Inteligência Artificial

Relatório do Problema 4
2 ou 3?
2025/26

Imagem gerada por AI

Grupo 4	
Nome	Número
André Martins	35463
António Matoso	84100
Tomás Machado	84014

Índice

Descrição do Problema e Algoritmos Utilizados

Descrição do Problema

O objetivo deste projeto é desenvolver um classificador binário capaz de distinguir dígitos manuscritos "2" e "3" do dataset MNIST. Este é um problema de classificação supervisionada onde cada imagem de 20×20 pixels (400 *features*) deve ser classificada como pertencente a uma das duas classes.

Características do Dataset

- Imagens de entrada: 20×20 pixels (400 features normalizadas em [0,1]).
- Classes: 2 (dígito "2") e 3 (dígito "3").
- Codificação das labels: 0.0 para "2", 1.0 para "3".
- Divisão: 80% treino, 20% teste.

Algoritmos Implementados

Técnicas de Otimização

Momentum Clássico (β = 0.9)

O momentum clássico acumula parte do update anterior para acelerar a convergência e reduzir oscilações:

$$\Delta w[k] = \eta\delta[k] + \alpha\Delta w[k-1], \qquad \alpha \in [0,1]$$

Nota: $\delta[k]$representa o gradiente da função de custo em relação aos pesos na iteração k.

Vantagens:

- Acelera a descida do gradiente em direções consistentes.
- Reduz oscilações em vales íngremes.

Gradient Clipping (norm = 2.0)

$$\|g\| = \sqrt{\sum_i g_i^2}$$

Caso $\|g\| > norm$ (2.0), o gradiente é reescalado.
Vantagens:

- Garante treino estável**,** evitando saltos enormes nos pesos.
- Permite usar learning rates maiores sem risco de instabilidade.

OneCycleLR (pctUp =0.3)

O learning rate varia durante o treino numa "subida rápida" seguida de "descida gradual":

- **Fase de warm-up:** lr aumenta de lr/10 a 3×lr
- **Fase de cool-down:** lr diminui de 3×lr a lr/100
- pctUp = 0.3, isto é, 30% do treino é warm-up, 70% é cool-down

Vantagens:

- Convergência mais rápida nas primeiras épocas (warm-up).
- Melhor estabilidade nas últimas épocas (cool-down).
- Pode ajudar a escapar de mínimos locais**.**
- Reduz a necessidade de tuning manual do learning rate.

Early Stopping Adaptativo

O treino é interrompido automaticamente quando o Mean Squared Error (MSE) no conjunto de teste não melhora por *n* épocas consecutivas. Nesse caso, os pesos da rede são restaurados para aqueles que tiveram o menor MSE, garantindo que o modelo final corresponde ao melhor desempenho observado durante o treino.

Onde:

$$MSE = \frac{1}{n}\sum(y\_true - y\_pred)^2$$

A arquitetura da Rede Utilizada

Arquitetura Final Escolhida

Topologia: 400 → 256 → 1

Justificação:

- **Camada de entrada (400 neurônios)**: Um neurônio por pixel da imagem 20×20.
- **Camada oculta (256 neurônios)**:
  - Capaz de capturar features complexas sem ser excessiva, evitando overfitting.
  - Equilíbrio entre capacidade de representação e generalização.
- **Camada de saída (1 neurônio)**:
  - Classificação binária.
  - Output via sigmoid em [0,1].

Inicialização de Pesos

He Initialization:

$W \sim \mathcal{N}(0, \sigma^2)\ onde\ \sigma = \sqrt{\frac{2}{n\_input}}$, otimizada para ReLU/Leaky ReLU, prevenindo gradientes muito pequenos ou muito grandes.

Bias Initialization:

$b \sim U(-0.01, 0.01)$, pequenos valores aleatórios para quebrar simetria.

Hiperparâmetros Finais

- Learning Rate inicial: 0.01 (varia dinamicamente com OneCycleLR, mínimo 0.001).
- Momentum clássico: 0.9.
- Batch Size: 64 (ajustado dinamicamente até 256 durante treino).
- Max Epochs: 16000.
- Paciência (Early Stopping): 800 épocas sem melhora no MSE.
- Gradient Clip Norm: 2.0.

Opções de Design Consideradas

Arquitetura

Decisão 1: Número de Camadas Ocultas

Opções consideradas:

- **1** camada oculta
- 2 camadas ocultas
- 3+ camadas ocultas

Justificação: Para um problema de classificação binária relativamente simples (distinguir 2 vs 3), uma única camada oculta com neurônios suficientes é capaz de aprender as *features* necessárias. Camadas adicionais aumentariam risco de overfitting e tempo de treino.

Decisão 2: Número de Neurônios

Testamos: 48, 64, 128, 256, 512 neurônios.

- **Análise:**
  - 48: muito restritivo, capacidade limitada.
  - 64: bom balanço entre capacidade e generalização.
  - 128: desempenho adequado.
  - **256:** melhor capacidade para capturar features complexas, sem causar overfitting significativo, mas treino lento.
  - 512: overfitting e treino muito lento.
- **Escolha final:** 256 neurônios.

Funções de Ativação

Função	Vantagens	Desvantagens
ReLU	Rápida, sem *vanishing gradient*	Neurônios mortos
Leaky ReLU	Resolve neurônios mortos	Ligeiramente mais complexa

Função	Vantagens	Desvantagens
Sigmoid	Smooth	*Vanishing gradient* forte

Leaky ReLU (Rectified Linear Unit) nas camadas ocultas:

$$\begin{cases} f(x) = x & se\ x > 0 \\ f(x) = \alpha x & se\ x \leq 0 \end{cases}$$

$$\begin{cases} f(x) = 1.0 & se\ x > 0 \\ f(x) = \alpha & se\ x \leq 0 \end{cases}$$

A escolha: Leaky ReLU (α = 0.01), escolhida por evitar neurônios mortos e *vanishing gradients*, garantindo um treino mais estável e rápido.

Camada de Saída:

Função	Adequação
Sigmoid	Ideal para probabilidades [0,1]

Sigmoid na camada de saída:

$$\begin{cases} \sigma(x) = \dfrac{1}{(1 + e^{-x})} \\ \sigma'(x) = \sigma(x) \times (1 - \sigma(x)) \end{cases}$$

A escolha: Sigmoid, adequada para classificação binária, produzindo saída em [0,1] interpretável como probabilidade.

Data Augmentation

Decisão 1: Intensidade

Um nível excessivo de data augmentation pode gerar amostras demasiado diferentes dos dados de teste, prejudicando a capacidade de generalização do modelo.
Por outro lado, uma intensidade insuficiente de augmentation pode levar a *overfitting*, reduzindo a robustez do modelo.

Decisão 2: Técnicas

- **Noise**: Adiciona pequenas perturbações nos pixels, ajudando o modelo a torna-se mais robusto.
- **Elastic**: Muito eficaz para dados manuscritos, pois simula variações naturais do traço.
- **Rotation**: Essencial para garantir coerência a pequenas rotações dos dígitos.
- **Shift**: Simula variações de posição do dígito dentro da imagem.
- **Brightness**: Ajuda a tornar o modelo robusto a variações de iluminação na digitalização.
- **Scaling:** Ajusta o tamanho do dígito dentro da imagem, melhorando a generalização para diferentes proporções.

- **Contrast:** Modifica o contraste da imagem, tornando o modelo menos sensível a diferentes condições de fotografia.

Decisão Final

Com base nestas considerações, decidiu-se utilizar:

- 1 cópia uma combinação Gaussian noise, com desvio padrão de 0.02, com *scaling* de 0.9 a 1.1 e *contrast* de 0.9 a 1.1**.**
- 1 cópia com uma combinação de *rotation* de 6.0 graus e *shift* de 1 pixel.
- 1 cópia com uma combinação de *elastic***,** com parâmetros $\alpha = 6.0$ e $\sigma = 2.0$ e *brightness* aleatório uniforme no intervalo [0.85, 1.15].f

Esta configuração permitiu melhorar a capacidade de generalização do modelo sem introduzir distorções excessivas nos dados de treino. Todas as transformações foram aplicadas com intensidade moderada e em número limitado de cópias, garantindo que as amostras augmentadas permanecem próximas da distribuição dos dados originais.

Batch Size

Opções Testadas

- 32**:** Gradientes mais ruidosos, levando a convergência instável.
- 64: Bom compromisso entre estabilidade e velocidade de convergência.
- 128**:** Treino mais estável, mas com aumento do tempo por época.
- 256+: Treino significativamente mais lento e maior consumo de memória, sem ganhos claros de desempenho.
- **Variável (adaptativo)**: Testado como alternativa, e demonstrou ser o melhor.

Decisão Final

Optou-se por uma estratégia adaptativa com **limite superior de 256**, permitindo beneficiar da estabilidade de batches maiores nas fases finais do treino, mantendo diversidade suficiente nos gradientes nas fases iniciais.

Threshold de Decisão

Opções

- Fixo em 0.5
- **Otimizado via *grid search***

Decisão Final

O threshold é otimizado automaticamente testando valores de 0.1 a 0.9 com *step* de 0.05, escolhendo o que maximiza *accuracy* no conjunto de validação.

Resultados, Análise e Discussão

Métricas de Avaliação

As seguintes métricas foram utilizadas para avaliar o desempenho:

1. Exatidão:

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)}$$

2. Precisão:

$$Precision = \frac{TP}{(TP + FP)}$$

3. Revocação/Sensibilidade:

$$Recall = \frac{TP}{(TP + FN)}$$

4. F-Measure:

$$F = 2\frac{(Precision \times Recall)}{(Precision + Recall)}$$

Onde:

- TP (True Positives): Corretamente classificados como '3'
- TN (True Negatives): Corretamente classificados como '2'
- FP (False Positives): '2' classificados como '3'
- FN (False Negatives): '3' classificados como '2'

Resultados Experimentais Locais

Melhor Configuração do Mooshak:

```
===Training Results===
Best Epoch: 228
Best Test MSE: 0.005936
=== Prediction Distribution ===
Count: 3268
Min prediction: 0.000000
Max prediction: 1.000000
Avg prediction: 0.492902
=== Threshold Optimization ===
Best threshold: 0.45

=== Evaluation Metrics ===
Accuracy:  99.33%
Precision: 99.38%
Recall:    99.26%
F-Measure: 0.9932
Confusion Matrix:
              Predicted 0  Predicted 1
Actual 0:     1644           10
Actual 1:     12            1602
```

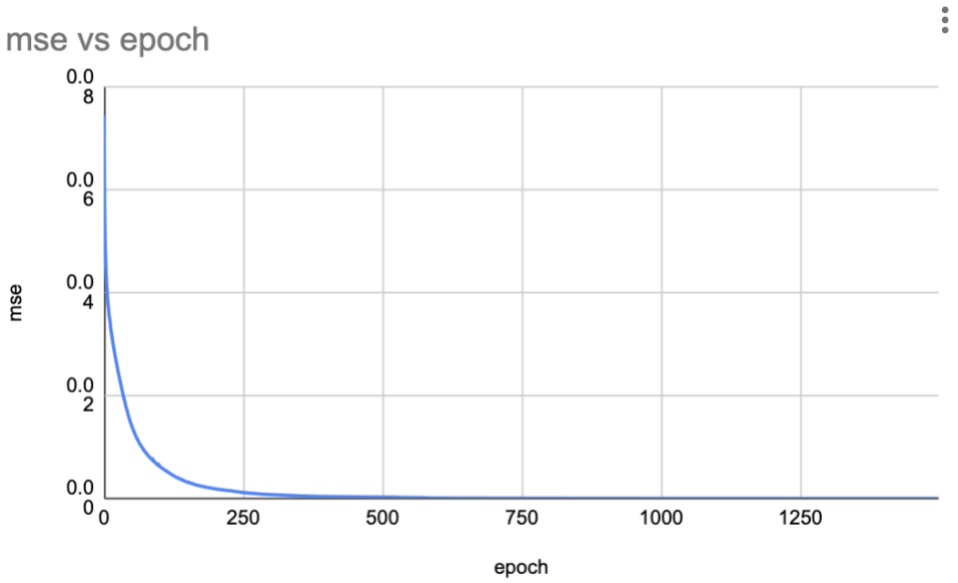
Análise Geral

Os resultados obtidos demonstram que o modelo apresenta elevado desempenho e excelente capacidade de generalização, evidenciada por uma accuracy de 99.330% e um F-Measure de 0.9932. A análise da confusion matrix confirma um número muito reduzido de erros de classificação, bem como um equilíbrio adequado entre falsos positivos e falsos negativos. O baixo MSE no conjunto de teste reforça a estabilidade do treino e valida a eficácia das opções de arquitetura e das técnicas de data augmentation aplicadas.

Análise da Confusion Matrix

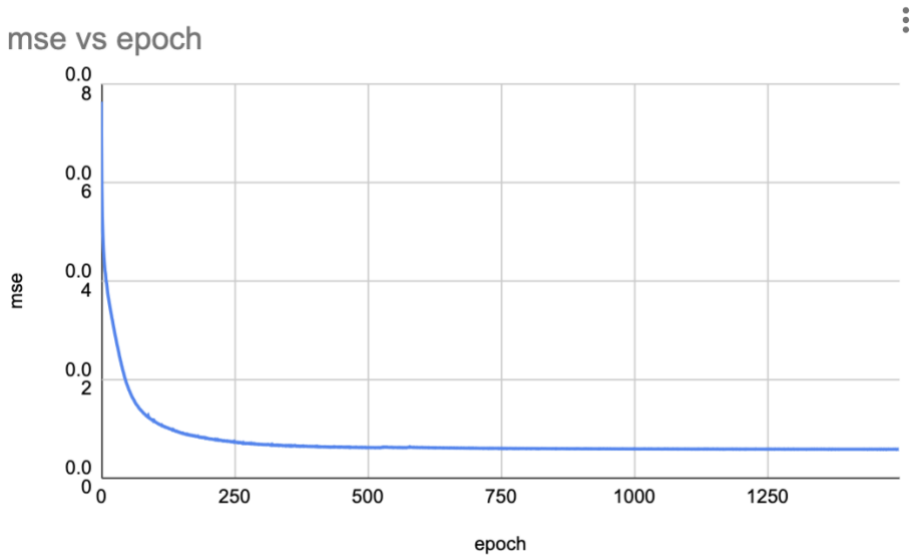
- **True Negatives (TN)**: 1644 dígitos '2' corretamente classificados
- **False Positives (FP)**: 12 dígito '2' classificado como '3'
- **False Negatives (FN)**: 10 dígitos '3' classificados como '2'
- **True Positives (TP)**: 1602 dígitos '3' corretamente classificados

Gráfico MSE no Conjunto de Treino



O gráfico do MSE de treino evidência uma convergência rápida nas primeiras épocas, seguida de uma redução progressivamente mais lenta até atingir um valor próximo de zero. A ausência de oscilações indica um processo de treino estável, com parâmetros adequadamente escolhidos, após cerca de 300 épocas, os ganhos tornam-se pequenos, demonstrando que a rede atinge o seu limite de capacidade e que o early stopping poderia ser aplicado, que não haveria uma perda significativa de desempenho.

Gráfico MSE no Conjunto de Teste



O gráfico do MSE no conjunto de teste evidência uma redução significativa nas primeiras épocas, demonstrando boa capacidade de generalização do modelo. Após cerca de 100-150 épocas, o erro estabiliza, indicando que o modelo atinge o seu limite de desempenho no conjunto de teste, a ausência de aumento do MSE ao longo do treino sugere que não ocorre overfitting, sendo o treino adicional após esse ponto desnecessário.

Validação com Múltiplas Seeds

Para garantir reprodutibilidade e robustez, testamos com 14 seeds diferentes:

Seeds: {42, 97, 123, 456, 789, 1337, 2023, 9999, 314159, 271828, 123456, 424242, 8675309}, sendo a que mais se destacou para o modelo apresentado foi a seed 2023.

Avaliação no Mooshak

A diferença entre os resultados locais, 99.330%, e os obtidos no Mooshak, 97.000%, deve se ao facto de o conjunto de teste do segundo conter variações não observadas durante o treino local, mas que mesmo assim permitiu ter um desempenho competitivo, realçando assim a importância das técnicas de melhor convergência e data augmentation implementadas.

Conclusão

Lições Aprendidas

1. **A simplicidade com boas práticas vence:** uma topologia simples, quando combinada com inicialização apropriada (He), Leaky ReLU, adaptação do learning rate e regularização funcional, produz resultados competitivos.
2. **O momentum clássico é eficiente:** acelera a convergência ao incorporar parte do update anterior, estabilizando o treino.
3. **Aprender a controlar augmentations:** gera variantes do dataset, melhorando a generalização.
4. **A monitorização é crucial:** early stopping e restauro dos melhores pesos evitam overfitting**.**

Considerações Finais

Pelo que vimos, para o MNIST 2 vs 3, não é necessário complicar demasiado, um MLP simples chega e funciona bem, o que realmente ajuda são alguns detalhes, como inicializar bem os pesos, usar Leaky ReLU, ajustar o learning rate com One-Cycle, aplicar clipping de gradiente e usar early stopping, aumentar a rede não muda muito por si só, e por vezes pode piorar se os parâmetros nao forem adequados.

Para concluir, este trabalho mostra nos que em problemas “simples” como este, não é a profundidade ou a complexidade da rede que faz a diferença, porque o que conta mesmo são as decisões de treino e a forma como regularizamos o modelo para não exagerar e chegar o mais perto possível da perfeição.

Referências bibliográficas

Artigos Científicos

1. **Glorot, X., & Bengio, Y. (2010).** "Understanding the difficulty of training deep feedforward neural networks." *Proceedings of AISTATS*, 9, 249-256.
    a. Xavier/Glorot initialization

2. **He, K., Zhang, X., Ren, S., & Sun, J. (2015).** "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification." *Proceedings of ICCV*, 1026-1034.
    a. He initialization e análise de ReLU
3. **Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014).** "Dropout: A simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research*, 15(1), 1929-1958.
    a. Paper original sobre Dropout
4. **Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013).** "Rectifier nonlinearities improve neural network acoustic models." *Proceedings of ICML*, 30(1), 3.
    a. Leaky ReLU e variantes

Livros

1. **Simard, P. Y., Steinkraus, D., & Platt, J. C. (2003).** "Best practices for convolutional neural networks applied to visual document analysis." *Proceedings of ICDAR*, 958-963.
    a. Elastic distortions para MNIST
2. **Cubuk, E. D., Zoph, B., Mane, D., Vasudevan, V., & Le, Q. V. (2019).** "AutoAugment: Learning augmentation strategies from data." *Proceedings of CVPR*, 113-123.
    a. Automated data augmentation