온디바이스 한국어 핵심어 검출을 위한 TinyML 모델 구현

최진우, 임재봉, 김태구, 조용훈, 이상화, 전동근, 김동현, 신기훈, 백윤주*

부산대학교

{jwchoi9965, jaebonglim, tbg8577, kchoyh95, lsanghwa, happyjdk, ehdgus1714, skh2929209}
@pusan.ac.kr, *yunju@pusan.ac.kr

Implementation of TinyML Model for On-Device Korean Keyword Spotting

Choi Jin Woo, Lim Jae Bong, Kim Tae Gu, Cho Yong Hun, Lee Sang Hwa,
Jeon Dong Keun, Kim Dong Hyun, Shin Ki Hun, Baek Yun Ju*

Pusan National Univ.

요 약

하드웨어의 발전과 IoT 시장 성장에 힘입어 센서에서 수집한 데이터로 온디바이스 인공지능 추론을 수행하는 Tiny Machine Learning(TinyML)에 대한 수요가 증가하고 있다. 핵심어 검출 분야에서는 연산 능력과 메모리 크기에 한계가 있는 Micro Controller Unit(MCU)에 TinyML 모델을 탑재하여 온디바이스 동작을 구현하는 연구가 미비한 실정이다. 또한, 영어 핵심어 검출에 대한 활발한 연구에 비해 한국어 핵심어 검출에 대한 연구는 더 필요한 상황이다. 본 논문에서는 18종의 한국어 핵심어를 검출하는 Depthwise Separable Convolution(DSC) 모델과 Residual block을 추가한 ResDSC 모델을 구현하고 성능을 평가한다. 온디바이스 동작 검증을 위해 각 모델은 양자화와 최적화 과정을 거쳐 ESP32 및 nRF9160 MCU에 탑재된다. 제안된 ResDSC 모델은 정확도 96%, 모델 크기 151KB, 온디바이스 추론 시간 0.12s를 달성하여 모든 평가 지표에서 DSC 모델보다 개선된 수치를 나타내었다.

Ⅰ. 서 론

인공지능 기술의 발달로 산업 현장과 헬스케어 등 다양한 환경에서 인공지능을 활용하는 응용 서비스와 디바이스가 개발되고 있다. 특히 매년 성장하는 IoT 시장에서 초소형, 저비용, 저전력의 강점을 가진 MCU를 인공지능과 접목하려는 시도가 증가하고 있다. 일반적으로 인공지능 모델은 많은 양의 입력 데이터와 높은 연산 능력을 요구하기 때문에 고성능의 디바이스를 사용한다. 따라서 대부분의 IoT 시스템에서는 말단 MCU에 부착된 센서로 수집한 데이터를 서버나 클라우드에 전송하여 인공지능 추론을 수행한다. 최근에는 하드웨어와 TinyML의 발전으로 인공지능 모델을 MCU에 탑재하여 온디바이스 추론 시스템을 구현하는 연구가 활발하다. 온디바이스 TinyML 추론 방식은 서버 및 클라우드에서 추론하는 방식보다 통신 지연시간 감소, 통신 방법 및 통신 음영지역 고려 불필요, 개인정보 보호 측면에서 이점을 가진다.

TinyML 적용 분야 중 핵심어 검출(Keyword spotting)은 제한된 수의 핵심어를 검출하는 음성 인식 방법으로 애플의 '헤이 시리(Hey Siri)', 구글의 '오케이 구글(OKay Google)'을 비롯해 각종 가전제품과 완구에 탑재되는 음성 명령 시스템에 활용된다. 주로 MCU를 활용하여 구현하는 시스템이므로 자원 제약적인 환경에서 인공지능 추론을 위한 TinyML 기술이 요구된다. 온디바이스 동작을 위해 CNN 모델의 연산량 감소 및 모델 경량화에 효과적인 DSC 모델 구조를 제안하는 연구[1]로 온디바이스 핵심어 검출 성능 향상에 기여하였으나, 한국어에 대한 성능 평가 연구가 필요하다. 대표적인 영어 핵심어 데이터 세트인 Google speech commands 데이터 세트에서 State Of The Art(SOTA)를 달성한 연구[2]는 한국어에 대한 성능 검증도 하였으나, 연산량 및 모델의 크기를 최적화하여 온디바이스 동작을 고려한 연구가 필요하다. 온디바이스 한국어 핵심어 검출 시스템을 구현한 연구[3]는 5종의 핵심어에 대해 90.91%의 정확도를 달성하는 DSC 모델을 구현하여 ESP32 MCU로 온디바이스 추론을 수행한다. 모델의 크기에 따른 성능 비교를 하지만 모델 및 MCU 종류, 핵심어의 개수를 늘리는 등 다양한 환경에서의 검증 연구가 필요하다.

본 논문에서는 온디바이스 한국어 핵심어 검출을 위한 2가지 TinyML 모델을 구현하고 성능을 평가한다. 18종의 한국어 핵심어를 검출하는 DSC모델과 Residual block 구조를 추가한 ResDSC 모델을 제안한다. 각 모델은 MCU에 탑재하기 위해 16bit 및 8bit 선형 양자화(Quantization)와 모델 파라미터 최적화를 수행한다. 온디바이스 동작은 ESP32와 nRF9160 2가지 MCU에서 검증한다.

Ⅱ. 본론

1. 한국어 데이터 세트 구성 및 전처리

한국어 핵심어 검출 TinyML 모델을 구현하기 위해 표 1에 나타낸 18종의 한국어 핵심어 데이터를 사용한다. 가전제품과 완구에 활용할 수 있는 핵심어를 선정하여 73명의 음성을 녹음하였다. 각 핵심어는 샘플링 주파수 16KHz, 녹음 길이 1s의 wav 파일로 저장하여 총 16,365개의 데이터를 수집하였다. 실생활에서도 안정적으로 성능을 유지하기 위해 ESC-50, FSD50k, AI hub 공개 데이터 세트에서 추출한 다양한 배경소음 데이터 22,913개와 한국어 대화 데이터 16,000개를 수집하였다.

표 1. 수집한 한국어 핵심어 18종

꺼줘	켜줘	틀어줘	조명	안녕	잘자
배고파	사랑해	살려줘	심심해	무서워	노래
동화	새우야	아이유	방탄	호출명령1	호출명령2

데이터 세트는 검출 대상인 18종의 한국어 핵심어와 검출 대상이 아닌 음성 분류를 위한 1종의 한국어 대화로 구성된다. 19종의 데이터는 볼륨 조절, 음높이 조절, 소리 늘리고 줄이기(Audio time stretching) 증강 기법을 적용하여 데이터의 다양성을 확보한다. 증강기법이 적용된 데이터는 랜덤한 확률로 배경소음을 합성하여 배경소음이 있는 환경과 없는 환경

모두에서 검출 성능의 강건성을 갖도록 한다.

음성 데이터는 음성의 시계열 정보를 보존하며 주파수 특성을 검출하기 위해 Short time fourier transform을 적용하여 스펙트로그램을 생성한다. 스펙트로그램은 사람의 달팽이관 특성을 고려한 데이터 멜 스케일링을 위해 멜 필터 뱅크를 활용하여 멜 스펙트로그램으로 변환한다. 멜 스펙트로그램은 압축 기법인 Discrete Cosine Transform(DCT)을 적용하여 Mel Frequency Cepstral Coefficient(MFCC)를 생성하여 TinyML 모델의 입력으로 사용한다. 멜 스펙트로그램의 DCT 압축으로 연산량 및 메모리 크기를 감소시켜 TinyML 환경에 더 적합한 입력 데이터를 생성한다.

2. TinyML 모델 구현

인공지능 모델에 Residual block 구조를 추가하면 성능 향상에는 효과적이나 연산량 및 모델 크기가 증가하여 TinyML 모델에는 적합하지 않다. 해당 문제를 모델 경량화에 이점이 있는 DSC 모델에 적용하여 모델의 성능을 향상시키며 TinyML에 적합하도록 설계한 ResDSC 모델을 제안한다. 그림 1과 표 2는 DSC 모델 및 Residual block을 추가한 ResDSC 모델 구조와 하이퍼파라미터를 나타낸다. Convolution 연산마다 배치 정규화(Batch normalization)와 ReLU 연산을 수행한다. 모델의 출력은 Global Average Pooling(GAP)과 Softmax 연산의 결과로 얻어진다. 과적합 방지를 위한 Dropout을 적용하고 모든 계층은 바이어스를 사용하지 않는다.

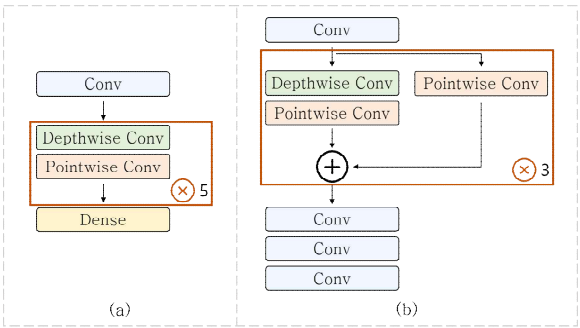

그림 1. (a)DSC 및 (b)ResDSC 모델 구조

표 2. (좌)DSC 및 (우)ResDSC 모델 하이퍼파라미터

Block	Output channel	Kernel	Stride	Block	Output channel	Kernel	Stride
Conv1	32	3	2	Conv1	32	3	2
DSC1	32	3	1	ResDSC1	40	3	2
DSC2	64	3	2	ResDSC2	40	3	1
DSC3	64	3	1	ResDSC3	40	3	1
DSC4	128	3	2	Conv2	32	3	1
DSC5	128	3	1	Conv3	128	1	1
GAP	-	-	-	Conv4	19	1	1
Dense	19	-	-	GAP	-	-	-
Softmax	-	-	-	Softmax	-	-	-

3. 온디바이스 동작 구현

구현된 TinyML 모델은 표 3에 나타낸 서로 다른 제원을 가지는 MCU에 탑재한다. 온디바이스 동작을 구현하기 위해 모델의 연산량과 크기를 저사양 MCU에 최적화하는 모델 경량화 기법을 적용한다. nRF9160의 경우 EPS32에 비해 크게 낮은 성능과 메모리 용량을 가지고 있어 모델의 하이퍼파라미터를 조절하고 경량화 기법을 적용한다. 모델의 Dropout 계층은 제거하고 배치 정규화 계층은 파라미터를 그 이전 계층의 바이어스로 추가한 후 제거한다. 32bit 실수 파라미터로 구성된 모델을 선형 양자화 과정을 거쳐 ESP32는 16bit, nRF9160은 8bit 정수 파라미터로 변환한다. 변환된 모델은 MCU에 C 헤더파일 형식으로 탑재되고 모델의 각 계층은 CMSIS-NN 라이브러리를 활용하여 구현한다.

표 3. TinyML 모델을 탑재한 MCU 제원

	ESP32	nRF9160
Memory	4MB Flash 8MB RAM	1MB Flash 256KB RAM
Processor	Xtensa Dual-core 240MHz 32bit	Arm Cortex-M33 64MHz 32bit

Ⅲ. 실험 및 성능평가

모델 평가 지표는 검출 정확도, 모델의 파라미터 개수 및 용량, 온디바이스 추론 시간으로 선정하였다. 자원 제약적인 MCU 환경에서의 온디바이스 동작을 위해 모델의 크기와 실제 추론 시간은 중요한 성능 지표이다. 표 4는 지정된 평가 지표에 따른 TinyML 모델 성능 실험 결과를 나타낸다. ESP32에 탑재되는 모델의 입력은 (49, 40) 형태로 설정하고, nRF9160에 탑재되는 모델의 입력은 (49, 10) 형태로 설정하였다. 실험은 ResDSC 모델이 모든 평가 지표에서 성능이 향상되는 결과를 나타내었다. 이는 제안하는 ResDSC 모델 구조가 추론 연산량 및 모델의 크기 감소에 효과적이고 동시에 검출 정확도 개선에도 효과가 있음을 의미한다. 최고 정확도는 96%의 높은 성능을 달성하였고 모델의 파라미터와 용량도 저사양 MCU에 적합한 크기로 경량화하였다. 온디바이스 추론 시간은 0.12s를 달성하여 실시간 응용 서비스에 대한 활용 가능성을 보여준다.

표 4. MCU 종류별 DSC 및 ResDSC 모델 실험 결과

	Model	Accuracy (%)	Number of parameter	Memory (KB)	Inference time (s)
ESP32	DSC	94	38,131	198	0.16
	ResDSC	**96**	**28,848**	**151**	**0.12**
nRF9160	DSC	93.29	21,075	63.1	0.23
	ResDSC	**94.14**	**19,283**	**58.2**	**0.22**

Ⅳ. 결론

본 논문은 온디바이스 한국어 핵심어 검출을 위한 2가지 TinyML 모델을 구현한다. 경량화가 강점인 DSC 모델을 구현하고 DSC 모델과 성능 개선에 효과적인 Residual block을 결합한 ResDSC 모델을 구현하여 성능을 비교하였다. 4가지 평가 지표에서 ResDSC 모델이 DSC 모델 성능을 개선하였고 제안하는 ResDSC 모델의 온디바이스 TinyML 적합성을 확인하였다. 18종의 한국어 핵심어에 대해 검출 정확도 96%를 달성하였으며 ESP32, nRF9160 2가지 MCU로 온디바이스 동작을 검증하였다.

ACKNOWLEDGMENT

본 논문은 교육부와 한국연구재단의 재원으로 지원을 받아 수행된 3단계 산학연협력 선도대학 육성사업(LINC 3.0)의 연구결과입니다.

참 고 문 헌

[1] Y. Zhang, N. Suda, L. Lai and V. Chandra, "Hello edge: Keyword spotting on microcontrollers," arXiv preprint arXiv:1711.07128, 2017.

[2] D. Seo, H. Oh and Y. Jung, "Wav2KWS: Transfer learning from speech representations for keyword spotting," IEEE Access, vol. 9, pp. 80682-80691, May. 2021.

[3] J. Lee, J. Lim, H. Park and Y. Baek, "Design and implementation of the korean keyword spotting system in embedded device," The Korea Institute of Information and Communication Engineering, vol. 24, no. 2, pp. 45-47, Oct. 2020.