WebRTC 기반 화상회의 시스템의 가상 배경에 딥 러닝 적용에 대한 성능 분석

유상우, 고경찬, 홍원기
포항공과대학교

{rswoo, kkc90, jwkhong}@postech.ac.kr

Performance Analysis of Applying Deep Learning in Virtual Background of WebRTC-based Video Conferencing System

Sangwoo Ryu[1], Kyungchan Ko[2], James Won-Ki Hong[2]
Graduate School of Artificial Intelligence, POSTECH[1]
Department of Computer Science and Engineering, POSTECH[2]

요 약

최근 다양한 서비스에서 인공지능이 활용되고 있고, 영상회의 시스템에서 또한 더 나은 성능을 위해 화질 개선 및 가상 배경 등과 같은 기능에 인공지능을 적용하고 있다. 하지만 웹 기반 영상회의에서는 웹 브라우저라는 제한적인 환경으로 인해 이런 기능들의 적용이 제한된다. 본 논문에서는 웹 기반 서비스에서 딥 러닝 모델을 사용하는 기능을 제공하기 위해 웹 환경에 딥 러닝을 적용하는 여러 방식을 소개하고, 실제 영상회의 서비스에 여러 딥 러닝 모델을 적용해 성능을 평가한다. 마지막으로 웹 기반 영상회의에 딥 러닝 모델을 적용하기 위해 고려해야 하는 부분을 논의한다.

Ⅰ. 서 론

많은 서비스들이 인공지능을 이용한 기능을 제공하고, 코로나-19 로 인해 사용량이 늘어난 영상회의 서비스에서도 이를 이용한 부가 기능들을 추가하고 있다. 예를 들어 화질을 개선하기 위한 초해상화(Super Resolution), 사용자의 배경을 바꾸는 가상 배경(Virtual Background) 등에서 딥 러닝[1]을 이용해 좋은 사용자 경험을 제공할 수 있다. 영상회의에 이런 딥 러닝을 이용한 기능을 사용자들에게 제공하기 위해, 영상회의 서비스 제공자는 서비스에 딥 러닝 모델을 적용할 방법을 선택을 해야 한다. 웹 브라우저에서 서비스를 제공하는 영상회의의 경우, 사용자의 컴퓨터에 설치를 할 필요가 없는 대신 웹 브라우저라는 제한된 환경에서 기능을 제공해야 한다. 본 논문에서는 영상회의의 가상 배경 기능에 초점을 맞추어, 이 기능을 딥 러닝을 이용해 수행하기 위해 웹 브라우저 기반 영상회의에 딥 러닝을 적용하기 위한 방법을 제공하고 성능 분석을 수행한다. 또한 웹 브라우저 기반 영상회의에 딥 러닝을 적용하기 위해 고려해야 하는 부분을 정리하고 사용자 경험 개선을 위해 필요한 향후 연구들을 제시한다.

Ⅱ. 배경

웹 환경에 딥 러닝을 적용하는 방법은 모델을 적용하는 위치에 따라 나눌 수 있으며, 각각의 장단점이 존재한다.

먼저 딥 러닝 모델이 서버에 위치하는 경우 서버의 자원(CPU, GPU)과 코드를 이용해 딥 러닝을 수행한다. 따라서 웹 브라우저 및 사용자의 기기와 독립적으로 동작하기 때문에 모든 사용자에게 유사한 사용자 경험을 제공할 수 있다. 하지만 클라이언트와 입출력 데이터를 주고받아야 하기 때문에 추가적인 대역폭이 필요할 수 있고, 서버에서의 연산으로 실시간 처리에 지연이 일어날 수 있다.

딥 러닝 모델이 클라이언트에 위치하는 경우 지연이 적고 서버에 부담이 줄어들지만, 클라이언트 기기의 자원을 이용해 딥 러닝이 수행되어야 해서, 기기의 성능에 따라 서로 다른 사용자 경험을 받을 수 있다.

딥 러닝 모델을 클라이언트 측(웹 브라우저)에 적용하는 대표적인 방법으로는, 자바스크립트 (JavaScript) 라이브러리를 사용하는 방법과 웹 어셈블리를 사용한 방법이 있다.

대표적인 자바스크립트 라이브러리로는 TensorFlow.js [2], Keras.js, ConvNetJS 등이 존재하며, 특히 TensorFlow.js 에서는 이미지 분류, 객체 감지 등 일반적인 사용 사례들에 대해서 선행 학습된 모델을 제공한다.

웹 어셈블리[3]는 웹에서 C/C++ 등으로 작성된 코드를 실행할 수 있도록 하는 기술로, C/C++ 라이브러리를 빌드하거나, 딥 러닝 모델 추론 과정 전체를 빌드해 사용할 수 있다. 웹 어셈블리를 사용하는 경우 SIMD (Single Instruction, Multiple Data), 멀티 스레드(Multi Threads)와 같이 딥 러닝을 가속화 할 수 있는 옵션을 선택할 수 있다는 장점이 있다.

Ⅲ. 성능 분석

이 장에서는 딥 러닝 모델을 WebRTC[9] 기반 영상회의 서비스인 Vmeeting[4] 클라이언트에 적용하였을 때 여러 적용 방식 별 성능을 보여준다. 성능 평가는 2.90GHz i5-9400F CPU, 16.0GB RAM 데스크탑 PC 의 Chrome 브라우저에서 이루어졌다. 딥 러닝 모델의 속도를 측정하기 위해 100 초 동안의 FPS 및 추론 시간의 평균을 이용하였다. 라이브 스트리밍 지원 프로그램인 OBS Studio[5]에서 제공하는 가상 카메라를 이용하였으며, JVT(Joint Video Team) 테스트 영상들 중 영상회의와 유사하게 실내 환경에서 상체만 나오는 10 초 길이의 "Akiyo" 영상을 반복 재생하였다. 모델의 속도 뿐만 아니라, 실제로 가상 배경 기능을 이용할 때 사용자의 화면에 보여지는 품질도 중요하다. 결과의 품질은 한 모델에서 인자들이나 모델의 종류에 따라 달라질 수 있기 때문에 추가적으로 각 모델의 추론 정확도를 측정하였고, 그 방식으로 IoU(Intersection over Union)를 이용하였다. 영상 시퀀스의 다섯 프레임(1 초, 3 초, 5 초, 7 초, 9 초)의 IoU 의 평균을 이용하였다.

Multiplier	quantBytes=2				quantBytes=1			
	크기	FPS	추론 시간 (ms)	IoU	크기	FPS	추론 시간 (ms)	IoU
1	~6MB	8.15	80.47	0.9510	~3MB	8.95	79.01	0.3012
0.75	~2MB	9.01	69.45	0.9518	~1MB	8.54	70.63	0.8760
0.5	~1MB	9.59	61.4	0.9074	~0.6MB	10.52	57.53	0.7961

표 1. TensorFlow.js MobileNet-V1 WebGL 백엔드 성능

표 1 은 TensorFlow.js 에서 사람-배경 분리를 위해 기본으로 제공하는 모델인 MobileNet-V1[6]을 이용해 영상회의에서 배경 분리를 수행한 결과이다. 컨볼루션(Convolution) 연산에서 채널 수 (Multiplier), 양자화 정도(quantByte)를 조절해 모델의 크기와 연산의 수를 조정할 수 있었다. 크기가 큰 모델일수록 속도는 느리지만, 높은 정확도를 가지는 경향을 확인할 수 있다.

quantBytes=2				quantBytes=1			
크기	FPS	추론 시간 (ms)	IoU	크기	FPS	추론 시간 (ms)	IoU
~45MB	3.96	202.36	0.9648	~22MB	3.98	205.27	0.9649

표 2. TensorFlow.js ResNet50 WebGL 백엔드 성능

표 2 는 역시 TensorFlow.js 에서 기본으로 제공할 수 있는 모델인 ResNet50[7]을 이용해 실험을 수행한 결과이다. 이 모델에서는 quantByte 의 변화에 따라 결과에 큰 변화를 확인할 수 없었다.

모델 입력 크기	SIMD 미사용				SIMD 사용			
	크기	FPS	추론 시간 (ms)	IoU	크기	FPS	추론 시간 (ms)	IoU
256x144	400KB	14.59	19.08	0.9786	400KB	18.1	6.96	0.9786
190x96	400KB	17.52	9.39	0.9761	400KB	19.07	3.4	0.9761

표 3. 웹 어셈블리 - Google Meet Segmentation Model 성능

SIMD 미사용				SIMD 사용			
크기	FPS	추론 시간 (ms)	IoU	크기	FPS	추론 시간 (ms)	IoU
2.7MB	3.12	256.9	0.9148	2.7MB	8.34	66.69	0.9148

표 4. 웹 어셈블리 – DeepLab-V3 모델 성능

표 3 은 Google Meet 에서 Apache 2.0 라이센스로 제공했던 MobileNetV3-small 개선 모델을 이용해 실험을 수행한 결과이고, 표 4 는 TensorFlow Lite 에서 제공하는 DeepLab-V3[8]을 이용한 결과이다.

Google Meet Segmentation 모델은 양자화 및 최적화가 수행된 모델이기 때문에 상대적으로 크기가 큰 DeepLab-V3 보다 좋은 성능을 보여주었다. 두 모델 모두 SIMD 를 사용했을 때 같은 정확도에 더 좋은 성능을 보여주었다. 현재 Vmeeting 에는 Google Meet Segmentation 모델이 적용되어 있다.

Ⅳ. 결론

본 논문에서는 웹 브라우저 환경에 딥 러닝 적용을 위해, 영상회의 서버 및 클라이언트에서 딥 러닝 모델 적용 방식을 설명하였다. 특히 클라이언트에 적용하는 방식의 경우 실제로 웹 기반 영상회의인 Vmeeting 에 여러 딥 러닝 모델을 적용하여 성능을 분석하였다.

웹 기반 영상회의에 딥 러닝을 적용하기 위해서는 다양한 요소들이 고려되어야 한다. 먼저 딥 러닝 모델의 자원 및 대역폭 사용량을 파악하여 서버, 클라이언트 중 적절한 모델 적용 위치를 선택해야 한다. 딥 러닝 모델 적용 방식과 모델의 크기에 따라 속도 및 정확도가 달라지기 때문에, 사용하려는 모델에 맞는 적용 방식을 선택해야 한다. 영상회의는 실시간 처리가 중요하기 때문에 성능과 최적화 두 부분을 모두 고려할 필요가 있다.

최근에는 모바일 기기로 영상회의를 참여하는 사용자들이 늘어나고 있는데, 클라이언트에 딥 러닝 모델을 적용하는 경우 클라이언트 기기의 성능에 영상회의의 품질이 좌우되기 때문에 저성능 디바이스에서 사용할 수 있는 딥 러닝 모델이나 이런 환경에 적응할 수 있는 모델의 지속적인 연구가 필요하다.

ACKNOWLEDGMENT

이 논문은 2021 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (2018-0-00749, 인공지능 기반 가상 네트워크 관리기술 개발, 2017-0-01633, 대학 ICT 연구센터육성지원사업).

참 고 문 헌

[1] LeCun, Y., Bengio, Y. & Hinton, G. "Deep learning," Nature 521, pp. 436- 444, 2015, (https://doi.org/10.1038/nature14539).

[2] Martín Abadi et al., "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, (https://tensorflow.org).

[3] Haas, A., et al. "Bringing the Web up to Speed with WebAssembly," SIGPLAN Not., 52(6), pp. 185- 200, 2017.

[4] Vmeeting, https://vmeeting.io

[5] OBS studio, https://obsproject.com/

[6] Howard, A. et al. "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," arXiv e-prints, arXiv:1704.04861, 2017.

[7] He, Kaiming et al. "Deep Residual Learning for Image Recognition," pp. 770-778. 10.1109/CVPR.2016.90, 2016.

[8] Chen, L.C. et al. "Rethinking Atrous Convolution for Semantic Image Segmentation," arXiv e-prints, arXiv:1706.05587, 2017.

[9] A. Bergkvist et al. "WebRTC 1.0: Real-time communication between browsers," W3C Working Draft, Feb. 2015.