MOTR: End-to-End Multiple-Object Tracking with TRansformer

School of Industrial and Management Engineering, Korea University

Jin Hyeok Park

KOREA UNIVERSITY

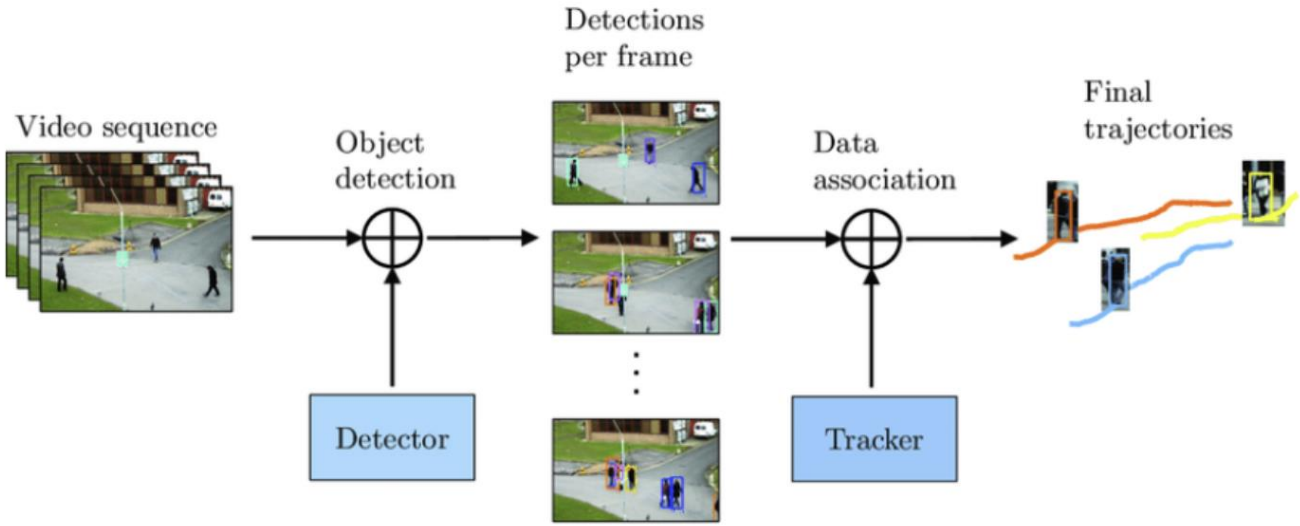
DMQA

Contents

Introduction

Multiple Object Tracking(MOT)

- 영상 내 다양한 object의 trajectory를 tracking하는 연구 분야
- 다양한 object를 동시에 처리함과 동시에 오랜 시간동안 정보의 tracking이 가능해야 함
- 모든 프레임에서 object 위치를 추출하기 어려운 상황을 해결하기 위해 Tracking-by-detection을 사용



<Tracking-by-detection>

DMQA

Overview of MOTR

MOTR

- MEGVII Technology
- 2021년 8월 24일 기준 1회 인용
- Transformer를 multi object tracking에 적용한 최초의 연구

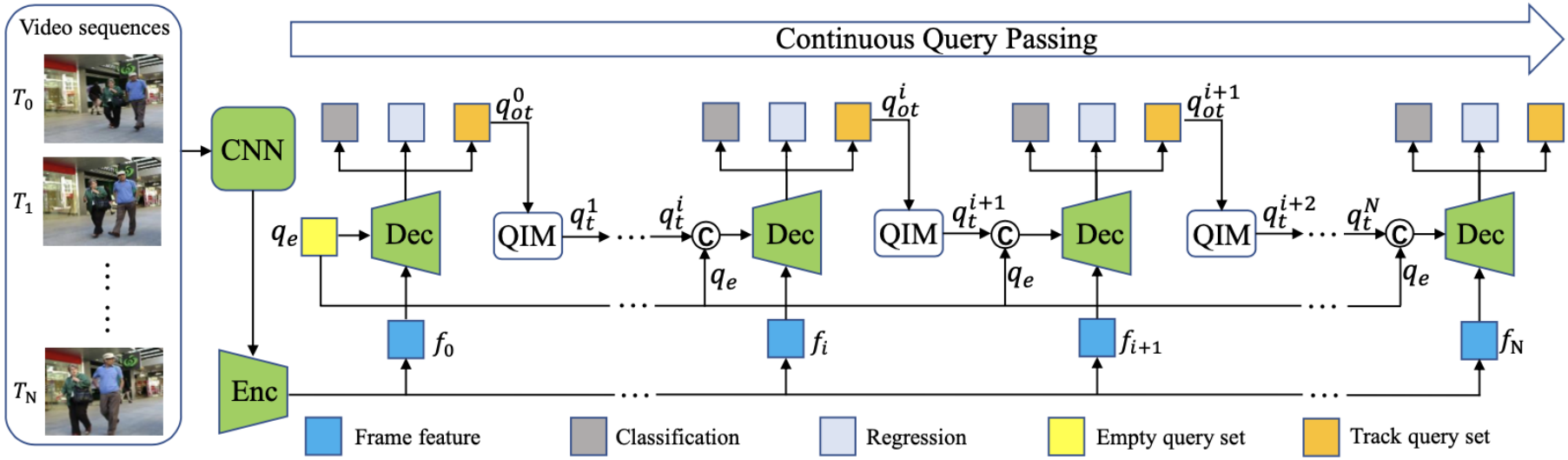
MOTR: End-to-End Multiple-Object Tracking with TRansformer

Fangao Zeng * Bin Dong * Tiancai Wang * Cheng Chen
Xiangyu Zhang Yichen Wei
MEGVII Technology

Overview of MOTR

MOTR

- End-to-End로 연결된 architecture
- Track query과 contiguous passing mechanism활용
- Temporal aggregation network(TAN) 활용



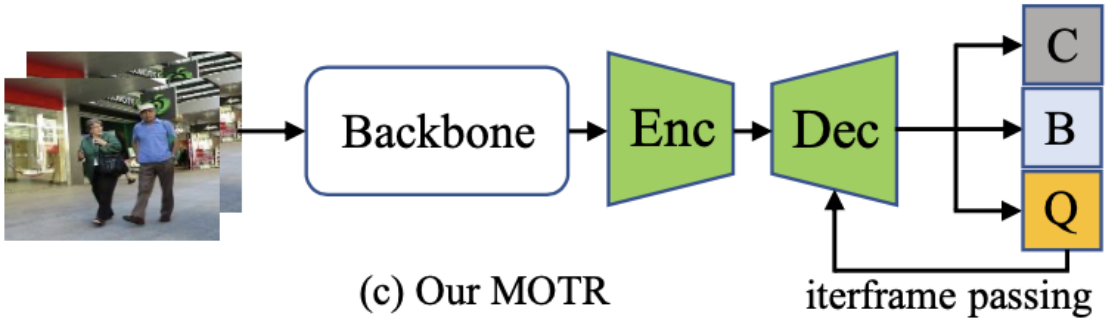
<MOTR >

DMQA

MOTR Architecture

Track Query

- MOTR은 frame마다 track query set을 예측함
- Track query set은 각 object가 영상 내에서 등장부터 사라질때까지의 전체 track을 예측함
- Decoder에 track query set이 들어가면 현재 frame에 대한 tracking prediction을 생성
- Output으로 업데이트된 track query setd은 다음 frame의 decoder에 input으로 사용됨
- 전체 비디오 내 frame별로 위의 단계를 반복하며 이를 continuous query passing 부름



(c) Our MOTR

MOTR Architecture

Track Query

- ❖ DETR의 object query는 특정 object에 대해서만 예측하지 않아 다른 object에 대한 track을 예측 할 수 있음
- ❖ 반면 track query는 특정 frame내에서 object에 매칭 되면 object가 사라질때까지 해당 object만 예측함
- ❖ 결과적으로 예측된 track과 object 사이의 연결 또는 NMS 없이 end-to-end 구조를 가짐

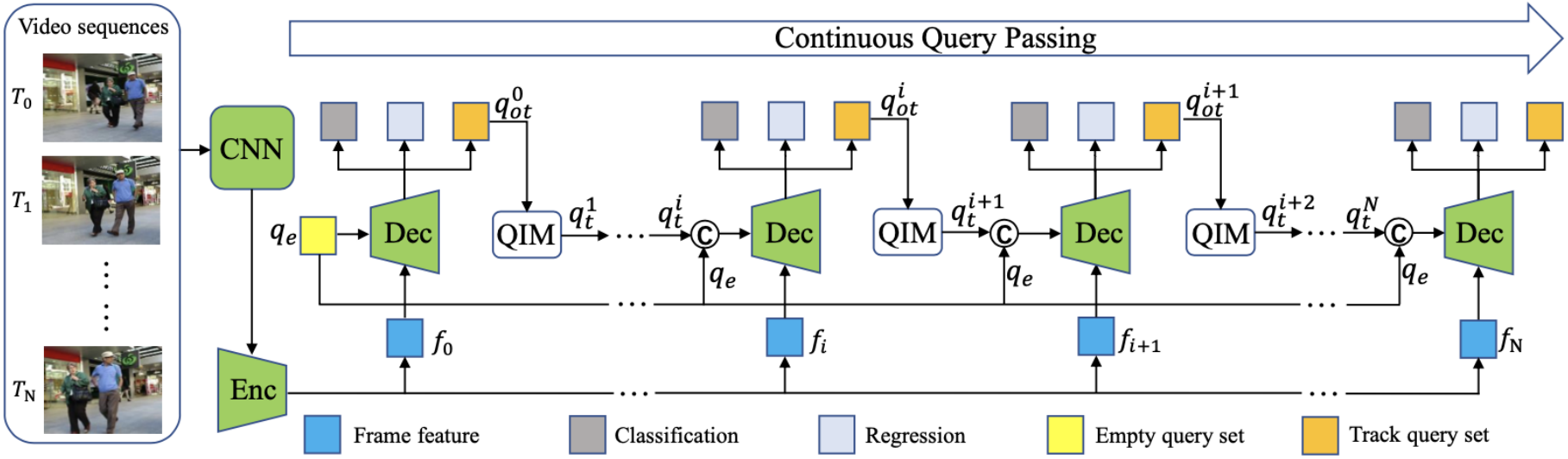
(a) Detection query for object detection.

(b) Track query for multiple-object tracking.

MOTR Architecture

Continuous Query Passing

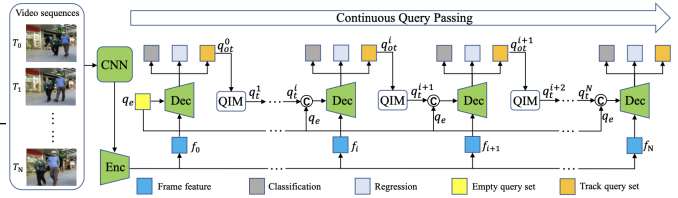
- 비디오가 cnn에 input으로 입력된 후, Deformable DETR의 encoder를 거쳐 $f = f_0, f_1, \ldots, f_N$을 추출함
- MOTR의 encoder와 decoder는 Deformable DETF를 참조함



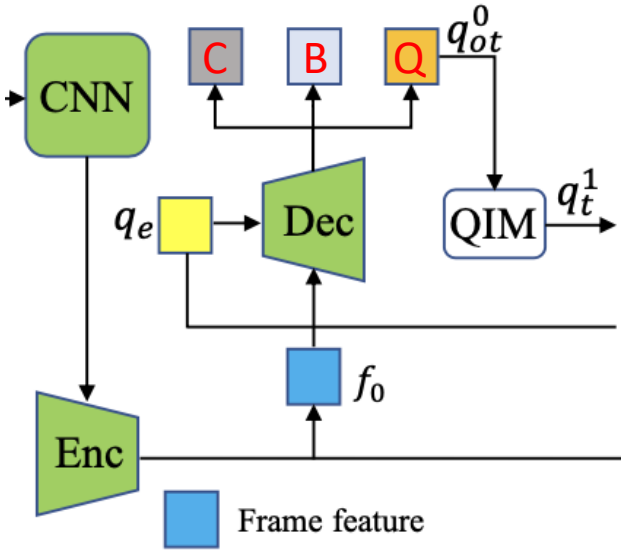
DMQA

MOTR Architecture

Continuous Query Passing



- 비디오 frame T_0의 Dec input으로는 f_0와 비어있는 query set인 q_e가 들어옴
- f_0와 q_e를 통해 모든 object의 초기 위치를 추출하고 original track query set인 q_{ot}^1를 생성
- q_{ot}^1은 Query Interaction Module(QIM)을 통과한 뒤 frame T_1에 넘겨줄 q_t^1를 생성
- 최종적으로 q_t^1과 f_1이 다음 frame의 Dec input으로 들어가게 됨

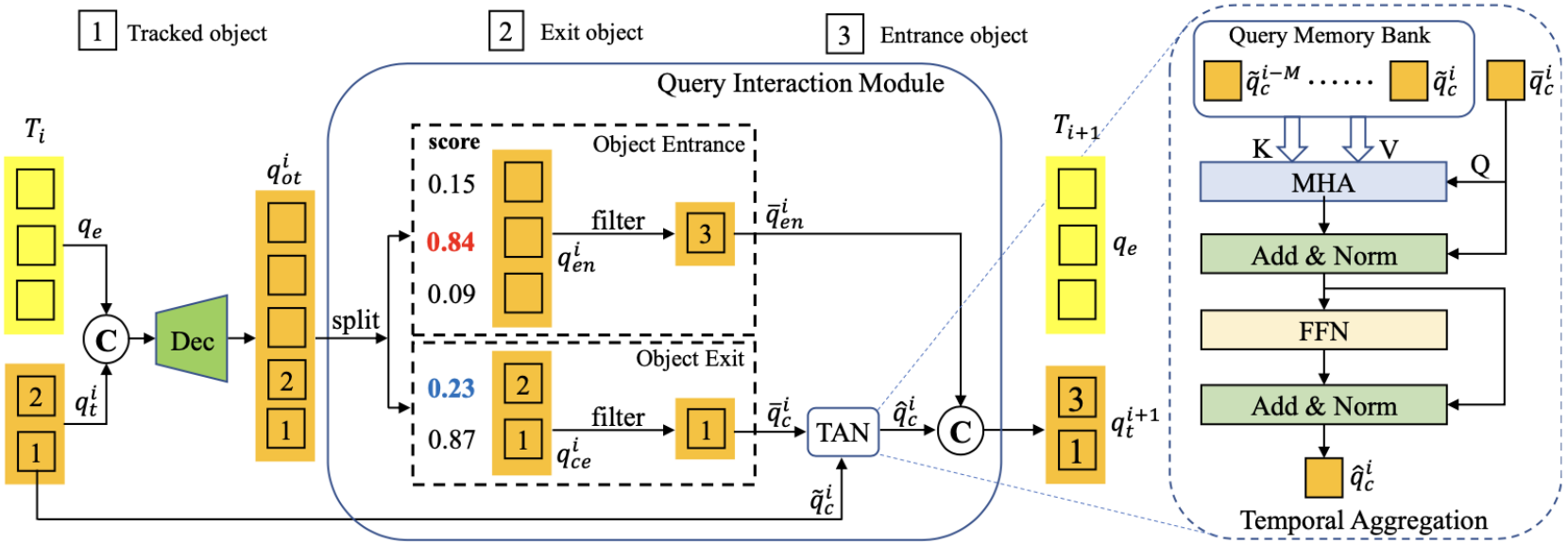


DMQA

MOTR Architecture

Query Interaction Module(QIM) & Temporal Aggregation Network(TAN)

- ❖ TAN은 과거에 처리한 frame들에서 추적한 object들에 대한 query를 모으고 한번에 처리함
- ❖ 현재 frame의 track query는 multi-head attention을 통해 TAN내의 각각의 query와 상호작용함
- ❖ TAN은 과거의 object에 대한 query를 모으고 한번에 처리해주는 Query memory bank와 같은 역할을 함

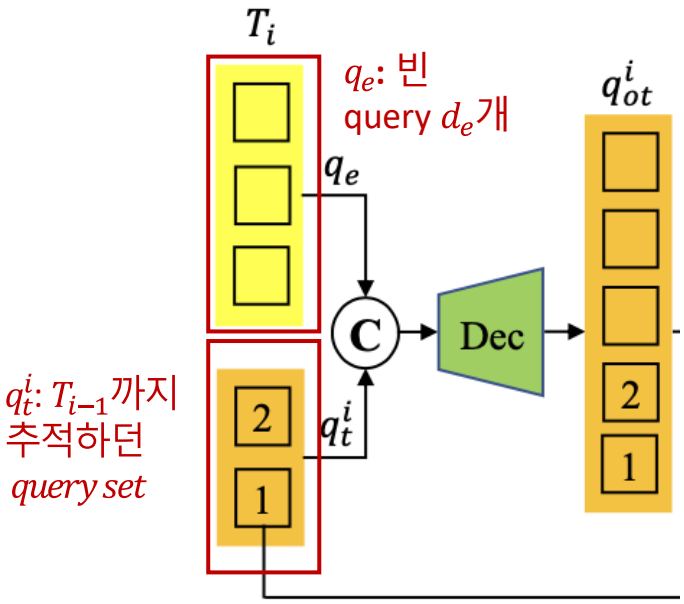


DMQA

MOTR Architecture

Query Interaction Module(QIM) & Temporal Aggregation Network(TAN)

- MOTR의 track query는 특정 object에 매칭되면 해당 object만 추적함
- 따라서 매 frame별로 새로 등장하는 object에 매칭할 빈 query가 필요함
- 매 frame마다 사라지는 object가 있으면 추적중인 query를 제거해야함
- Training 과정에서 새로 등장하거나 사라지는 객체의 판별은 track score를 사용함
- T_i 번째 frame의 Dec input은 이전시점에서 받은 q_t^i와 빈 q_e를 병합하여 input으로 받음
- Dec가 q_{ot}^i를 track score와 함께 생성한 뒤 index q_e를 기준으로 두개의 query set으로 분할함

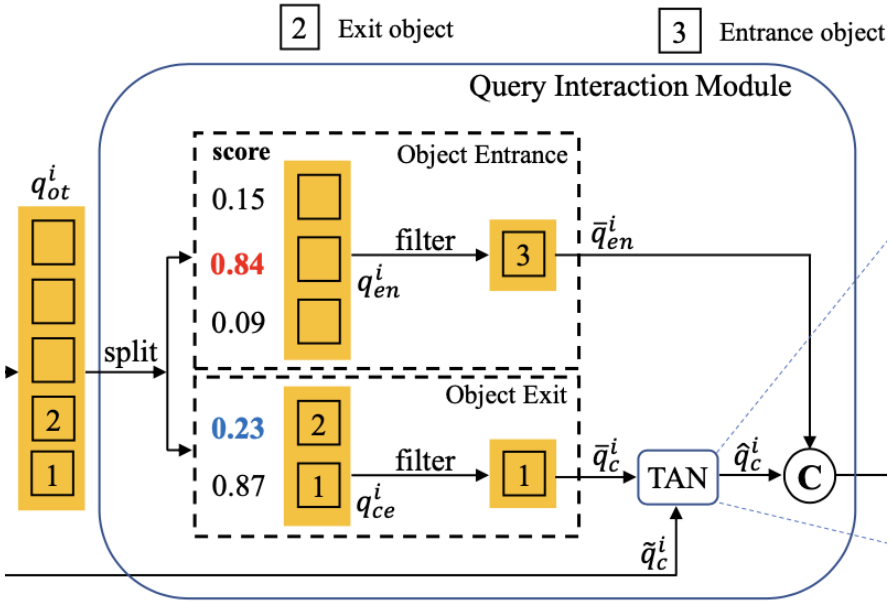


DMQA

MOTR Architecture

Encoder

- 0번째부터 $d_e - 1$까지와 d_e부터 마지막 index로 분리 후 q_{en}^i는 entrance object의 판별 담당
- q_{ce}^i는 tracked와 exit object의 판별을 담
- Entrance: 미리 지정된 임계치값보다 score가 큰 경우에 새로 생성된 object라 판단하고 query만 남긴 뒤, score가 작은 query는 제거함
- Tracked &Exit: 이전의 연속된 총 M개의 frame정보를 확인하여 M개의 query score의 최댓값이 지정된 임계치값보다 작을 경우 사라진 object라 판단하여 제거



DMQA

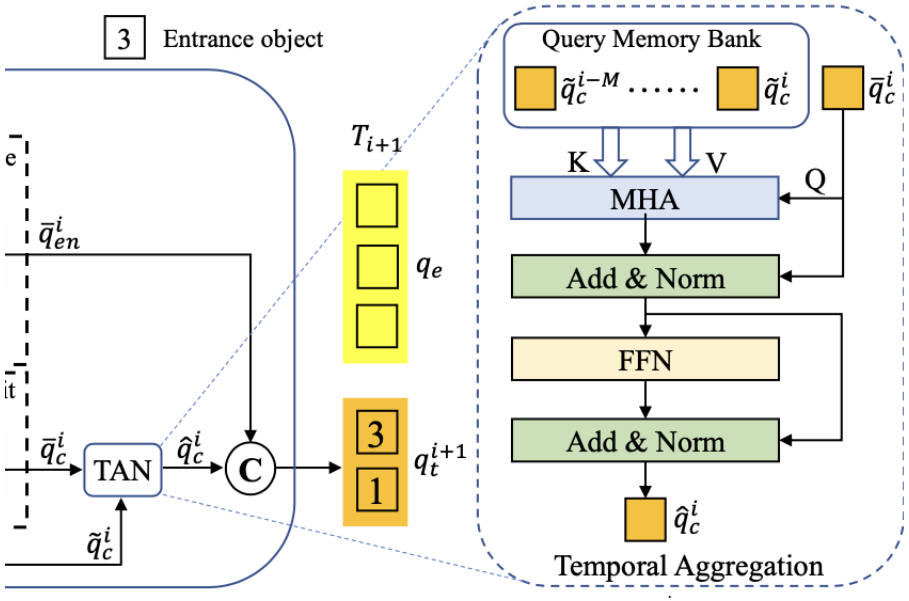
MOTR Architecture

Query Interaction Module(QIM) & Temporal Aggregation Network(TAN)

- Tracked query는 TAN을 통해 이전의 query memory bank와 상호작용 후 새로 build-up
- Memory bank $q_{bank} = \{\tilde{q_c}^{i-M} \ldots, \tilde{q_c}^{i}\}$는 과거 M개의 frame을 병합하여 생성됨
- M개의 frame 병합을 통해 tgt생성
- tgt가 MHA모듈에 input으로 들어가 attention weight를 생성
- tgt와 q_c^{-i}와 내적을 통해 계산
- 최종적으로 q_{sa}^i가 FFN과 normalized를 거친 후 최종 output $\hat{q}_c^i$를 생성함

$$tgt = \tilde{q}_c^{i-M} \oplus \cdots \tilde{q}_c^{i-1} \oplus \tilde{q}_c^i$$

$$q_{sa}^i = \sigma_s(\frac{tgt \cdot tgt^T}{\sqrt{d}}) \cdot \bar{q}_c^i$$



DMQA

Result

Experiments

- ❖ 다른 MOT모델에 비해 MOTR은 월등한 성능을 보여줌
- ❖ IDS가 다른 모델에 비해 크게 감소(IDS: 객체 추적 중 둘 이상의 객체가 겹쳐 객체 id가 바뀌는 것)

Dataset	Tracker	Public Detection	MOTA↑	IDF1↑	MT (%) ↑	ML (%)↓	FP↓	FN↓	IDS↓
MOT16	FWT[13]	✓	47.8	44.3	19.1	38.2	8886	85487	852
	MOTDT[9]	✓	47.6	50.9	15.2	38.3	9253	85431	792
	GCRA[21]	✓	48.2	48.6	12.9	41.1	5104	88586	821
	EAMTT[30]		52.5	53.3	19.0	34.9	**4407**	81223	910
	Tracktor++[1]	✓	54.4	52.5	19.0	36.9	**3280**	79149	682
	SORTwHPD16[2]		59.8	53.8	25.4	22.7	8698	63245	1423
	smartSORT[23]		60.4	56.1	28.9	21.2	11183	59867	1135
	DeepSORT_2[41]		61.4	62.2	32.8	**18.2**	12852	56668	781
	JDE[39]		64.4	55.8	**35.4**	**20.0**	/	/	1544
	MOTR (Ours)		**65.7**	**67.0**	**37.2**	20.9	16512	**45340**	**648**
	MOTR (Ours)	✓	**65.8**	**67.1**	32.5	27.4	9914	**51965**	**547**
MOT17	MHT DAM[14]	✓	50.7	47.2	20.8	36.9	22875	252889	2314
	MOTDT17[9]	✓	50.9	52.7	17.5	35.7	24069	250768	2474
	FWT[13]	✓	51.3	47.6	21.4	35.2	24101	247921	2648
	SST[36]		52.4	49.5	21.4	30.7	25423	234592	8431
	Tracktor++[1]	✓	53.5	52.3	19.5	36.6	**12201**	248047	2072
	Tracktor v2[1]	✓	56.5	55.1	21.1	35.3	**8866**	235449	3763
	CenterTrack[45]	✓	61.5	59.6	26.4	31.9	14076	200672	2583
	TrackFormer [22]	✓	61.8	59.8	/	/	35226	177270	2982
	TransTrack[35]		65.8	56.9	32.2	**21.8**	24000	163683	5355
	CTracker[25]		**66.6**	57.4	32.2	**24.2**	22284	160491	5529
	MOTR (Ours)		65.1	**66.4**	**33.0**	25.2	45486	**149307**	**2049**
	MOTR (Ours)	✓	**66.5**	**67.0**	**33.5**	26.2	31302	**155715**	**1884**

Conclusion

❖ MOTR: End-to-End Multiple-Object Tracking with TRansformer

- End-to-End로 연결된 architecture
- Track query과 contiguous passing mechanism활용
- Temporal aggregation network(TAN) 활용

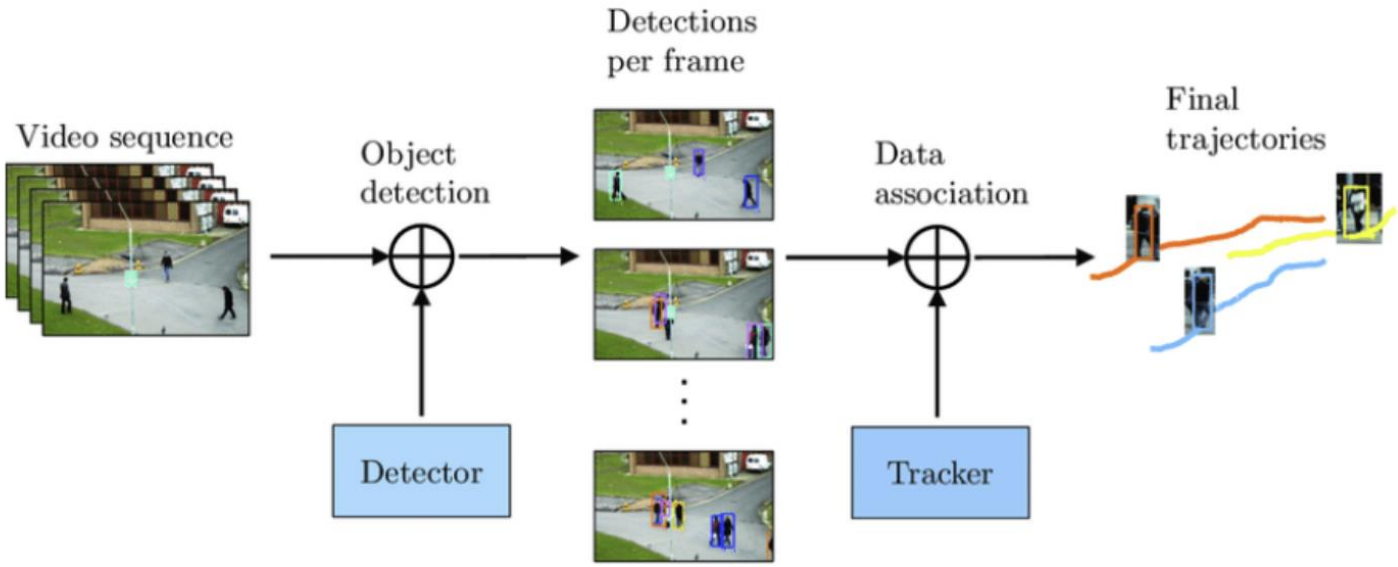

Thank You

Appendix

Appendix

❖ Tracking-by-detection

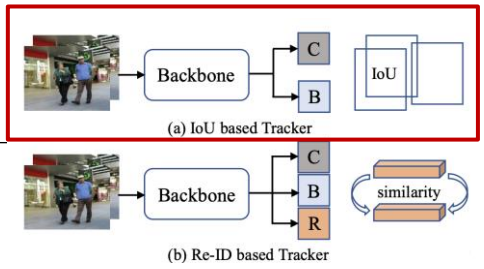
- Object Detection: Detector가 순간 순간의 frame에서 객체의 위치를 추출함
- Data Association: Tracker가 앞뒤 frame의 위치정보를 연결해서 trajectory 예측



<Tracking-by-detection>

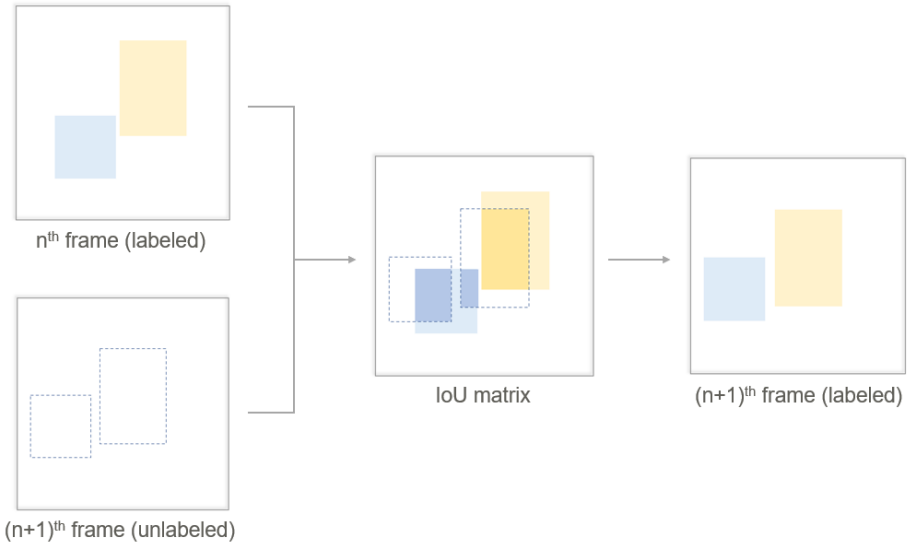
Appendix

DETR Architecture



❖ IoU based Tracker

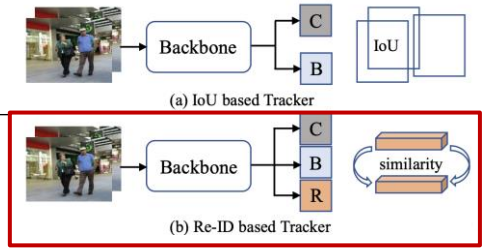
- 각 frame에서 추출한 bounding box의 IoU값을 계산하여 연결함
- IoU matrix를 통해 겹치는 영역이 일정 threshold 이상이면 같은 object로 판단
- Box regression과 classification을 이용함



DMQA

Appendix

DETR Architecture



❖ Re-ID based Tracker

- IoU tracking 방식에서 Re-ID에 해당하는 방식을 추가함
- Rd-ID feature embedding을 예측하고 두 frame에서 나온 feature의 형태적 유사도로 tracking

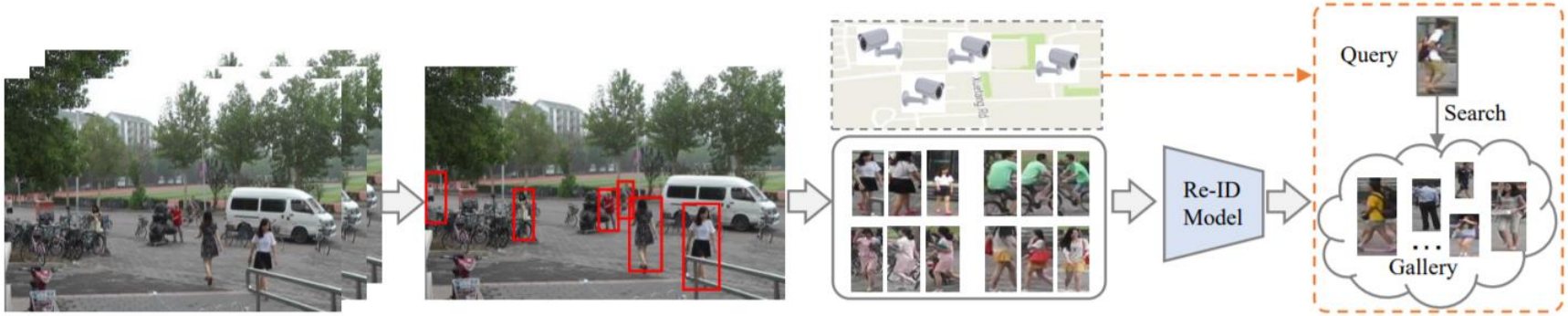


Fig. 1: The flow of designing a practical person Re-ID system, including five main steps: 1) *Raw Data Collection*, (2) *Bounding Box Generation*, 3) *Training Data Annotation*, 4) *Model Training* and 5) *Pedestrian Retrieval*.

DMQA