \alpha \frac{\partial J_{original}}{\partial W} - \alpha \lambda W^{(t)}$$

Rearranjando:

$$W^{(t+1)} = (1 - \alpha\lambda) W^{(t)} - \alpha \frac{\partial J_{original}}{\partial W}$$

Isso mostra que L2 causa "weight decay- os pesos decaem exponencialmente na ausência de gradiente.

3.3 Regularização L1 (Lasso)

Definição 3.3: Regularização L1

A regularização L1 penaliza a norma L1 (Manhattan) dos pesos:

$$R_{L1}(W) = \sum_{i,j} |W_{ij}| = \|W\|_1$$

Observação 3.1: L1 vs L2

Comparação entre regularização L1 e L2:

- **L2**: Reduz uniformemente todos os pesos, mantendo proporções
- **L1**: Força esparsidade, zerando pesos menos importantes
- **Gradiente L2**: λW (linear nos pesos)
- **Gradiente L1**: $\lambda \cdot \text{sign}(W)$ (constante exceto em zero)

4 Dropout

4.1 Conceito Fundamental

Definição 4.1: Dropout

Dropout é uma técnica de regularização que, durante o treinamento, desativa aleatoriamente neurônios com probabilidade p (dropout rate). Seja $\mathbf{r}^{(l)} \sim \text{Bernoulli}(1-p)$ um vetor de máscaras binárias. A ativação com dropout é:

$$\tilde{\mathbf{a}}^{(l)} = \mathbf{r}^{(l)} \odot \mathbf{a}^{(l)}$$

onde $\odot$ denota produto elemento a elemento (Hadamard).

Teorema 4.1: Dropout como Ensemble

Dropout pode ser interpretado como treinamento de um ensemble exponencialmente grande de redes neurais que compartilham pesos. Para uma rede com n neurônios, dropout treina implicitamente 2^n sub-redes diferentes.

4.2 Implementação do Dropout

Algoritmo 4.1: Dropout Durante Treinamento

Algorithm 1 Forward Pass com Dropout

1: **Entrada:** Ativação $\mathbf{a}^{(l)}$, taxa de dropout p
2: **Fase de Treinamento:**
3: Gerar máscara: $\mathbf{r}^{(l)} \sim \text{Bernoulli}(1-p)$
4: Aplicar máscara: $\tilde{\mathbf{a}}^{(l)} = \mathbf{r}^{(l)} \odot \mathbf{a}^{(l)}$
5: Escalar (inverted dropout): $\tilde{\mathbf{a}}^{(l)} = \frac{\tilde{\mathbf{a}}^{(l)}}{1-p}$
6: Propagar: $\mathbf{z}^{(l+1)} = W^{(l+1)}\tilde{\mathbf{a}}^{(l)} + \mathbf{b}^{(l+1)}$
7: **Retorna:** $\mathbf{z}^{(l+1)}$

Observação 4.1: Inverted Dropout

O "inverted dropout"escala as ativações por $\frac{1}{1-p}$ durante o treinamento, permitindo que a rede seja usada sem modificações durante o teste. Isso é preferível ao dropout tradicional, que requer escalar os pesos por $(1-p)$ durante o teste.

4.3 Análise Matemática do Dropout

Teorema 4.2: Esperança e Variância com Dropout

Seja a uma ativação e $\tilde{a}$ sua versão com dropout (inverted). Então:

$$\mathbb{E}[\tilde{a}] = a \tag{1}$$

$$\text{Var}[\tilde{a}] = \frac{p}{1-p} \cdot a^2 \tag{2}$$

A esperança é preservada, mas a variância aumenta com a taxa de dropout.

Definição 4.2: Dropout como Regularização Adaptativa

Dropout pode ser visto como uma forma de regularização L2 adaptativa com penalização:

$$\lambda_{eff} \approx \frac{p}{1-p} \cdot \|\nabla J\|^2$$

onde a força da regularização se adapta automaticamente ao gradiente local.

4.4 Variantes do Dropout

Observação 4.2: Tipos de Dropout

- **Standard Dropout**: Aplicado a unidades individuais
- **DropConnect**: Aplicado a conexões (pesos) em vez de unidades
- **Spatial Dropout**: Para CNNs, desativa canais inteiros
- **Variational Dropout**: Mesma máscara para toda sequência em RNNs
- **Concrete Dropout**: Taxa de dropout aprendida automaticamente

5 Early Stopping

5.1 Fundamento Teórico

Definição 5.1: Early Stopping

Early stopping interrompe o treinamento quando o erro de validação para de melhorar por k épocas consecutivas (patience). Formalmente, paramos na época t^* onde:

$$t^* = \arg\min_{t \leq T} J_{val}(t) \text{ tal que } J_{val}(s) > J_{val}(t^*) \text{ para } s \in [t^* + 1, t^* + k]$$

Teorema 5.1: Early Stopping como Regularização

Early stopping é equivalente a regularização L2 com parâmetro:

$$\lambda_{eff} \approx \frac{1}{\alpha \cdot t^*}$$

onde α é a taxa de aprendizado e t^* é o número de iterações até a parada.

5.2 Implementação Prática

Algoritmo 5.1: Early Stopping com Checkpoint

Algorithm 2 Early Stopping com Restauração de Melhor Modelo

Inicializar: $best\_loss = \infty$, $patience\_counter = 0$
for época $t = 1$ até max_epochs **do**
  Treinar uma época
  $current\_loss$ = avaliar no conjunto de validação
  if $current\_loss < best\_loss$ **then**
    $best\_loss = current\_loss$
    $best\_weights$ = copiar pesos atuais
    $patience\_counter = 0$
  else
    $patience\_counter = patience\_counter + 1$
  end if
  if $patience\_counter \geq patience$ **then**
    break
  end if
end for
Restaurar $best\_weights$

6 Comparação de Técnicas de Regularização

6.1 Eficácia Relativa

Observação 6.1: Quando Usar Cada Técnica

- **L2**: Sempre útil como baseline, especialmente para redes pequenas
- **L1**: Quando esparsidade é desejada ou para seleção de features
- **Dropout**: Essencial para redes profundas com muitos parâmetros
- **Early Stopping**: Sempre recomendado, custo computacional zero
- **Data Augmentation**: Quando possível no domínio (imagens, áudio)
- **Batch Normalization**: Tem efeito regularizador além da normalização

Exemplo 6.1: Combinação de Técnicas

Para uma rede neural profunda típica, uma estratégia efetiva seria:

1. Early stopping com patience = 10-20 épocas
2. L2 regularization com $\lambda = 10^{-4}$ a 10^{-2}
3. Dropout com $p = 0.5$ nas camadas densas
4. Dropout com $p = 0.2$ nas camadas convolucionais
5. Data augmentation se aplicável

7 Análise Experimental

7.1 Efeito do Dropout Rate

Observação 7.1: Escolha da Taxa de Dropout

Experimentos empíricos sugerem:

- $p = 0.5$ é ótimo para camadas totalmente conectadas
- $p = 0.2$ a 0.3 para camadas convolucionais
- $p = 0.1$ a 0.2 para camadas de entrada
- p maior para redes com mais parâmetros
- Reduzir p gradualmente em camadas mais profundas

Observação 7.2: Limite Superior do Dropout

Para manter a capacidade de aprendizado, a taxa de dropout deve satisfazer:

$$p < 1 - \frac{1}{\sqrt{n}}$$

onde n é o número de neurônios na camada. Para $n = 100$, isso dá $p_{max} \approx 0.9$.

8 Considerações Avançadas

8.1 Dropout e Batch Normalization

Observação 8.1: Interação Dropout-BatchNorm

A ordem de aplicação importa:

1. **Recomendado**: Dense → BatchNorm → Activation → Dropout
2. BatchNorm antes de Dropout estabiliza as distribuições
3. Dropout após ativação preserva não-linearidade

8.2 Regularização Implícita

Definição 8.1: Regularização Implícita

Algumas técnicas têm efeito regularizador não intencional:

- **SGD**: Ruído do mini-batch atua como regularização
- **Batch Normalization**: Adiciona ruído através das estatísticas do batch
- **Data Augmentation**: Aumenta efetivamente o tamanho do dataset
- **Label Smoothing**: Suaviza distribuições de saída

9 Debugging e Diagnóstico

9.1 Validação da Implementação

Observação 9.1: Testes de Sanidade

Para verificar se a regularização está funcionando:

1. **Sem regularização**: Modelo deve conseguir overfit em dataset pequeno
2. **Dropout = 1**: Rede deve falhar completamente (todas unidades desligadas)
3. **L2 muito alto**: Performance deve degradar significativamente
4. **Monitorar normas**: $\|W\|$ deve ser menor com regularização

9.2 Métricas de Diagnóstico

Definição 9.1: Gap de Generalização

O gap de generalização é definido como:

$$\text{Gap} = \frac{J_{test} - J_{train}}{J_{train}}$$

Valores típicos:

- Gap < 0.1: Boa generalização
- Gap $\in [0.1, 0.3]$: Regularização moderada necessária
- Gap > 0.3: Forte regularização necessária

10 Conclusões e Recomendações

10.1 Estratégia Prática

Algoritmo 10.1: Pipeline de Regularização

1. Começar sem regularização para estabelecer baseline
2. Adicionar early stopping (sempre)
3. Se overfitting persistir, adicionar L2 com $\lambda = 10^{-4}$
4. Aumentar λ gradualmente se necessário
5. Adicionar dropout começando com $p = 0.2$
6. Ajustar dropout por camada se necessário
7. Considerar data augmentation se aplicável
8. Fine-tuning final dos hiperparâmetros

10.2 Trade-offs

Observação 10.1: Considerações Práticas

- **Tempo de treinamento**: Dropout aumenta em 2-3x
- **Memória**: L2 não adiciona custo, Dropout adiciona máscaras
- **Inferência**: L2 não afeta, Dropout precisa ser desligado
- **Interpretabilidade**: L1 produz modelos mais interpretáveis

A regularização é essencial para o sucesso de redes neurais profundas. A combinação cuidadosa de diferentes técnicas, ajustada ao problema específico, é fundamental para ob-

ter modelos que generalizam bem. O dropout, em particular, revolucionou o treinamento de redes profundas e continua sendo uma das técnicas mais importantes em deep learning moderno.