일반논문 (Regular Paper)
방송공학회논문지 제29권 제3호, 2024년 5월 (JBE Vol.29, No.3, May 2024)
https://doi.org/10.5909/JBE.2024.29.3.330
ISSN 2287-9137 (Online) ISSN 1226-7953 (Print)

E2E 딥러닝 비디오 코덱의 VCM 내부 코덱 적용을 통한 성능 향상에 관한 연구

전 상 균[a], 이 동 민[a], 추 현 곤[b], 서 정 일[a)‡]

Improving VCM Performance by applying E2E Deep Learning based Video Codecs

SangKyun Jeon[a], Dongmin Lee[a], Hyongon Choo[b], and Jeongil Seo[a)‡]

요 약

컴퓨터 비전 인공지능 엔진이 임무를 수행함에 있어서 임무 수행 능력은 유지하면서 처리하는 영상 정보의 양을 최대한 줄이는 것을 목표로 하는 기계를 위한 비디오 부호화 기술(VCM)에 대한 국제 표준화가 진행되고 있다. VCM은 백본 네트워크 잠재공간의 Feature를 대상으로 하는 FCM과 이미지나 영상 신호를 대상으로 하는 VCM으로 구분하여 표준화가 진행되고 있으나, 부호화기의 최종 단계에서 영상 데이터를 압축하고 비트스트림을 생성하는 단계에서는 기존의 영상 부호화 표준인 VVC를 내부 코덱으로 사용하고 있다. 이는 FCM과 VCM의 성능 개선을 제한하는 결과를 초래하고 있으며, 딥러닝 네트워크와 E2E 딥러닝 영상 부호화 네트워크를 동시에 훈련하여 성능을 최적화할 가능성도 배제하게 된다. 본 논문에서는 이러한 문제를 해결하고자 FCM과 VCM의 내부 코덱을 딥러닝 기반 코덱으로 변경함으로써 FCM과 VCM의 성능 개선 가능성을 확인하였다.

Abstract

Efforts for international standardization of Video Coding for Machine(VCM) aim to reduce video data processed by computer vision AI while maintaining performance. VCM is divided into two tracks: FCM focuses on the backbone networks' latent space features, and VCM on image and video signals. Currently, FCM and VCM uses VVC standard codec for compressing video data and generating bitstream at internal codec. This limits performance imporvements and rulls out the simultaneous training deep learning networks for optimization. This paper suggests replacing FCM and VCM's internal codec with a deep learing based codec, demonstrating potential enhancements in FCM and VCM's performance.

Keyword : Video Coding for Machine, Image & Video Compression, Feature Coding, Computer Vision

a) 동아대학교 컴퓨터AI공학부(Department of Computer Engineering & AI, Dong-A University)
b) 한국전자통신연구원(ETRI)
‡ Corresponding Author : 서정일(Jeongil Seo)
E-mail: jeongilseo@dau.ac.kr
Tel: +82-51-200-7796
ORCID: https://orcid.org/0000-0001-5131-0939

※본 연구는 과학기술정보통신부의 재원으로 한국연구재단(No.RS-2023-00273349)과 정보통신기획평가원(No.2023-0-00076, SW중심대학(동아대학교), No.2020-0-00011, 기계를 위한 영상 부호화 기술)의 지원을 받아 수행된 결과임.
· Manuscript April 7, 2024; Revised May 20, 2024; Accepted May 22, 2024.

Ⅰ. 서 론

컴퓨터 비전 기반 인공지능 엔진이 임무를 수행함에 있어서 임무 수행 능력을 유지하면서 처리하는 영상 정보의 양을 줄이는 것은 매우 중요하다. 이를 위해 MPEG에서는 기계를 위한 영상 부호화(Video Coding for Machine, VCM)[1]에 대한 표준화를 진행 중이다. MPEG은 VCM의 표준화 범위를 Track 1인 FCM과 Track 2인 VCM으로 구분했다. FCM은 영상 특징 벡터 부호화로, 입력 영상을 백본 네트워크에 통과시킨 잠재공간(Latent space) 데이터를 입력으로 받아 특징 맵과 데이터로 부호화한다. VCM은 이미지와 비디오를 직접 압축하는 이미지 및 비디오 부호화이다. 해당 트랙은 이미지나 비디오를 입력받아 인공지능 엔진이 필요로 하는 부분을 선택하거나 변형하여 압축한다.

FCM에서는 컴퓨터 비전 기반 인공지능 엔진을 통과해 만들어진 각각의 특징 맵을 딥러닝 네트워크를 통해 하나로 합치고, 내부 코덱(Inner Codec)을 이용해 부호화한다. 그런 다음 다시 내부 코덱을 통해 복원된 특징 맵을 원래 컴퓨터 비전 기반 인공지능 엔진이 출력하는 특징 맵의 개수에 맞춰 복원한다. VCM에서는 이미지나 비디오로부터 ROI(Region Of Interest)[4]를 특정하고, 해당 부분을 추가적으로 부호화한다. 그 다음, 내부 코덱을 통해 부, 복호화한다. 이때, 두 트랙이 공통적으로 공유하는 부분은 내부 코덱이다. 내부 코덱은 VVC(Versatile Video Coding)[2]의 테스트 모델인 VTM(VVC Test Model)을 사용한다. 이때, VCM에서는 옵션으로 LIC(Learned Image Compression)[3]를 그룹별 첫 번째 프레임에 사용할 수 있다.

FCM과 VCM 모두 내부 코덱으로 VTM를 사용하면서 딥러닝 기반의 처리와 기존 방식의 코덱이 혼합된 형태를 가진다. 이로 인해 몇가지 문제가 발생하게 된다. 먼저 FCM의 경우에는 학습 시 딥러닝 기반 내부 코덱을 이용하여 학습하고, 실제 Inference 단계에서는 VTM을 이용한다. 이를 위해 전, 후처리를 담당하는 네트워크의 파라미터를 VTM-12.0에 맞게 조정하는 파인 튜닝 과정을 거친다[19]. 이 때문에 내부 코덱을 변경하는 경우에 성능 저하가 발생하며 내부 코덱 전, 후의 네트워크를 재조정해야 하는 문제가 발생한다. 다음으로 VCM에서는 VTM과 함께 딥러닝 기반의 LIC를 사용할 수 있으나 I-Frame에만 적용이 가능하다. 이 경우에는 부, 복호화 과정을 거치면서 각각의 코덱(VCM, VTM, LIC)들이 너무 많은 임시 파일들을 생성하게 되고 복잡도가 증가하는 문제가 보고되고 있다. 마지막으로 LIC에서 부, 복호화한 영상 정보가 VTM에 제대로 전달되지 않는 문제가 발생한다. 이로 인해 영상의 복원률이 떨어지는 문제를 가지고 있으며, 컴퓨터 비전 인공지능의 성능까지 저하되는 현상이 벌어진다.

본 논문에서는 FCM과 VCM의 내부 코덱을 End-To-End(E2E) 딥러닝 기반 비디오 코덱을 적용하여 기존의 방식과 비교하고자 한다. 먼저, LIC와 VTM이 결합된 기존 방식의 내부 코덱을 End-To-End 딥러닝 기반 비디오 코덱으로 변경한다. 그 다음 변경된 FCM과 VCM 코덱을 이용해 영상들을 부, 복호화하고 Object Detection과 Object Tracking 딥러닝 네트워크에 입력하여 임무 수행 결과를 비교한다. 이를 통해 딥러닝 네트워크가 전체에 적용된 경우와 아닌 경우의 코덱 성능을 비교한다.

본 논문의 구성은 다음과 같다. 2장의 1절에서 End-To-End 딥러닝 기반의 비디오 코덱의 개요와 본 논문에서 적용한 DCVC-DC(Deep Contextual Video Compression with Diverse Context)[5]를 소개한다. 2절에서는 기존의 VCM, FCM 표준 소프트웨어의 내부 코덱에 DCVC-DC를 적용한 설계를 설명한다. 3장에서는 기존의 표준 소프트웨어와 비교한 실험을 설명한다. 3장의 1절에서 실험 방법에 대해 설명하고, 2절에서 실험 결과를 분석한다. 마지막으로 4장에서 결론을 맺는다.

Ⅱ. VCM Architecture의 구성과 End-To-End 딥러닝 기반 비디오 코덱의 적용

본 장의 1절에서는 End-To-End 딥러닝 기반 비디오 코덱에 대해 소개하면서, 본 논문에서 적용한 비디오 코덱의 설계와 이점에 대해 소개한다. 2절에서는 기존의 VCM, FCM 설계와 본 논문에서 변경한 설계를 설명하면서 기존 방식의 문제점과 End-To-End 딥러닝 기반 비디오 코덱의 적용 방법에 대해 논의한다.

1. End-To-End 딥러닝 기반 비디오 코덱

딥러닝을 기반으로 이미지나 비디오를 압축하고 복원하는 기술인 Neural Video Compression은 2017년 ICLR을 통해 처음 제안되었다. 제안된 E2E Deep Image Compression(DIC)[7]는 변환과 양자화 부분을 이진 코사인 변환에서 딥러닝 네트워크로 대체했다. 이후 2019년에는 DIC를 동영상으로 확장한 E2E Deep Video Compression Framework (DVC)[8]가 제안되었다. 이 코덱은 Optical Flow 예측[9]과 Motion Vector 예측을 딥러닝 네트워크를 통해 진행하도록 한다. 훈련 과정에서 DIC를 통해 예측된 비트 스트림과 실제 비트 스트림 간의 에러, 그리고 복원한 프레임과 원본 영상 프레임 간의 에러가 줄어드는 방향으로 학습시킨다. DVC는 기존의 비디오 코덱이 블록으로 나눠 부호화했던 것과 달리, 각 화소별로 부, 복호화를 진행한다. 이런 방식을 통해 기존 코덱보다 공간 복잡도나 부, 복호화에 걸리는 시간을 단축시켰지만, 압축률이 기존 코덱보다 낮은 문제가 발생했다.

DCVC-DC는 딥러닝 기반 코덱의 장점인 속도나 낮은 복잡도를 유지하면서, 압축률을 높이는 데 집중했다. DCVC-DC는 기존의 DVC 구조에서 벗어나 각 프레임이 가진 특징 정보를 다음 프레임으로 전달해 학습하여 프레임을 생성한다. DCVC-DC는 세 가지의 주요 특징을 가지고 있다. 먼저 계층적 품질 패턴이다. 기존 코덱은 주로 이전 한, 두 개의 프레임에 대한 정보를 가지고 부, 복호화를 진행한다. 반면 해당 모델에서는 다음 프레임에 대한 상관관계를 매 프레임마다 학습하면서 더 넓은 시각적 상관관계를 활용할 수 있다. 두 번째로 다중 오프셋 방식을 사용한다. 기존 코덱들이 사용하는 가중치 예측 방식을 활용하여 변조 마스크를 씌워 오프셋을 만든다. 이 오프셋들을 모션을 기준으로 정렬하면서 복잡하거나 큰 모션 벡터에 대한 워핑 오류를 줄인다. 마지막으로 쿼드 트리 분할을 기반으로 한 엔트로피 코딩 기법을 사용한다. 이 기법은 단일 이미지의 모든 공간을 동시에 처리한다. 채널 차원을 통해 4개의 그룹으로 분할하고 각 그룹들을 다시 4개로 나누어 각각을 부, 복호화한다. 이를 통해 기존의 방식[10,11]보다 더 많은 특징을 사용하면서도 더 빠른 속도로 진행한다. 그림 1에서 볼 수 있듯이, 이 세 가지의 처리를 프레임마다 진행하고 복원 프레임을 생성한다. 이 복원 프레임을 원본 프레임과 비교하여 에러를 줄이도록 한다.

그림 1은 DCVC-FM(Feature Modulation)[6]의 전체 프레임워크를 나타내는 그림이다. DCVC-FM은 DCVC-DC를 기반으로 복원된 프레임의 화질을 결정하는 q_t를 추가하였다. 그림 1에서 볼 수 있듯이 현재 이미지가 입력되었을 때,

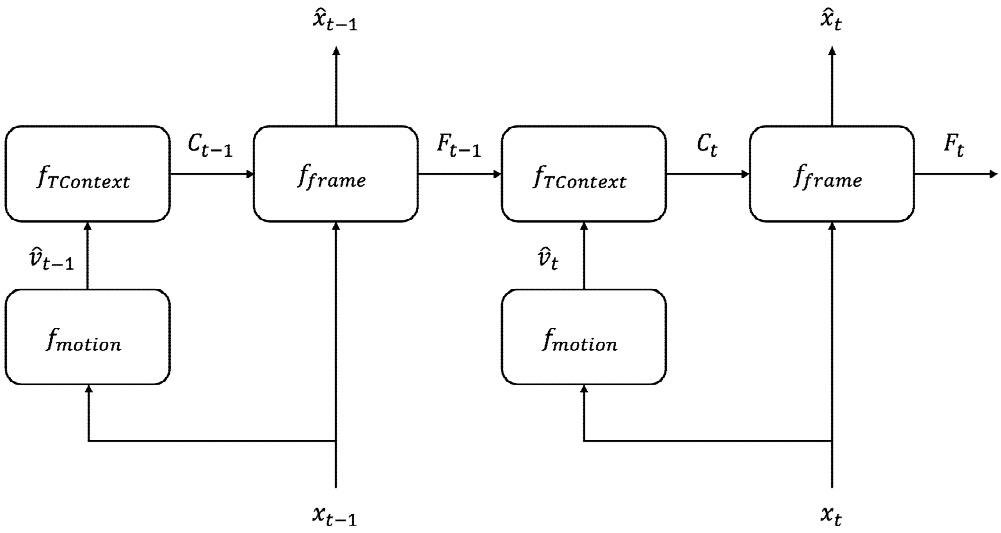


그림 1. DCVC-FM의 전체 프레임워크
Fig. 1. The framework of DCVC-FM built on DCVC-DC

이전 프레임으로부터 전달받은 F_{t-1}을 $f_{TContext}$를 통해 복원한다. 이때, f_{motion}의 optical flow 네트워크를 통해 생성된 모션 벡터를 이용하여 이전 프레임과 현재 프레임의 특징인 C_t를 생성한다. 그다음으로 f_{frame}이 해당 특징을 이용해 프레임을 부, 복호화하고 원본과 비교하여 파라미터를 조정한다. 해당 과정을 모든 프레임에서 반복하면서 영상 전체의 특징을 유지하게 된다. 또, 프레임마다 양자화 계수(Quantization Parameter, QP)를 새로 계산하면서 영상의 화질을 유지하려는 특징을 가지고 있다.

2. End-To-End 딥러닝 기반 코덱을 적용한 VCM과 FCM

그림 2의 (a)는 기존의 VCM 설계이다. 그림 2의 (a)에서 볼 수 있듯이 VCM은 크게 세 가지 단계로 나누어져 있다. 먼저, 전처리 단계에서 Temporal Resample, Spatial Resample, ROI Generator를 거쳐 ROI를 생성한다. 그 다음 내부 코덱인 LIC와 VVC를 이용하여 각 프레임을 부, 복호화한다. 마지막으로 후처리 단계에서 ROI Generator, Spatial Resample, Temporal Resample 순서로 최종 복원한다. 이때, 내부 코덱에서 LIC는 GOP 별 첫 번째 프레임인 I-Frame에 대해 작동하고 생성된 영상 정보를 VVC에 전달한다. 그러면 나머지 P-Frame들에 대해 VVC가 복원을 진행한다. 이 과정에서 여러 문제가 발생하는데, 첫 번째 문제는 LIC가 생성한 정보가 VVC에 원활하게 전달하지 못한다는 것이다. 각 GOP 별 첫 번째 프레임에 대한 정보가 일부 유실된 채로 복원을 진행하면서 복원률이 떨어지는 문제가 발생한다. 두 번째는 LIC와 VVC가 영상을 부, 복호화하는 과정에서 너무 많은 양의 파일을 생성한다는 것이다. 이것은 곧 공간 복잡도 문제와 이어져 성능을 저하시킬 수 있다.

그림 2의 (b)는 기존 FCM의 설계이다. FCM은 MSFF,

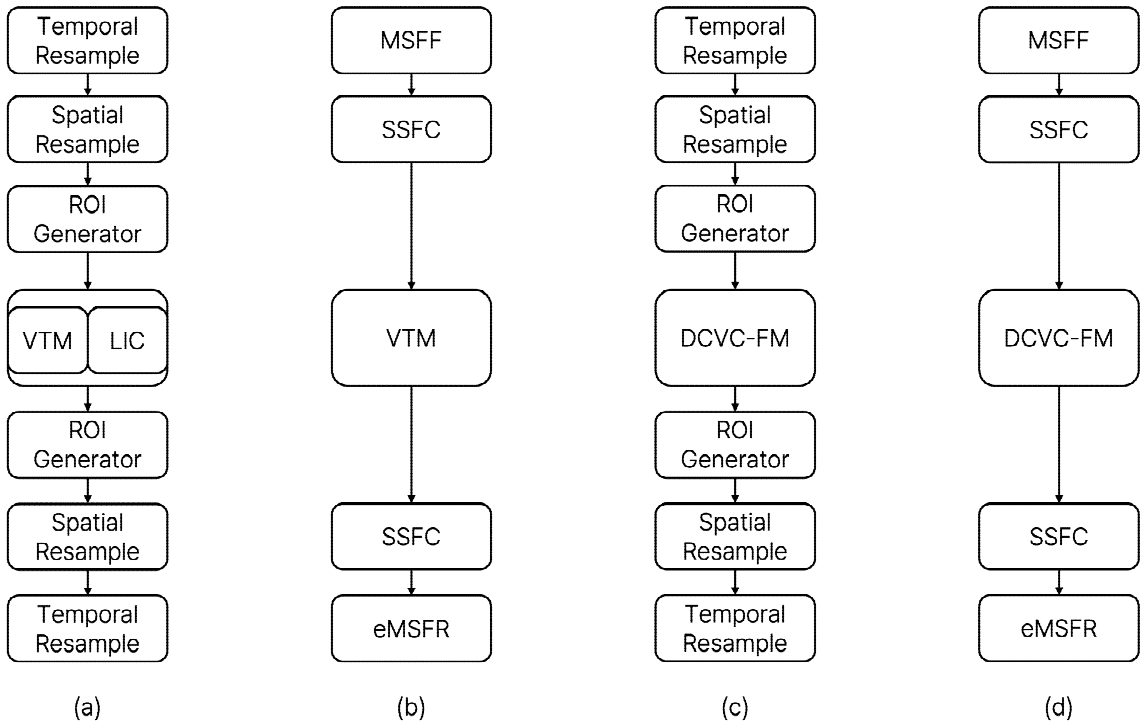


그림 2. 기존의 VCM과 FCM의 구조와 DCVC-FM을 적용한 VCM과 FCM의 구조이다. (a)는 기존의 VCM 설계이고 (b)는 기존의 FCM 설계이다 (c)는 DCVC-FM을 적용한 VCM 설계이고 (d)는 DCVC-FM을 적용한 FCM 설계이다
Fig. 2. Original architecture of VCM and FCM and modified architecture of VCM and FCM with DCVC-FM. (a) is original VCM (b) is original FCM (c) is modified VCM and (d) is modified FCM

SSFC 모듈을 통과하면서 여러 층으로 이루어진 피처 맵을 하나로 합친다. 그다음 VVC를 통해 부, 복호화하고 다시 SSFC, eMSFR 모듈을 통과하면서 원래의 피처 맵으로 복원된다. 이 과정의 MSFF, SSFC, eMSFR과 같은 모듈들은 모두 딥러닝 네트워크이다. 딥러닝 네트워크 모듈 사이에 VVC와 같은 기존 코덱을 넣으면서 딥러닝 네트워크에 대한 End-To-End 훈련이 불가능해지게 되었다. 이로 인해 발생하는 성능 저하를 해결하기 위해 기존의 FCM에서는 파인 튜닝(fine tuning)과 같은 방법을 사용한다. 이러한 과정으로 인해서 VTM의 버전이 바뀌는 등 변경 사항이 있는 경우 제 성능을 발휘하지 못하는 문제가 발생한다.

그림 2의 (c), (d)는 각각 내부 코덱을 DCVC-FM으로 변경한 VCM과 FCM의 설계이다. 본 논문에서는 따로 파인 튜닝 등을 거치지 않고 기존의 모델 weight를 그대로 사용하였다. 영상이 입력되면 기존의 소프트웨어와 동일하게 전처리 과정을 수행한다. FCM의 경우에는 MSFF, SSFC 모듈을 통과한다. 그 다음 교체된 DCVC-FM의 인코더에 영상 정보를 입력하여 부, 복호화한다. 복호화된 영상에 대해 후처리 과정을 수행한다. FCM의 경우에는 SSFC, eMSFR 모듈에 통과시킨다. DCVC-FM의 경우 YCbCr420 포맷을 사용하기 때문에 YCbCr400 포맷을 사용하는 FCM은 피처 맵을 휘도 성분으로 보고 CbCr 성분을 생성하여 입력한다.

본 논문에서는 위와 같은 설계 변경을 통해 End-To-End 딥러닝 기반 코덱이 내부 코덱에 적용된 경우의 성능 향상 가능성을 확인한다. 다만, 딥러닝 기반 코덱이 파인 튜닝되지 않고도 작동할 수 있게끔 하여 기존의 문제를 해결하고자 한다. 또한 현재 표준화가 진행되고 있는 FCM에서도 NN-based inner coding 방식을 진행하고 있다[19]. 하지만, LIC 등에서 사용하는 엔트로피 코딩만을 적용하는 방식이다. 따라서 본 논문에서는 엔트로피 코딩과 더불어서 영상 정보를 활용해 압축하는 DCVC-FM을 적용하였다.

Ⅲ. 실험 및 결과 분석

1. 실험 방법

실험은 VCM과 FCM의 CTC(Common Test Condition)[14,15]에 맞추어 진행하였다. End-To-End 딥러닝 기반 비디오 코덱을 적용한 설계에 대해 실험을 진행할 때는 각 비트레이트 별로 실험하였다. 이를 기존 방식에 CTC 문서상의 QP 값을 적용하여 실험하였을 때 복원된 영상의 초당 비트 수와 유사한 초당 비트 수를 가지는 영상을 찾아 비교하였다.

VCM과 FCM 모두 두 가지의 실험으로 구성된다. 하나는 Object Detection이고, 다른 하나는 Object Tracking이다. 이는 컴퓨터 비전 인공지능이 성능을 얼마나 유지할 수 있는지를 평가하는데, MPEG 회의에 의해 결정된 CTC 문서를 따랐다. 본 논문에서는 Object Detection 실험을 위해 Detectron2[10]를, Object Tracking을 위해서는 Toward-Realtime-MOT[11]을 사용했다. Detectron2에 내장된 Object Detection 네트워크 중 Faster RCNN X101 FPN[12]을 사용해 실험을 진행했으며, Toward-Realtime-MOT에서는 JDE 1088x608[11] 네트워크를 사용했다. Object Detection의 데이터셋으로는 SFU[17]를 사용하였으며, Object Tracking에서는 TVD[18]를 사용하였다.

표 1은 VCM과 FCM의 실험을 위한 데이터셋별 설정값을 나타낸다. VCM과 FCM이 같은 종류의 데이터셋을 사용하지만, 세부적인 설정값이 다르다. 표 1에서 위의 16개 행은 VCM 실험을 위한 설정값이다. VCM의 CTC에서는 Object Detection과 Object Tracking 모두 QP 값을 각각 다른 6개로 설정한다. 반면 FCM의 CTC에서는 Object Detection과 Object Tracking 모두 QP 값을 각각 다른 4개로 설정하였다. 본 논문에서는 CTC 문서에서 설정한 QP 값을 그대로 사용하되, 잘못 설정되어 실제 소프트웨어의 설정값과 차이가 발생한 부분은 실제 소프트웨어의 설정값을 사용하였다.

실험에서 사용된 평가 지표는 mAP(mean Average Precision)[15], MOTA(Multiple Object Tracking Accuracy)[16], mAP@0.5이다. VCM의 Object Detection 실험에서는 mAP가 사용되었는데, FCM의 Object Detection 실험에서는 mAP@0.5가 사용되었다. 여기서 mAP@0.5는 mAP의 threshold를 0.5로 설정했을 때의 값을 의미한다. Object Tracking 실험에서는 VCM과 FCM 모두 MOTA가 사용되었다.

표 1. VCM과 FCM의 실험을 위한 데이터셋 별 설정값. 데이터셋 별 fps(frame per sec)와 입력할 QP값, 실험에서 사용할 프레임의 수 등이다
Table 1. The dataset specific settings for VCM and FCM experiment. This table include the fps, QP values to input, and the number of frames to be used in the experiment

Settings for VCM experiment						
Task	Dataset	Class	Sequence	fps	QP	Frame coded
Object Detection	SFU	A	Traffic	30	{39, 41, 47, 49, 56, 59}	33
		B	ParkScene	24	{32, 37, 40, 44, 48, 52}	33
		B	Cactus	50	{46, 48, 52, 54, 55, 56}	97
		B	BasketballDrive	50	{40, 43, 46, 49, 52, 55}	97
		B	BQTerrace	60	{40, 43, 47, 49, 52, 55}	129
		C	BasketballDrill	30	{27, 31, 35, 39, 43, 47}	65
		C	BQMall	60	{27, 32, 37, 42, 47, 52}	129
		C	PartyScene	50	{31, 34, 39, 43, 47, 52}	97
		C	RaceHorsesC	50	{27, 32, 35, 39, 43, 47}	97
		D	BasketballPass	30	{22, 26, 30, 34, 38, 42}	65
		D	BQSquare	60	{22, 26, 30, 34, 38, 42}	129
		D	BlowingBubbles	50	{27, 31, 34, 37, 40, 43}	97
		D	RaceHorsesD	50	{22, 26, 30, 34, 38, 42}	97
Object Tracking	TVD	TVD	TVD-01	50	{22, 24, 26, 29, 32, 35}	500
		TVD	TVD-02	50	{26, 30, 35, 40, 45, 48}	636
		TVD	TVD-03	50	{34, 39, 42, 46, 50, 54}	500
Settings for FCM experiment						
Task	Dataset	Class	Sequence	fps	QP	Frame coded
Object Detection	SFU	A	Traffic	30	{17,21,24,29}	33
		B	Kimono	24	{35,37,41,45}	33
		B	ParkScene	24	{18,20,32,35}	33
		B	Cactus	50	{41,43,47,49}	97
		B	BasketballDrive	50	{17,21,24,27}	97
		B	BQTerrace	60	{16,19,25,29}	129
		C	BasketballDrill	30	{20,24,32,39}	65
		C	BQMall	60	{19,27,29,32}	129
		C	PartyScene	50	{18,27,32,37}	97
		C	RaceHorsesC	50	{21,32,39,41}	97
		D	BasketballPass	30	{19,27,32,35}	65
		D	BQSquare	60	{17,23,25,31}	129
		D	BlowingBubbles	50	{19,20,21,27}	97
		D	RaceHorsesD	50	{20,25,37,39}	97
Object Tracking	TVD	TVD	TVD-01	50	{17,29,33,37}	500
		TVD	TVD-02	50	{21,25,29,33}	636
		TVD	TVD-03	50	{17,29,33,37}	500

2. 실험 결과

표 2는 VCM에 대한 실험 결과를 나타낸 표이다. VCM-RS는 현재 VCM의 표준 소프트웨어이고, Proposed는 본 논문이 제한하는 방법이다. 각각에 대해 압축 시에 작성된 파일의 비트레이트인 kbps와 복원 후 딥러닝 네트워크에 입력했을 때 측정된 mAP와 MOTA를 표기했다. 단, 데이터셋의 특성에 의해 비교가 불가능한 데이터셋에 대한 결과는 표기하지 않았다. 해당 표를 보면 Object Detection의 경우 비슷한 kbps에서 제안 방법이 훨씬 높은 mAP를 보인다. 이는 딥러닝 기반 코덱이 단일 이미지에 대해서 훨씬 높은 복원률을 보이기 때문이다. 이런 특성 때문에 데이터

표 2. VCM에 End-To-End 딥러닝 코덱을 적용한 실험의 결과표
Table 2. Experiment result table on VCM

Task	Dataset	Sequence	QP	VCM-RS		Proposed	
				kbps	mAP/MOTA	kbps	mAP/MOTA
Object Detection	SFU	BasketballDrill	39	195.57	12.71	193.42	59.19
			35	119.51	10.35	120.65	58.62
			31	72.52	6.61	72.37	58.76
		BQMall	42	236.63	36.24	238.12	31.32
			37	122.05	29.50	123.29	31.66
			32	59.72	23.25	60.09	30.20
		PartyScene	52	420.19	62.53	423.95	58.95
			47	277.69	55.34	279.69	58.66
			43	133.33	50.28	134.96	56.84
		RaceHorsesC	47	526.49	43.27	522.51	67.35
			43	262.86	38.95	262.20	65.61
			39	181.41	38.11	180.46	66.25
			35	109.50	33.86	109.08	65.50
		BasketballPass	30	107.49	12.12	105.93	27.90
			26	64.73	10.27	64.49	27.64
			22	38.42	8.47	38.46	26.85
		BQSquare	34	139.34	31.86	139.84	27.81
			30	77.48	26.30	77.11	28.96
			26	45.29	22.59	44.87	32.33
			22	26.38	16.50	26.45	25.79
		BlowingBubbles	37	153.57	47.54	154.70	25.65
			34	96.62	39.11	96.39	26.53
			31	59.48	26.84	59.62	25.98
			27	35.44	24.53	35.28	25.51
		RaceHorsesD	34	129.18	37.76	130.68	66.27
			30	75.55	29.33	76.49	65.91
			26	44.96	22.40	44.57	65.48
			22	27.09	12.98	33.09	65.28
Object Tracking	TVD	TVD-01	32	271.65	50.30	269.76	51.50
			29	196.36	46.10	194.97	46.30
			26	120.36	39.00	118.96	38.70
			24	71.05	32.60	71.81	31.60

셋 중 역동적이지 않고 물체를 정확하게 특정할 수 있는 이미지에서 더 높은 정확도를 보였다. 반면에 Object Tracking의 경우 비슷한 kbps에서 비슷한 수준의 MOTA 성능을 보였다. 이는 딥러닝 기반 코덱에서 프레임별 특징이 계속해서 다음 프레임으로 전파되면서 앞부분의 특징들이 유실되는 경향을 보이기 때문이다.

표 3은 FCM에 대한 실험 결과를 보여준다. FCTM-v1.0.0은 현재 FCM의 표준 소프트웨어를 나타내며, Proposed는 본 논문이 제안하는 방법이다. 실험 결과 중 두 방법을 비교할 수 있는 결과만 작성하였다. 결과는 비트레이트인 kbps와 복원된 피처 맵을 딥러닝 네트워크에 입력했을 때의 성능 지표인 mAP, MOTA, 그리고 부호화부터 복호화까지 걸린 시간을 측정하여 나타냈다. Object Detection의 경우 유사한 비트레이트에서 제안 방법이 더 우수한 성능을 보였다. 1000kbps 이상에서는 기존 방법에 비해 평균적으로 높은 구간에 고정된다. 특히나 BQMall과 같은 데이터셋에서는 기존 방법보다 15% 이상 높은 성능을 보였다. 다만, 낮은 kbps에서는 많은 정보가 유실되어 기존 방법보다 성능이 낮아졌다. Object Tracking의 경우, 유사한 kbps에서 제안 방법의 성능이 떨어지는 것이 확인되었다. VCM에서와 마찬가지로 적은 비트레이트를 사용하는 실험에서 기존 방법보다 성능이 떨어지는데, 사용하는 비트레이트가 높아질수록 영상 전체의 특징을 더 많이 활용하면서 성능이 개선된다.

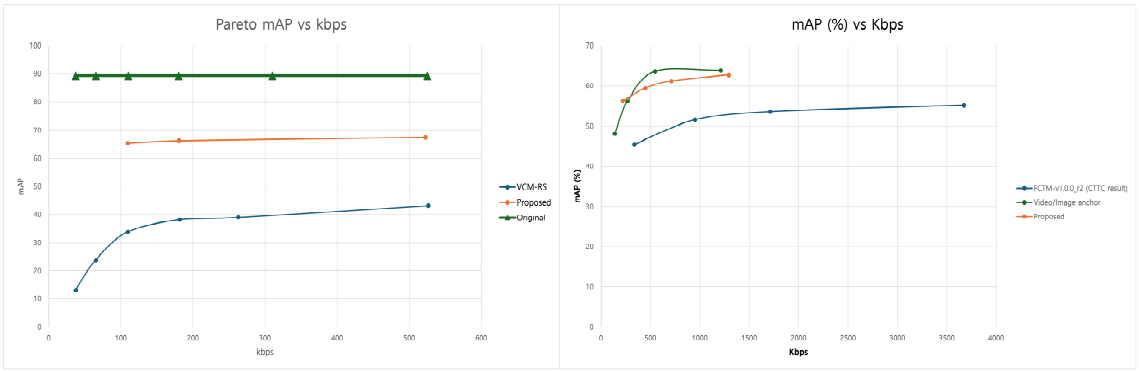
표 3. FCM에 End-To-End 딥러닝 코덱을 적용한 실험의 결과표
Table 3. Experiment result table on FCM

Task	Sequence	QP	FCTM-v1.0.0			Proposed		
			kbps	mAP/MOTA	Time (s)	kbps	mAP/MOTA	Time (s)
Object Detection	Traffic	17	1438.47	69.63	757.69	1321.11	72.29	7.55
		21	941.51	68.85	650.97	1003.78	70.80	7.49
		24	619.35	68.40	525.32	755.00	68.55	7.65
		29	309.22	64.81	331.04	298.04	10.34	7.47
	Kimono	35	73.07	100.00	91.20	106.93	100.00	7.28
	ParkScene	18	1193.73	90.40	725.37	1073.76	86.46	7.36
		20	937.64	89.87	694.15	1015.95	85.46	7.26
		32	139.62	71.77	184.73	146.64	65.12	7.16
		35	86.45	61.95	117.40	81.76	49.73	7.29
	Cactus	41	51.01	100.00	132.86	214.29	100.00	20.79
		43	35.65	100.00	107.66	156.40	99.92	20.95
	BasketballDrive	17	3942.98	75.25	1619.21	3742.34	82.46	20.52
		21	2536.04	73.82	1534.98	2431.10	71.59	20.73
		24	1541.70	71.85	1343.64	1468.31	66.34	20.84
		27	878.99	65.20	1015.10	861.37	61.17	20.70
	BQTerrace	16	2617.66	59.76	1434.00	2371.77	60.60	27.53
		19	1701.40	59.34	1206.14	1827.46	56.55	28.69
		25	621.68	56.58	806.35	772.16	35.32	27.51
		29	327.88	52.31	547.24	413.25	17.59	27.48
	BasketballDrill	20	3538.33	46.42	1957.94	2410.92	45.97	24.74
		24	1958.38	45.87	1746.05	1888.96	45.92	24.83
		32	447.86	42.54	734.20	476.46	35.05	24.78
		39	119.37	29.31	268.35	135.46	1.59	24.66
	BQMall	19	4739.68	68.60	2111.15	3085.89	84.13	34.22
		27	1179.63	66.66	1335.46	1424.95	80.85	32.86
		29	826.50	63.34	1124.55	1067.24	79.16	33.95
		32	443.73	60.50	775.79	435.01	62.98	33.92
	PartyScene	18	3089.90	87.81	1539.03	2476.98	91.54	24.74
		27	690.74	84.16	754.48	843.69	88.72	24.91
		32	258.56	75.03	371.13	470.42	76.11	24.73
		37	96.11	65.08	212.17	207.76	50.12	25.54
	RaceHorsesC	21	2575.94	83.20	998.24	1653.06	82.13	17.40
		32	358.27	82.69	449.68	375.77	79.68	16.60
		39	98.96	77.85	150.37	104.33	57.53	17.19
		41	69.52	74.76	116.56	65.19	27.99	17.22
	BasketballPass	19	5145.32	55.84	2262.19	2667.46	55.97	24.84
		27	1555.94	52.33	1540.61	1668.33	53.02	24.73
		32	604.55	44.15	899.07	764.62	49.83	25.74
		35	344.93	39.55	605.72	325.63	39.59	25.51
	BQSquare	17	3885.65	50.75	2033.39	2702.64	54.15	33.86
		23	1467.26	49.23	1455.50	1669.91	50.13	32.91
		25	965.20	48.17	1252.53	992.79	44.88	32.84
		31	268.17	39.73	557.91	306.41	17.07	33.93
	BlowingBubbles	19	2850.14	80.85	1648.80	2664.35	90.50	24.79
		20	2501.58	81.16	1574.05	2087.21	90.13	24.89
		21	2181.09	80.50	1455.15	1630.89	89.79	25.78
		27	669.88	77.21	848.49	700.18	82.88	24.89
	RaceHorsesD	20	2879.12	79.75	1010.56	1665.84	78.28	16.67
		25	1386.41	78.25	867.92	1330.96	76.77	16.62
		37	145.91	75.17	207.11	168.73	71.82	16.74
		39	100.53	72.70	146.91	104.55	56.27	16.59
Object Tracking	TVD-01	17	823.25	29.27	2195.37	612.77	32.52	424.68
		29	125.92	27.60	2014.46	397.40	28.43	424.03
		33	62.42	23.71	1798.32	282.69	22.90	423.32
		37	29.52	11.67	1521.17	182.13	12.08	422.16

기존 방법과 제안 방법 간의 수행 시간을 비교하면 제안 방법이 훨씬 개선된 모습을 보인다. 이런 모습을 볼 때, 현 FCM의 전, 후 네트워크에서 VVC의 복원 오류를 보완하기 위한 파인 튜닝 작업을 제외하고 내부 코덱까지 End-To-End 훈련 과정을 거친다면 본 논문에서 실험한 방법보다 더 높은 성능을 보일 수 있을 것으로 보인다.

본 논문에서는 기존 방법과 제안 방법의 비교를 위해, 각 컴퓨터 비전 네트워크를 통해 측정된 mAP와 MOTA가 같은 비트레이트 상에서 어느 정도 개선되었는지 비교했다. 그림 3은 해당 비교의 그래프이다. 그림 3의 (a)는 VCM에서 Object Detection에 대해 mAP를 비교한 것이다. VCM에서 제안 방법의 mAP가 적게는 10에서 많게는 30까지 차이가 발생했다. 그림 3의 (b)는 같은 데이터셋에서 FCM을 비교한 그래프이다. 평균적으로 더 낮은 비트레이트에서 높은 mAP 점수를 보였다. (c)는 VCM에서 MOTA를 비교한 그래프이다. VCM의 경우 MOTA에서 비슷한 점수를 보이고 있는데, 앞서 설명하였듯이 많은 특징들이 학습되었을 때, 앞부분의 특징이 유실되면서 발생하는 것으로 확인하였다. (d)는 FCM에서 MOTA를 비교한 그래프이다. Object Detection에서와 마찬가지로 평균적으로 더 높은 MOTA 성능을 보였다. 또한 기존 방법과 제안 방법의 부호화와 복호화에 걸리는 시간을 비교하면, 기존 방법은 평균적으로 1018.24초의 수행 시간이 측정되었으며, 제안 방법은 평균적으로 298.658초의 수행 시간이 측정되었다. 따라서 제안 방법의 수행 시간이 기존 방법의 수행 시간보다 빠른 것으로 확인됐다.

이러한 실험 결과를 통해, VCM과 FCM의 내부 코덱을 딥러닝 기반의 비디오 코덱으로 변경함으로써 VCM과 FCM의 성능을 모두 극대화할 수 있을 것으로 기대할 수 있다. 본 논문에서는 딥러닝 기반 비디오 코덱을 별도로 훈련시키는 과정을 수행하지 않았다. 그러나, 딥러닝 기반 비디오 코덱을 포함하여 End-To-End로 훈련하는 과정을 거



(a) SFU, RaceHorsesC result by VCM

(b) SFU, ClassD result by FCM

(c) TVD, TVD-01 result by VCM

(d) TVD, Overall result by FCM

그림 3. VCM과 FCM의 실험 결과 그래프
Fig. 3. Graph for experiment result on VCM and FCM

치면 Object Detection 뿐 아니라, Object Tracking에서도 성능을 더욱 높일 수 있을 것이다.

Ⅳ. 결 론

VCM과 FCM은 딥러닝 기반의 인공지능 엔진이 더 적은 데이터로도 성능을 유지할 수 있는 압축 기술 개발을 목표로 표준화를 진행하고 있다. 현재는 VVC를 내부 코덱으로 하여 LIC를 옵션으로 사용하고 있다. 이 때문에 기존의 VCM과 FCM은 복원률 저하 문제, 공간 복잡도 문제 등이 남아있다. 또, 기존 방식의 비디오 코덱을 사용함에 따라 End-To-End 훈련을 진행할 수 없으며, 파인 튜닝 과정을 거치므로 내부 코덱이 바뀌었을 때 성능이 저하된다. 본 논문의 실험 결과에서 볼 수 있듯이, 내부 코덱에 End-To-End 딥러닝 기반의 비디오 코덱을 적용한 경우에도 컴퓨터 비전 인공지능이 충분히 제 성능을 발휘한다. 또한 이 경우에 하나의 딥러닝 네트워크를 사용할 수 있기 때문에 공간 복잡도 문제를 해결할 수 있을 것이며, End-To-End 훈련을 진행할 수 있기 때문에 추후 최적화된 성능을 발휘할 수 있을 것으로 예상된다. 본 연구를 통해 VCM과 FCM의 내부 코덱을 딥러닝 기반 코덱으로 변경하더라도 성능이 유지, 향상됨을 알 수 있었으며 기존 코덱의 문제들을 해결할 수 있는 가능성을 확인하였다.

참 고 문 헌 (References)

[1] D. Lee, S. Jeon, Y. Jeong, J. Kim, and J. Seo, “Exploring the Video Coding for Machines Standard: Current Status and Future Directions,” Jounal of Broadcast Engineering, Vol. 28, No. 7, pp.888-903, December 2023.

[2] I. O. f. Standardization, “ISO/IEC 23090-3:2021 – Information technology – Coded representation of immersive media – Part 3: Versatile video coding,” 2021.

[3] Cheng, Z., Sun, H., Takeuchi, M., & Katto, J. Learned image compression with discretized gaussian mixture likelihoods and attention modules. InProceedings of the IEEE/CVF conference on computer vision and pattern recognition(pp. 7939-7948), 2020.

[4] Sang-Kyun Kim et. al., [VCM] CfP response: Region-of-interest based video coding for machine, ISO/IEC JTC1/SC29/WG2/m60758, October 2022.

[5] Li, Jiahao, Bin Li, and Yan Lu. “Neural video compression with diverse contexts.” Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2023.

[6] Li, Jiahao, Bin Li, and Yan Lu. “Neural Video Compression with Feature Modulation.” arXiv preprint arXiv:2402.17414(2024).

[7] Ballé, Johannes, Valero Laparra, and Eero P. Simoncelli. “End-to-end optimized image compression.” arXiv preprint arXiv:1611.01704. 2016.

[8] Lu, G., Ouyang, W., Xu, D., Zhang, X., Cai, C., and Gao, Z. (2019). Dvc: An end-to-end deep video compression framework. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition(pp. 11006-11015). 2019.

[9] Ranjan, Anurag, and Michael J. Black. “Optical flow estimation using a spatial pyramid network.” Proceedings of the IEEE conference on computer vision and pattern recognition. pp.4161-4170. 2017.

[10] Detectron2, https://github.com/facebookresearch/detectron2, (accessed Feb. 1. 2024).

[11] Wang, Z., Zheng, L., Liu, Y., Li, Y., and Wang, S. “Towards real-time multi-object tracking.” European conference on computer vision. Cham: Springer International Publishing, pp. 107-122, August 2020. https://github.com/Zhongdao/Toward-Realtime-MOT, (accessed Dec. 10. 2023.)

[12] Ren, S., He, K., Girshick, R., and Sun, J. “Faster R-CNN: Towards real-time object detection with region proposal networks.” IEEE transactions on pattern analysis and machine intelligence, Vol 39, No.6, pp1137-1149, 2016.

[13] Liu, S., H. Zhang, and C. Rosewarne, Common test conditions for video coding for machines, ISO/IEC JTC1/SC29/WG4/wg04n311, 2023. 06.

[14] Call for Proposal on Feature Compression for Video Coding for Machines update, ISO/IEC JTC1/SC29/WG2/w22911, 2023. 07.

[15] Henderson, Paul, and Vittorio Ferrari, “End-to-end training of object class detectors for mean average precision,” Computer Vision – ACCV 2016: 13th Asian Conference on Computer Vision, Taipei, Taiwan, November pp.20-24, 2016, Revised Selected Papers, Part V 13. Springer International Publishing, 2017.

[16] Bernardin, Keni, and Rainer Stiefelhagen, "Evaluating multiple object tracking performance: the clear mot metrics,” EURASIP Journal on Image and Video Processing, pp.1-10, 2008.

[17] Choi, H., Hosseini, E., Ranjbar Alvar, S., Cohen, R., Bajić and I. “SFU-HW-Objects-v1: Object labelled dataset on raw video sequences,” Federated Research Data Repository. doi: https://doi.org/10.25314/7d8efc0a-3943-4738-b7a5-72badb04d765, (accessed 2020.)

[18] X. Xu, S. Liu and Z. Li, “A Video Dataset for Learning-based Visual Data Compression and Analysis,” International Conference on Visual Communications and Image Processing (VCIP), Dec. 2021.

[19] FCM CE 3 NN-based inner coding, ISO/IEC JTC1/WG4/wg04n 00463, 2024.04.

저 자 소 개

전 상 균

- 2022년 3월 : 동아대학교 컴퓨터AI공학부 컴퓨터공학과 입학
- ORCID : https://orcid.org/0009-0004-4287-6745
- 주관심분야 : 멀티미디어, 오디오/비디오 부호화, 컴퓨터 비전, 딥러닝

이 동 민

- 2020년 3월 : 동아대학교 컴퓨터공학과 입학
- ORCID : https://orcid.org/0009-0007-2009-8753
- 주관심분야 : 머신러닝, 딥러닝, 비디오 부호화

추 현 곤

- 1998년 2월 : 한양대학교 전자공학과(공학사)
- 2000년 2월 : 한양대학교 전자공학과(공학석사)
- 2005년 2월 : 한양대학교 전자통신전파공학과(공학박사)
- 2015년 ~ 2017년 : 한국전자통신연구원 디지털홀로그래피연구실장
- 2017년 ~ 2018년 : Warsaw University of Technology, Poland 방문연구원
- 2023년 ~ 현재 : 한국전자통신연구원 실감미디어연구실장
- ORCID : https://orcid.org/0000-0002-0742-5429
- 주관심분야 : Computer Vision, 3D imaging and holography, 3D depth imaging, 3D broadcasting system

서 정 일

- 2005년 2월 : 경북대학교 전자공학과(공학박사)
- 1998년 ~ 2000년 : LG반도체 선임연구원
- 2000년 ~ 2023년 : 한국전자통신연구원 실감미디어연구실장
- 2023년 ~ 현재 : 동아대학교 컴퓨터공학과 부교수
- ORCID : https://orcid.org/0000-0001-5131-0939
- 주관심분야 : 멀티미디어, 오디오/비디오 부호화, 딥러닝, 머신 비전