구간별 선형 근사를 통한 On-device 용 화자 인식 모델 최적화

박재홍, 안성환, 김남수
서울대학교 전기정보공학부 뉴미디어통신공동연구소

{jhpark, shahn}@hi.snu.ac.kr, nkim@snu.ac.kr

Optimization of on-device speaker recognition models through piecewise linear approximation

Jaehong Park, SungHwan Ahn, Nam Soo Kim
Department of Electrical and Computer Engineering and INMC, Seoul National Univ.

요 약

딥 러닝 모델의 활성화 함수는 신경망의 깊이를 활용하기 위해 필수적인 역할을 하며, 비선형성을 통해 신경망의 각 층이 복잡한 특징을 학습할 수 있도록 한다. 하지만 많은 활성화 함수들은 지수 함수와 같은 비선형 연산을 필요로 하며, 이는 메모리 사용량과 연산 속도 측면에서 단점을 갖는다. 본 연구에서는 딥 러닝 모델의 메모리 사용량을 줄이고 연산 속도를 향상시키기 위해 활성화 함수 내부에서 piecewise linear 연산을 지수 연산의 대체제로 사용할 것을 제안한다. 이를 통해 자원이 제약된 환경에서 모델에 최적화 된 활성화 함수를 찾을 수 있다.

Ⅰ. 서 론

딥 러닝 모델에서 활성화 함수는 비선형성을 통해 각 신경망 층이 복잡한 특징을 학습할 수 있도록 한다. 이를 위해 많은 활성화 함수는 비선형 연산을 필요로 하는데, 이는 연산 속도와 메모리 사용량 측면에서 단점을 갖는다. 그 중 하나인 지수 연산은 많은 부동소수점 연산(Floating Point Operations, FLOPs)을 필요로 하며 연산 속도 개선을 위해 미리 계산된 Look-Up Table(LUT)을 활용하는 경우가 많지만, 여전히 선형 연산에 비해서는 연산 속도가 느리며 LUT 의 크기가 커지면 필요 메모리도 커진다는 한계가 있다. 이는 특히 모바일 기기나 임베디드 시스템과 같이 자원에 제약이 있는 on-device 환경에서 장애물이 된다.

이러한 문제를 해결하기 위해 본 연구에서는 piecewise linear 연산을 지수 연산의 대체제로 사용할 것을 제안한다. 이를 통해 on-device 환경에서 LUT 의 필요성을 없애 메모리 부담을 줄이고 선형 연산을 이용해 전반적인 연산 속도를 올릴 수 있다. 성능을 확인하기 위해 화자 인식 모델인 ECAPA-TDNN[1]을 사용하여 실험을 진행했다. 해당 모델의 softmax layer 에서 지수 연산 대신 piecewise linear 연산을 사용하여 실험을 진행하였으며, 구간을 나누는 방식을 달리하며 최적화된 활성화 함수를 설계했다. 결과적으로 베이스라인 모델과 비교했을 때, EER(Equal Error Rate) 측면에서 성능 저하 없이 연산 속도를 향상시키고 비선형 연산인 지수 연산을 제거하여 on-device 환경에서의 작동에 유리한 화자 인식 모델을 구축하였다.

Ⅱ. 본론

1) ECAPA-TDNN

본 연구에서는 ECAPA-TDNN 모델을 기반으로 실험을 진행하였다. ECAPA-TDNN 모델은 화자의 특징 벡터를 추출하고 이를 이용하여 화자의 유사도를 계산할 수 있는 화자 인식 모델로, hidden space 의 채널 간 연관성에 중점을 두었으며 convolution layer 가 주를 이룬다. 화자 인식 모델은 입력된 신호에서 전체 시간대의 정보를 하나로 압축하여 화자의 특징 벡터를 추출하는 과정을 거치는데, 본 연구에서는 이 과정에 포함되는 softmax 레이어를 수정하여 실험을 진행했다.

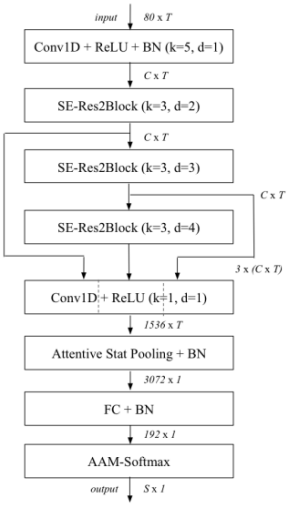

그림 1. ECAPA-TDNN 모델의 시스템 도식

2) Piecewise linear

Piecewise linear 함수는 구간별로 선형적인 형태를 갖는 함수로, 각 구간의 기울기를 다르게 설정할 수 있다. 본 연구에서는 지수 연산의 대체제로 piece-wise linear 연산을 사용해 실험을 진행하였으며, 학습의 안정성을 위해 구간의 경계에서 연속성을 갖도록 하였다. Piece-wise linear 의 특징을 갖는 기존의 활성화 함수로는 ReLU, Leaky ReLU 등이 있다.

본 연구에서 보이고자 하는 것은 pre-trained 모델에서 비선형적인 연산을 선형적인 연산으로 대체하였을 때 성능 저하 없이 연산 속도 향상과 필요 메모리 감소가 가능하다는 것으로, 기존의 piecewise linear 활성화 함수들을 사용해 실험하는 것과 차별성을 갖는다.

3) Look-Up Table

일부 연산들은 많은 FLOPs 를 필요로 하기 때문에 on-device 환경에서 효율성을 위해 사전에 계산된 Look-Up Table 을 활용하여 수행되는 경우가 많다. 이 방식은 입력될 수 있는 값들을 이산화 하여 해당 값들에 대한 연산 결과를 미리 계산해 테이블에 저장하고 이후에 테이블 조회로 연산을 대체하는 것이다. 이 방식은 기존의 연산을 수행하는 것보다는 빠르다는 장점이 있지만, 테이블을 조회하고 interpolation 을 통해 값을 구해야 한다는 점에서 추가적인 시간과 연산이 필요하며 테이블을 저장할 메모리가 필요하다는 단점이 있다.

본 연구에서는 piecewise linear approximation 을 통해 Look-Up Table 없이 기존의 연산을 선형 연산만 가지고 수행하고자 한다. 이 방식은 테이블이 존재하지 않아 그와 관련된 추가적인 과정이 존재하지 않으며 연산 속도, 메모리 측면에서 장점을 갖는다.

4) 실험 및 결과

실험은 총 세 가지 방식으로 진행했으며 각 방식은 softmax 레이어의 처리 방식에 따라 구분된다. Softmax 레이어는 입력으로 들어오는 벡터의 각 원소에 대해 지수 연산을 실행하고 정규화 하여 최종적으로 모든 원소들이 양수 값을 갖는 확률 분포로 변환한다. 이러한 특성을 반영하기 위해 각 방식들은 정규화 이전에 각 원소들이 양수가 되도록 하는 과정(clipping)을 거친다. 첫번째 방식은 입력 값들에 하한을 두어 모든 값들이 특정 값 이상이 되도록 한다. 두번째 방식은 입력 값들이 지수 함수에 근사한 1-구간 선형 함수를 통과한 후에 특정 값 이상이 되도록 한다. 세번째 방식은 입력 값들이 지수 함수에 근사한 2-구간 선형 함수를 통과한 후에 특정 값 이상이 되도록 한다. 이후 세 방식 모두 정규화를 진행한다.

측정은 두 가지 항목에 대해 진행했다. 첫번째 항목은 화자 인식 성능이며 베이스라인에 비해 성능 감소를 최소화하는 것을 목표로 한다. 측정 지표는 equal error rate(EER)으로, detection threshold 에 따른 false alarm rate 과 miss rate 이 같아지는 위치의 rate 을 나타내며 낮을수록 성능이 좋은 것을 의미한다. 세 가지의 실험 방식 모두에 대해 pre-trained 모델에 적용 후 fine-tuning 을 하지 않은 채로 성능을 측정하고, fine-tuning 을 진행한 후에 성능을 다시 측정하였다. 두번째 항목은 inference speed 로, 베이스라인에 비해 소요시간이 감소되는 것을 목표로 한다. 측정 지표는 relative elapsed time 으로 baseline 에 비해 소요되는 시간을 상대적으로 나타내며 낮을수록 소요 시간이 적은 것을 의미한다.

표 1. 화자 인식 성능 측정 결과

방식 구분	특징	EER(%) ↓ Before FT	EER(%) ↓ After FT
Baseline	Softmax	0.98	
방식 1	Clipping only	2.55	1.06
방식 2	1-구간 linear + clipping	2.09	**0.96**
방식 3	2-구간 linear + clipping	**2.02**	0.98

화자 인식 성능 측정 결과로 연산을 바꾸고 fine-tuning 을 진행하지 않은 상태에서는 세 가지 방식 모두 baseline 에 비해 성능이 크게 감소했으며 2-구간으로 가장 유사하게 근사한 방식 3 의 경우에 성능 감소가 가장 적었다. Fine-tuning 을 진행한 후에는 세 가지 방식 모두 baseline 과 거의 유사한 성능을 보였으며 방식 2 의 경우 오히려 성능이 향상되는 결과가 나왔다. 이를 통해 piece-wise linear approximation 을 통해 어느 정도 성능 유지가 가능하며, fine-tuning 을 통해 baseline 과 유사한 성능을 보일 수 있음을 확인했다.

표 2. Inference speed 측정 결과

방식 구분	특징	Relative elapsed time
Baseline	Softmax	1
방식 1	Clipping only	**0.9429**
방식 2	1-구간 linear + clipping	0.9643
방식 3	2-구간 linear + clipping	0.9714

Inference speed 측정 결과로 세 가지 방식 모두 소요 시간이 줄어들었으며 연산이 단순할수록 소요 시간이 많이 줄어드는 것을 확인할 수 있었다.

Ⅲ. 결론

본 연구를 통해 FLOPs 수가 많은 연산의 경우 piece-wise linear approximation 을 통해 on-device 환경에서 연산 속도를 향상시키고 필요 메모리를 줄일 수 있는 것을 확인했다. 추가적으로 piece-wise linear 함수의 계수를 학습 가능하도록 설정하여 모델에 최적화된 approximation 이 가능할 것으로 예상된다.

ACKNOWLEDGMENT

이 논문은 2024 년도 BK21 FOUR 정보기술 미래인재 교육연구단에 의해 지원되었음.

참 고 문 헌

[1] Desplanques, B., Thienpondt, J., Demuynck, K. (2020) ECAPA-TDNN: Emphasized Channel Attention, Propagation and Aggregation in TDNN Based Speaker Verification. Proc. Interspeech 2020, 3830-3834.