국문제목

MNIST 숫자 분류를 위한 경량 심층신경망 모델: 성능 및 효율성 연구

영문제목

Light weighting Deep Neural Networks for MNIST Digit Classification

요 약

최근 다양한 분야에서 심층신경망(deep neural networks)이 놀라운 결과를 보이고 있다. 특히, large language model(LLM)과 같이 신경망 모델의 성능을 향상시키기 위해 파라미터 개수를 증가시키려는 많은 시도와 연구들이 있었다. 이런 방법들은 신경망 모델의 학습에 대한 시간적 및 메모리 복잡도를 증가시키며, 심지어 추론(inference) 시에도 동일한 문제가 발생한다. 이런 비용 문제를 해결하기 위한 접근들 중 하나가 모델 경량화인데 최근 이 접근이 주목을 받고 있다. 본 연구는 MNIST 데이터셋을 활용하여 심층신경망 모델의 경량화를 탐구하였다. ResNet 계열의 잔차 연결 특징을 일부 차용한 CNN 모델로 사용하였으며, 모델 경량화를 위해 프루닝(pruning), 양자화(quantization) 등의 기법을 적용하였다. Stochastic weight averaging(SWA)와 gradient clipping 등 학습 안정화 기법을 병행하여 성능을 개선하였고, 분류 정확도, 모델 크기를 지표로 성능과 효율성을 평가하였다. 실험 결과, 약 99.55%의 분류 정확도를 달성하면서 모델 크기를 크게 감소시켰으며, 자원 제한 환경에서의 적용 가능성을 확인하였다.

1. 서 론

딥러닝은 다양한 컴퓨터 비전 문제에서 탁월한 성능을 입증하며, 이미지 분류, 객체 탐지, 자연어 처리 등 다양한 분야에서 혁신을 이루어왔다. 그중 MNIST 데이터셋은 손글씨 숫자 분류를 위한 대표적인 벤치마크 데이터셋으로, 많은 연구에서 기초 모델 개발과 검증을 위해 사용되고 있다. 그러나 심층신경망 모델의 높은 성능은 종종 큰 연산량과 메모리 요구사항을 수반하며, 이는 리소스가 제한된 엣지 디바이스나 모바일 환경에서의 활용을 제한하는 요인으로 작용한다.

이러한 한계를 극복하기 위해, 경량화된 심층신경망 모델 설계가 중요한 연구 주제로 떠오르고 있다. 모델 경량화는 모델의 크기와 연산량을 줄여 배포 및 추론 효율성을 개선하는 동시에, 정확도를 유지하거나 최소한의 성능 저하만을 허용하는 것을 목표로 한다. 특히, MNIST와 같은 비교적 단순한 데이터셋에서 모델을 경량화하면 고성능을 유지하면서도 리소스 효율성을 극대화할 수 있는 가능성을 탐구할 수 있다.

본 연구에서는 MNIST 데이터셋을 대상으로 심층신경망 모델 경량화를 탐구하며, 다양한 모델 압축 기법 및 경량화 전략이 성능과 효율성에 미치는 영향을 분석한다. 이를 통해 리소스가 제한된 환경에서의 심층신경망 모델 적용 가능성을 제고하고, MNIST 분류를 위한 최적의 경량 모델 설계 방향을 제시하고자 한다.

2. 연구 방법

본 연구에서는 MNIST 손글씨 숫자 데이터셋을 대상으로 심층신경망 모델 경량화 기법을 적용하여 성능과 효율성을 분석하였다. 실험은 캐글(kaggle) 환경에서 진행되었으며, 학습 데이터 42,000개와 테스트 데이터 28,000개를 사용하였다. 학습 데이터는 80%를 학습용으로, 20%를 검증용으로 분리하였으며, 데이터를 4차원 텐서로 변환하여 모델 학습에 활용하였다. 모든 데이터는 0에서 1 사이로 정규화하여 학습 효율을 향상시켰으며, 회전, 이동, 확대/축소와 같은 데이터 증강 기법을 추가적으로 적용하여 모델의 일반화 성능을 강화하였다.

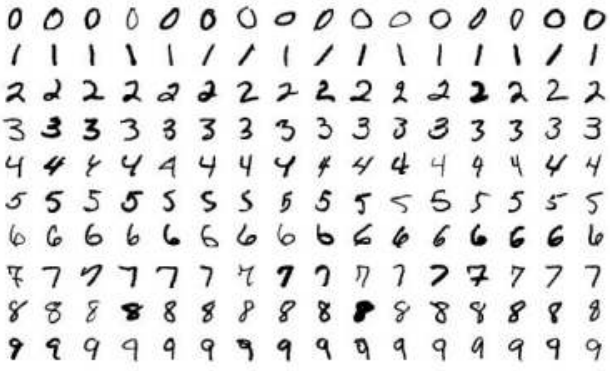

그림 1 MNIST 데이터셋

기본 모델로는 Residual Connection을 일부 포함한 간소화된 CNN 구조를 활용하였다. 이 모델은 ResNet 계열의 잔차 연결(residual connections) 특징을 일부 차용하여 학습 안정성과 성능을 개선하였다. AdamW 옵티마이저를 사용하여 가중치 감쇠(weight decay)를 통해 과적합을 방지하였으며, 초기 학습률은 0.01로 설정하였다. 손실 함수로는 Cross-Entropy Loss를 사용하였으며, 학습은 총 200 epoch 동안 진행되었다. 학습 과정에서는 SWA(Stochastic Weight Averaging)를 적용하여 모델의 일반화를 향상시켰다. SWA는 학습 과정에서 서로 다른 모델의 가중치를 평균화하여 일반화 성능을 강화하는 방법으로, 학습 후반부에서 성능의 안정적 수렴과 높은 일반화 성능을 보장하는 데 사용되었다. 또한, 검증 데이터셋을 사용하여 모델 성능을 지속적으로 평가하였다. SWA 적용 후 Batch Normalization 통계를 갱신하여 모델의 안정성을 더욱 높였다.

모델 경량화를 위해 다양한 기법을 적용하였다. 첫째, SWA 적용이 완료된 모델에 대해 프루닝(pruning)을 통해 각 합성곱 레이어의 가중치 중 50%를 제거함으로써 모델의 희소성을 증가시키고, 계산 효율성을 개선하였다. 둘째, 양자화(quantization)를 통해 모델 파라미터의 데이터 타입을 float32에서 int8로 변환하

여 모델의 크기를 줄이고 메모리 사용량을 최적화하였다.

추가적으로, SGD with Nesterov Momentum(SGD) 옵티마이저를 Stochastic Weight Averaging(SWA)와 함께 적용한 결과를 비교하였다. Nesterov Momentum은 빠른 수렴과 안정적인 학습 과정을 가능하게 하며, SWA와의 조합은 높은 일반화 성능을 보이는 경우가 많아 이를 AdamW와 비교하였다.

또한, Gradient Clipping을 사용하여 그래디언트 폭주(gradient explosion)를 방지하였다. Gradient Clipping은 학습 과정에서 그래디언트의 값이 일정 임계치를 초과하지 않도록 조정하여 학습의 안정성을 보장하였다.

모델 성능 평가는 정확도(accuracy), 모델 크기(model size), 를 주요 지표로 삼았다. 실험 결과는 경량화 적용 전후의 모델을 비교하여 분석하였다. 모든 실험은 Kaggle 환경에서 동일한 데이터셋과 설정으로 수행되어 공정성을 확보하였다.

3. 결과 및 고찰

본 연구에서는 MNIST 데이터셋을 활용하여 심층신경망 모델의 경량화 기법을 적용하고 성능과 효율성을 분석하였다. 캐글 리더보드에서 퍼블릭 스코어 1.0을 목표로 설정하려했으나 일부 시도가 학습 데이터와 테스트 데이터를 분리하지 않고 사용된 비현실적인 설정에서 비롯된 결과임을 확인하였다. 실험 결과, Residual Connection을 일부 포함한 간소화된 CNN 기반 모델은 약 99.55%의 검증 데이터 정확도를 달성하였으며, 이는 MNIST 데이터셋에서 현실적으로 가능한 최대 성능인 99.7%에 근접한 결과이다.

[그림 2]는 MNIST 데이터셋의 실제 성능 한계를 보여주며, 현실적으로 가능한 최대 성능은 약 99.7%로 제한된다. 본 연구는 이러한 성능 한계 내에서 모델의 효율성과 정확도의 균형을 최적화하는 데 중점을 두었으며, 경량화된 모델의 성능 손실을 최소화하면서도 계산 비용을 효과적으로 줄이는 데 성공하였다.

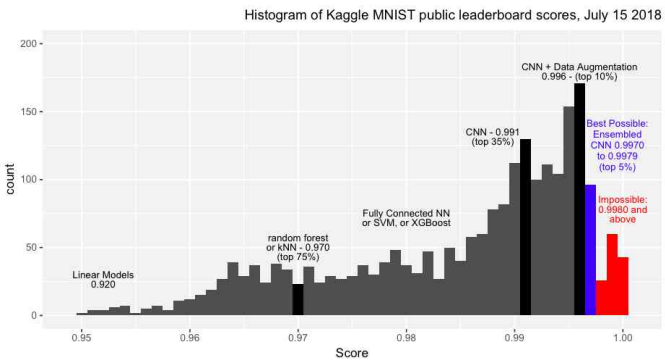

그림 2 2018년 기준 캐글 퍼블릭 스코어

특히, 본 연구에서는 SWA(Stochastic Weight Averaging), 가지치기(Pruning), 동적 양자화(Dynamic Quantization)와 같은 경량화 기법을 결합하여 모델의 크기와 연산 효율성을 대폭 개선하였다. 모델의 original_size는 25.08MB, quantized_size는 6.26MB로 모델은 약 75.04% 경량화되었음을 확인할 수 있다. SGD with Nesterove Momentum을 사용한 결과는 AdamW에 비해 낮은 정확도를 보여주어 코드에는 포함시키지 않았다.

본 연구에서 제안한 경량화 기법(SWA, 프루닝, 양자화)은 모델 크기를 크게 개선하면서도 높은 정확도를 유지하였으며, 리소스 제한 환경에서도 적용 가능한 심층신경망 모델 설계 가능성을 제시하였다. 향후 연구에서는 보다 다양한 데이터셋과 응용 분야에 모델을 확장하여 경량화 기법의 범용성을 검증하고 지식 증류와 같은 추가적인 모델 경량화 기법을 시도할 예정이다.

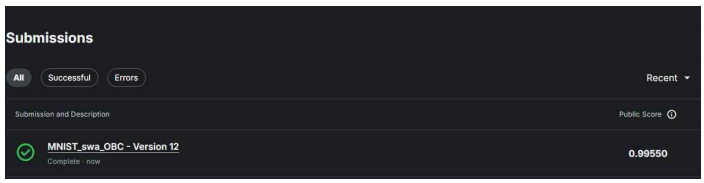

그림 3 캐글 리더보드 퍼블릭 스코어

4. 참고 문헌

1. Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., & Wilson, A. G. "Averaging Weights Leads to Wider Optima in Deep Learning." 2018. https://arxiv.org/abs/1803.05407

2. PyTorch 공식 문서. "stochastic-weight-averaging" https://pytorch.org/blog/stochastic-weight-averaging-in-pytorch/

3. Han, S., Pool, J., Tran, J., & Dally, W. "Learning both Weights and Connections for Efficient Neural Networks." 2015. https://arxiv.org/abs/1506.02626

4. Jacob, B., et al. "Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference." 2018. https://arxiv.org/abs/1712.05877

5. PyTorch 공식 문서. "Torch.quantization." https://pytorch.org/docs/stable/quantization.html

6. Kaggle. "Digit Recognizer Dataset." https://www.kaggle.com/c/digit-recognizer/data

7. LeCun, Y., Cortes, C., & Burges, C. "The MNIST Database." http://yann.lecun.com/exdb/mnist/

8. Frantar, E., Singh, S. P., & Alistarh, D. "Optimal Brain Compression: A Framework for Accurate Post-Training Quantization and Pruning." https://arxiv.org/abs/2210.03887