Projeto IC

Artigo Escolhido

Artigo

Dataset

UrbanSound8k

Objetivo

Este artigo foca em melhorar a modelagem acústica utilizando redes neurais convolucionais (CNNs) diretamente com dados de waveform (forma de onda) bruta, sem pré-processamento significativo. Mas, o que é modelagem acústica? A modelagem acústica é uma técnica que utiliza a computação para caracterizar indicadores de poluição sonora.
A modelagem acústica a partir de dados de waveform bruta é desafiadora devido à complexidade das sequências de dados (como vetores de tamanho 32000).

Principais Elementos

- **Número de Camadas:** A arquitetura mais profunda possui até **34 camadas** de pesos, sendo uma CNN muito profunda.
- **Número de Neurônios por Camada:**
  - Primeira camada: **80/4, 256 filtros**.
  - Segunda camada: **80/4, 48 filtros**.
  - As camadas seguintes usam filtros menores com números variáveis, como **3, 384**, **3, 192**, etc., conforme mostrado nas tabelas do artigo.
- **Taxa de Aprendizado:** O artigo utiliza o otimizador **Adam**, que ajusta dinamicamente a taxa de aprendizado. A taxa inicial de aprendizado padrão no Adam é **0.001**
- **Momentum:** O Adam, utilizado no estudo, não utiliza momentum tradicional como o SGD, mas ajusta dinamicamente os parâmetros de aprendizado, então o "momentum" é controlado por uma adaptação interna no Adam .
- **Dataset:** Dataset utilizado foi o UrbanSound8k, especificado acima.
- **Métricas (Precisões etc.):** O modelo M18 atinge uma precisão de **71.68%** no conjunto de testes, sendo o melhor desempenho observado .

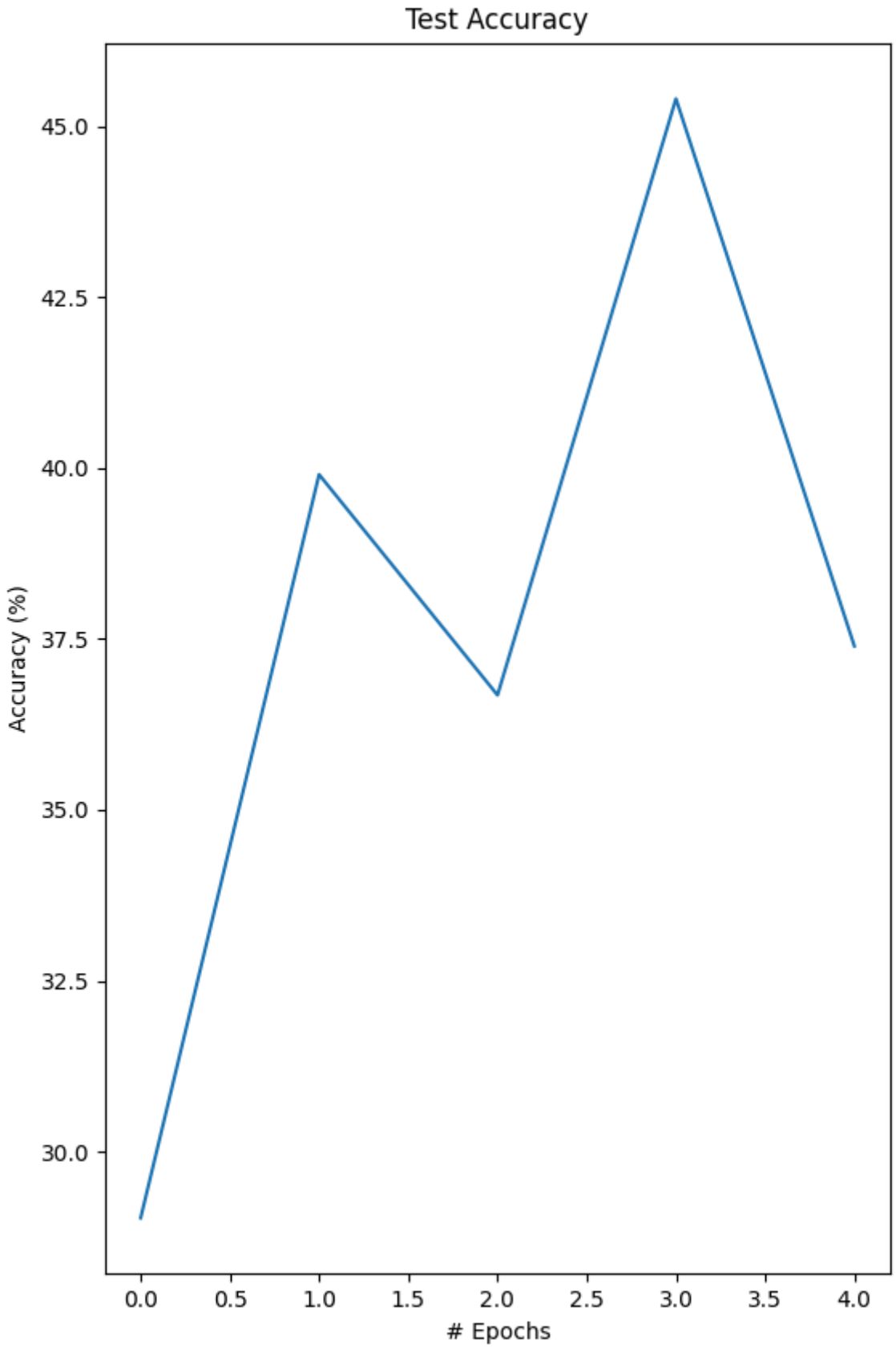
Código

O código, está presente no GitHub e foi desenvolvido utilizando as especificações do autor, o código não é o código original do autor, justamente por não ter sido disponibilizado pelo mesmo, porém, o código encontrado segue a risca tudo no artigo, foi modificado para satisfazer as nossas necessidades.

Experimentos do Autor

Model	Test	Time
M3	56.12%	77s
M5	63.42%	63s
M11	69.07%	71s
M18	71.68%	98s
M34-res	63.47%	124s

Como visto no artigo, o autor nos disponibilizou apenas esses dados.



A imagem acima, foi feita utilizando um código "inacabado" pelo autor em seu GitHub.

Experimentos

1º Experimento

```
Acurácia: 0.2009
Precisão: 0.3285
Recall: 0.2009
F1-Score: 0.1803
```

Taxa de aprendizagem de 0,001, 5 epochs e um batch size de 32.

2º Experimento

```
Acurácia: 0.1151
Precisão: 0.0132
Recall: 0.1151
F1-Score: 0.0237
```

Taxa de aprendizagem de 0,005, 5 epochs e um batch size de 32.

3º Experimento

```
Acurácia: 0.0887
Precisão: 0.0079
Recall: 0.0887
F1-Score: 0.0145
```

Taxa de aprendizagem de 0,015, 5 epochs e um batch size de 32.

Conclusão

Como pode ser observado, à medida que a taxa de aprendizagem aumenta, a acurácia dos resultados tende a diminuir. Isso ocorre porque uma taxa de aprendizagem menor, como 0.001, permite atualizações mais suaves nos pesos do modelo. Isso ajuda o modelo a convergir de forma mais estável e a encontrar um mínimo local mais eficaz. Em contraste, uma taxa de aprendizagem mais alta pode causar grandes saltos nos pesos, o que pode resultar em uma perda de estabilidade durante o treinamento e dificultar a convergência para um ótimo desempenho. Devido a limitações computacionais, não explorei muitos parâmetros. O algoritmo, por ter muitas camadas, tornou-se bastante pesado. Enquanto o autor utilizou uma GPU Titan X para realizar os testes, os experimentos realizados aqui

foram feitos com uma RX 5500XT, o que aumentou significativamente o tempo de execução da rede.