



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2023-0017679
(43) 공개일자 2023년02월06일

(51) 국제특허분류(Int. Cl.)
G06F 18/00 (2023.01) G06F 17/15 (2006.01)
G06N 20/00 (2019.01) G06N 5/04 (2023.01)
G06T 1/20 (2018.01)
(52) CPC특허분류
G06V 20/49 (2022.01)
G06F 17/153 (2013.01)
(21) 출원번호 10-2021-0099495
(22) 출원일자 2021년07월28일
심사청구일자 없음

(71) 출원인
현대모비스 주식회사
서울특별시 강남구 테헤란로 203 (역삼동)
(72) 발명자
이재영
경기도 용인시 처인구 중부대로1158번길 12, 201
동 1504호 (삼가동, 행정타운늘푸른오스카빌아파
트)
(74) 대리인
특허법인지명

전체 청구항 수 : 총 9 항

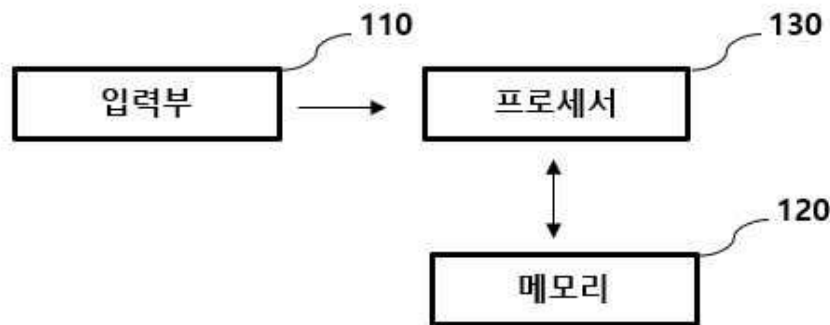
(54) 발명의 명칭 영상 의미 분할 시스템 및 그 방법

(57) 요약

본 발명은 영상 의미 분할 시스템 및 그 방법에 관한 것으로, 보다 상세하게는 최적의 자원 사용량으로 객체의 기하학적 변환을 수용하여 자율주행 시스템의 성능 및 신뢰도를 향상시키는 영상 의미 분할 시스템 및 그 방법에 관한 것이다.

본 발명에 따른 영상 의미 분할 시스템은 영상 데이터를 수신하는 입력부와, 영상 데이터에 대한 의미 분할을 수행하는 프로그램이 저장된 메모리 및 프로그램을 실행시키는 프로세서를 포함하고, 프로세서는 영상 데이터에 대한 입력 채널 수를 압축하는 압축된 변형 합성곱 연산을 제1 경로 및 제2 경로에 따라 병렬로 수행하되, 제2 경로에 대해서는 풀링하여 입력시키는 것을 특징으로 한다.

대표도 - 도3



(52) CPC특허분류

G06N 20/00 (2021.08)

G06N 5/04 (2023.01)

G06T 1/20 (2013.01)

명세서

청구범위

청구항 1

영상 데이터를 수신하는 입력부;

상기 영상 데이터에 대한 의미 분할을 수행하는 프로그램이 저장된 메모리; 및

상기 프로그램을 실행시키는 프로세서를 포함하고,

상기 프로세서는 상기 영상 데이터에 대한 입력 채널 수를 압축하는 압축된 변형 합성곱 연산을 제1 경로 및 제2 경로에 따라 병렬로 수행하되, 상기 제2 경로에 대해서는 풀링하여 입력시키는 것

인 영상 의미 분할 시스템.

청구항 2

제1항에 있어서,

상기 압축된 변형 합성곱은 출력 채널 수의 증가를 고려하여 커널 크기만큼 상기 입력 채널 수를 압축하는 것

인 영상 의미 분할 시스템.

청구항 3

제2항에 있어서,

상기 프로세서는 추론 과정에서 정수형 오프셋을 사용하여 메모리 사용량을 감소시키는 것

인 영상 의미 분할 시스템.

청구항 4

제1항에 있어서,

상기 프로세서는 상기 제1 경로 및 제2 경로에 따른 오프셋 추정 결과의 유효도를 고려하여 의미 분할에 반영하는 것

인 영상 의미 분할 시스템.

청구항 5

제1항에 있어서,

상기 제1 경로 및 제2 경로에서 오프셋을 구하는 합성곱과 특징을 처리하는 합성곱은 각각 공유되는 커널을 사용하는 것

인 영상 의미 분할 시스템.

청구항 6

(a) 영상 데이터를 수신하고, 입력 채널 수를 압축하는 압축된 변형 합성곱 연산을 제1 경로에 따라 수행하는

단계; 및

(b) 상기 제1 경로와 병렬 구조인 제2 경로에서, 입력에 대한 풀링 후 상기 압축된 변형 합성곱 연산을 수행하는 단계

를 포함하는 영상 의미 분할 방법.

청구항 7

제6항에 있어서,

상기 (a) 및 (b) 단계에서의 압축된 변형 합성곱 연산은 출력 채널 수 증가를 고려하여 커널 크기만큼 상기 입력 채널 수를 압축하는 것

인 영상 의미 분할 방법.

청구항 8

제6항에 있어서,

상기 (a) 및 (b) 단계에서의 압축된 변형 합성곱 연산은 추론 과정에서 정수형 오프셋을 사용하여 메모리 사용량을 감소시키는 것

인 영상 의미 분할 방법.

청구항 9

제6항에 있어서,

상기 (a) 단계가 수행되는 제1 경로 및 상기 (b) 단계가 수행되는 제2 경로에서, 오프셋을 구하는 합성곱과 특징을 처리하는 합성곱은 각각 공유되는 커널을 사용하는 것

인 영상 의미 분할 방법.

발명의 설명

기술 분야

[0001] 본 발명은 영상 의미 분할 시스템 및 그 방법에 관한 것으로, 보다 상세하게는 최적의 자원 사용량으로 객체의 기하학적 변환을 수용하여 자율주행 시스템의 성능 및 신뢰도를 향상시키는 영상 의미 분할 시스템 및 그 방법에 관한 것이다.

배경 기술

[0003] 종래 기술에 따른 의미 분할 방법은 자율주행 시스템 구동을 위해 도로 영상에서 차선, 자유 공간 등의 정보를 제공한다.

[0004] 종래 기술에 따른 deformable convolution은 임시 특징 맵(feature map)의 크기가 증가되고, GPU 메모리의 사용량이 증가되어 모듈 구성이 어려운 문제점이 있다.

발명의 내용

해결하려는 과제

[0006] 본 발명은 전술한 문제점을 해결하기 위해 제안된 것으로, 자원 사용량을 최적화한 압축 변형 합성곱(squeezed deformable convolution)을 통해 모듈(module)을 구성하고, 기하학적 변화에 강건하며 자원 사용량을 최적화한 변형 가능한 공간 피라미드 풀링 모듈(DSPP: Deformable Spatial Pyramid Pooling 모듈)을 제안하며, 확장된 변형 가능한 공간 피라미드 풀링 모듈을 제안하여, 도로 영상 의미 분할을 수행하는 시스템 및 방법을 제공하는 데 그 목적이 있다.

과제의 해결 수단

- [0008] 본 발명의 실시예에 따른 영상 의미 분할 시스템은 영상 데이터를 수신하는 입력부와, 영상 데이터에 대한 의미 분할을 수행하는 프로그램이 저장된 메모리 및 프로그램을 실행시키는 프로세서를 포함하고, 프로세서는 입력되는 상기 영상 데이터에 대한 합성곱 연산에 있어, 채널 수를 압축하는 압축된 변형 합성곱 연산을 수행한다.
- [0009] 프로세서는 오프셋을 구하는 합성곱의 출력 채널 수의 증가를 고려하여, 커널의 크기만큼 상기 채널 수를 압축하여 상기 압축된 변형 합성곱 연산을 수행한다.
- [0010] 프로세서는 채널 수 압축을 통해 임시 특징 맵의 크기 증가를 상쇄시킴으로써 샘플링과 재배치 과정에서 연산량을 감소시키고, 재배치된 데이터를 사용하여 상기 출력 채널 수를 생성하여 정보 손실을 방지한다.
- [0011] 프로세서는 추론 과정에서 정수형 오프셋을 사용하여 메모리 사용량을 감소시킨다.
- [0012] 본 발명의 실시예에 따른 영상 의미 분할 방법은 (a) 영상 데이터를 수신하고, 채널 수를 압축하는 압축된 변형 합성곱 연산을 수행하는 단계 및 (b) 추론 과정에서 정수형 오프셋을 사용하는 단계를 포함한다.
- [0013] (a) 단계는 오프셋을 구하는 합성곱의 출력 채널 수의 증가를 고려하여, 커널 크기만큼 상기 채널 수를 압축한다.
- [0014] (a) 단계는 상기 채널 수의 압축을 통해 임시 특징 맵의 크기 증가를 상쇄시키고, 재배치된 데이터를 사용하여 상기 출력 채널 수를 생성한다.
- [0015] (b) 단계는 메모리 사용량에 대한 추가 감소가 필요한 것으로 판단되면, 추론 과정에서 선형 보간법을 사용하지 않고 상기 정수형 오프셋을 사용하여 주변 데이터 추가 사용에 따른 메모리 증가 발생을 방지한다.
- [0017] 본 발명의 다른 실시예에 따른 영상 의미 분할 시스템은 영상 데이터를 수신하는 입력부와, 영상 데이터에 대한 의미 분할을 수행하는 프로그램이 저장된 메모리 및 프로그램을 실행시키는 프로세서를 포함하고, 프로세서는 영상 데이터에 대한 입력 채널 수를 압축하는 압축된 변형 합성곱 연산을 제1 경로 및 제2 경로에 따라 병렬로 수행하되, 제2 경로에 대해서는 풀링하여 입력시키는 것을 특징으로 한다.
- [0018] 압축된 변형 합성곱은 출력 채널 수의 증가를 고려하여 커널 크기만큼 상기 입력 채널 수를 압축한다.
- [0019] 프로세서는 추론 과정에서 정수형 오프셋을 사용하여 메모리 사용량을 감소시킨다.
- [0020] 프로세서는 상기 제1 경로 및 제2 경로에 따른 오프셋 추정 결과의 유효도를 고려하여 의미 분할에 반영한다.
- [0021] 제1 경로 및 제2 경로에서 오프셋을 구하는 합성곱과 특징을 처리하는 합성곱은 각각 공유되는 커널을 사용한다.
- [0022] 본 발명의 다른 실시예에 따른 영상 의미 분할 방법은 (a) 영상 데이터를 수신하고, 입력 채널 수를 압축하는 압축된 변형 합성곱 연산을 제1 경로에 따라 수행하는 단계 및 (b) 제1 경로와 병렬 구조인 제2 경로에서, 입력에 대한 풀링 후 상기 압축된 변형 합성곱 연산을 수행하는 단계를 포함한다.
- [0023] (a) 및 (b) 단계에서의 압축된 변형 합성곱 연산은 출력 채널 수 증가를 고려하여 커널 크기만큼 상기 입력 채널 수를 압축한다.
- [0024] (a) 및 (b) 단계에서의 압축된 변형 합성곱 연산은 추론 과정에서 정수형 오프셋을 사용하여 메모리 사용량을 감소시킨다.
- [0025] (a) 단계가 수행되는 제1 경로 및 (b) 단계가 수행되는 제2 경로에서, 오프셋을 구하는 합성곱과 특징을 처리하는 합성곱은 각각 공유되는 커널을 사용한다.

- [0027] 본 발명의 또 다른 실시예에 따른 영상 의미 분할 시스템은 영상 데이터를 수신하는 입력부와, 영상 데이터에 대한 의미 분할을 수행하는 프로그램이 저장된 메모리 및 프로그램을 실행시키는 프로세서를 포함하고, 프로세서는 제1 output stride에서 입력 채널 수를 압축하는 압축된 변형 합성곱 연산을 제1 경로 및 풀링 후 입력하는 제2 경로에 따라 병렬로 수행하고, 제1 output stride에 적용된 모듈을 직렬로 확장하여 제2 output stride에 적용하되, 제2 output stride에서는 제1 output stride 대비 풀링 후 입력하는 병렬 경로를 추가적으로 구성하여 압축된 변형 합성곱 연산을 수행한다.
- [0028] 압축된 변형 합성곱은 출력 채널 수의 증가를 고려하여 커널 크기만큼 상기 입력 채널 수를 압축한다.
- [0029] 프로세서는 추론 과정에서 정수형 오프셋을 사용하여 메모리 사용량을 감소시킨다.
- [0030] 제1 output stride 및 제2 output stride 적용 시, 오프셋을 구하는 합성곱과 특징을 처리하는 합성곱은 각각 공유되는 커널을 사용한다.
- [0031] 프로세서는 제1 output stride 및 제2 output stride에 적용되는 변형 가능한 공간 피라미드 풀링 모듈을 직렬로 중복 확장하여 연산을 수행한다.
- [0032] 본 발명의 또 다른 실시예에 따른 영상 의미 분할 방법은 (a) 제1 output stride에 적용되는 변형 가능한 공간 피라미드 풀링 모듈을 이용하여, 입력 채널 수를 압축하는 압축된 변형 합성곱 연산을 병렬 경로에 따라 수행하는 단계 및 (b) 상기 제1 output stride의 지수와 상이한 제2 output stride에 적용되는 변형 가능한 공간 피라미드 풀링 모듈을 직렬로 확장하여, 상기 압축된 변형 합성곱 연산을 병렬 경로에 따라 수행하는 단계를 포함한다.
- [0033] 상기 (a) 및 (b) 단계에서 수행되는 압축된 변형 합성곱 연산은 출력 채널 수 증가를 고려하여 커널 크기만큼 상기 입력 채널 수를 압축한다.
- [0034] 상기 (a) 및 (b) 단계에 수행되는 압축된 변형 합성곱 연산은 추론 과정에서 정수형 오프셋을 사용하여 메모리 사용량을 감소시킨다.
- [0035] 상기 (a) 및 (b) 단계의 병렬 경로에서, 오프셋을 구하는 합성곱과 특징을 처리하는 합성곱은 각각 공유되는 커널을 사용한다.
- [0036] 상기 (b) 단계는 상기 (a) 단계에서의 병렬 경로 대비, 풀링 후 입력하는 병렬 경로를 추가 구성하여 압축된 변형 합성곱 연산을 수행한다.
- [0037] 상기 (a) 및 (b) 단계는 각각의 변형 가능한 공간 피라미드 풀링 모듈을 직렬로 중복 확장하여, 상기 압축된 변형 합성곱 연산을 중복 수행한다.

발명의 효과

- [0039] 본 발명에 따르면, 자원 사용량을 최적화한 압축된 변형 합성곱을 제안하여, 모듈 구성이 가능하고, 입력 채널 압축 방법을 사용하여 메모리 증가 원인을 상쇄시키고 연산량을 최적화하는 효과가 있다.
- [0040] 본 발명에 따르면, 레이어를 병렬 배치하고 입력 크기를 변화시키는 변형 가능한 공간 피라미드 풀링 모듈, 확장된 변형 가능한 공간 피라미드 풀링 모듈을 이용하여, 커널 웨이트(kernel weigh)를 공유하여 표본 추출 과정에 제약을 추가하고, 유효 학습 샘플 수를 증가시키며, 인식 성능을 향상시키는 것이 가능한 효과가 있다.
- [0041] 본 발명에 따르면, 최적의 자원 사용량으로 객체의 기하학적 변환을 수용하는 것이 가능하므로, 전방 카메라, SVM 카메라, BSD 카메라, 전측방 카메라 등 모든 딥러닝 기반 도로 영상 인식 시스템에 적용되어 인식 성능을 향상시키고, 자율주행 시스템의 성능과 신뢰도를 크게 향상시키는 효과가 있다.
- [0042] 본 발명의 효과는 이상에서 언급한 것들에 한정되지 않으며, 언급되지 아니한 다른 효과들은 아래의 기재로부터 당업자에게 명확하게 이해될 수 있을 것이다.

도면의 간단한 설명

- [0044] 도 1은 격자 구역별 상이한 특성을 갖는 도로 영상을 도시한다.
- 도 2는 종래 기술에 따른 변형 합성곱의 메모리 증가 원인을 도시한다.
- 도 3은 본 발명의 실시예에 따른 영상 의미 분할 시스템을 도시한다.
- 도 4는 본 발명의 실시예에 따른 압축 변형 합성곱의 구조를 도시한다.
- 도 5는 본 발명의 다른 실시예에 따른 변형 가능한 공간 피라미드 풀링 모듈 구조를 도시한다.
- 도 6은 본 발명의 또 다른 실시예에 따른 확장된 변형 가능한 공간 피라미드 풀링 모듈 구조를 도시한다.
- 도 7은 본 발명의 또 다른 실시예에 따른 직렬 중복 확장 구조를 도시한다.
- 도 8은 본 발명의 실시예에 따른 메모리 사용량 실험 결과를 도시한다.
- 도 9는 본 발명의 실시예에 따른 정수 오프셋(offset) 영향성을 도시한다.
- 도 10은 본 발명의 실시예에 따른 ESPNet V2 기반 Cityscapes 검증 집합 인식 성능을 도시한다.
- 도 11은 본 발명의 실시예에 따른 DeepLab V3+(ResNet-50) 기반 Cityscapes 검증 집합 인식 성능을 도시한다.
- 도 12는 본 발명의 실시예에 따른 ESPNet V2 기반 Camvid 평가 집합 인식 성능을 도시한다.
- 도 13은 본 발명의 실시예에 따른 ESPNet V2 기반 Cityscapes 검증 집합 인식 결과, DeepLab V3+(ResNet-50) 기반 Cityscapes 검증 집합 인식 결과 및 ESPNet V2 기반 Camvid 평가 집합 인식 결과를 도시한다.

발명을 실시하기 위한 구체적인 내용

- [0045] 본 발명의 기술한 목적 및 그 이외의 목적과 이점 및 특징, 그리고 그것들을 달성하는 방법은 첨부되는 도면과 함께 상세하게 후술되어 있는 실시예들을 참조하면 명확해질 것이다.
- [0046] 그러나 본 발명은 이하에서 개시되는 실시예들에 한정되는 것이 아니라 서로 다른 다양한 형태로 구현될 수 있으며, 단지 이하의 실시예들은 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자에게 발명의 목적, 구성 및 효과를 용이하게 알려주기 위해 제공되는 것일 뿐으로서, 본 발명의 권리범위는 청구항의 기재에 의해 정의된다.
- [0047] 한편, 본 명세서에서 사용된 용어는 실시예들을 설명하기 위한 것이며 본 발명을 제한하고자 하는 것은 아니다. 본 명세서에서, 단수형은 문구에서 특별히 언급하지 않는 한 복수형도 포함한다. 명세서에서 사용되는 "포함한다(comprises)" 및/또는 "포함하는(comprising)"은 언급된 구성요소, 단계, 동작 및/또는 소자가 하나 이상의 다른 구성요소, 단계, 동작 및/또는 소자의 존재 또는 추가됨을 배제하지 않는다.
- [0049] 이하에서는, 당업자의 이해를 돕기 위하여 본 발명이 제안된 배경에 대하여 먼저 서술하고, 본 발명의 실시예에 대하여 서술하기로 한다.
- [0050] 자율 주행 시스템은 차량, 보행자 그리고 움직이는 물체 등의 장애물을 인식하여 차량의 가속도를 제어하며, 도로, 차선, 신호등, 표지판과 같은 주행 환경을 인식하여 차량의 주행 방향을 결정한다.
- [0051] 차량에 사용되는 RADAR와 LiDAR는 대기 중에 초고주파나 레이저를 송신한 후 장애물에서 반사된 신호를 수신 처리하므로, 높은 장애물 인식 성능을 제공하지만, 도로 경계나 차선 분류 등의 복잡한 환경 정보를 제공할 수 없다는 한계가 있다.
- [0052] 전방 카메라는 시각을 사용하여 주변을 인식하는 사람과 같이 영상을 처리하여 주행 환경 정보를 제공할 수 있으므로 반드시 필요한 센서이다.
- [0053] 특히 최근 딥러닝의 발전으로 영상 인식 성능이 크게 향상되면서, 높은 신뢰성이 필요한 차량에 object detection, semantic segmentation, panoptic segmentation 그리고 수백 개의 class를 갖는 표지판 구분을 위하여 classification network가 사용되고 있다.
- [0054] 이 중에서 의미 분할(semantic segmentation) 방법은 도로 영상에서 차선, 자유 공간 등 자율 주행시스템에 반드시 필요한 정보를 제공한다.

- [0055] 종래 기술에 따르면, 전방 카메라 영상에서 객체의 모습은 위치에 따라 다르게 나타난다.
- [0056] 일반적으로 차량의 뒷면은 영상 중앙의 자차선에서 나타나며, 근접 옆 차선에서 옆면이 나타난다.
- [0057] 그리고 영상 하단의 객체의 크기가 크고, 소실점으로 갈수록 작아진다.
- [0058] 종래 기술에 따르면, 딥러닝 네트워크에 사용되는 기본적인 layer는 모든 위치에 대하여 동일한 연산을 수행하므로, translation에 강건하나 viewpoint나 scale 등의 변화에 취약하다.
- [0059] 이러한 문제점을 해결하기 위해, 영상 위치 별로 다르게 나타나는 기하학적인 변화를 수용하기 위하여 변형 합성곱(deformable convolution)이 제안되었다.
- [0060] 이 방법은 객체의 기하학적인 변화에 맞춰서 특징을 연산하기 위한 표본 위치를 다르게 사용하므로 변화에 강건하다.
- [0061] 그러나, 이 layer는 각 위치 별로 커널(kernel)이 참조하는 표본을 구해야 하므로 중간 특징 맵(feature map)의 크기가 커널 크기만큼 증가하게 되며, 또한 참조하는 floating point로 표현되는 표본을 구하기 위하여 bilinear interpolation을 사용하므로 중간 특징 맵의 크기가 4배 더 커지게 된다.
- [0062] 만약 3x3 convolution을 수행할 경우, 특징 처리를 위하여 36배나 더 많은 GPU 메모리를 사용하게 되므로, 변형 합성곱(deformable convolution)은 모듈로 구성되지 않고 주로 단독으로 사용되는 것이 일반적이다.
- [0063] 즉, 전술한 바와 같이 종래 기술에 따른 변형 합성곱을 사용하여, 자율 주행 시스템의 전방 카메라를 이용하여 획득된 영상에서 나타나는 객체의 기하학적인 변화를 다루는 것이 가능하나, 중간 특징 맵의 크기가 증가하며 정확한 위치 데이터를 구하기 위한 interpolation 사용에 따라 GPU 메모리 사용량이 증가되어 모듈 구성이 어려운 문제점이 있다.
- [0065] 본 발명은 전술한 문제점을 해결하기 위해 제안된 것으로, 기하학적인 변화에 강건하면서도 자원 사용량을 최적화한 압축 변형 합성곱(squeezed deformable convolution)을 제안한다.
- [0066] 본 발명의 실시예에 따른 압축 변형 합성곱은 기존과 동일하게 데이터 위치를 추정하지만, 특징 처리 시 채널을 압축함으로써 표본 추출 과정에서 발생하는 메모리 증가량을 상쇄시키고, 연산량을 최적화하는 것이 가능하며, 에지 네트워크(edge network)를 위하여 추론(inference) 과정에서 정수 offset을 사용하여 interpolation 과정을 제거함으로써, 추가적인 메모리 사용량을 줄이는 것이 가능하다.
- [0067] 본 발명의 다른 실시예에 따르면, 기하학적인 변화에 강건하면서도 자원 사용량을 최적화한 변형 가능한 공간 피라미드 풀링 모듈(deformable spatial pyramid pooling (DSPP) module)을 제안하며, 압축된 변형 합성곱을 병렬로 배치한 후 입력을 풀링하여 공급함으로써, scale invariant 특성을 강화한다.
- [0068] 본 발명의 다른 실시예에 따르면, Squeeze layer를 제외한 모든 커널 웨이트(kernel weight)를 공유함으로써, 표본 추출 과정에 제약을 추가하고 학습에 사용되는 유효 샘플 수를 증가시킨다.
- [0069] 본 발명의 또 다른 실시예에 따르면, 기하학적인 변화에 강건하면서도 자원 사용량을 최적화한 확장된 변형 가능한 공간 피라미드 풀링 모듈을 제안하며, 이는 디코더의 기능 향상을 위해 변형 가능한 공간 피라미드 풀링 모듈을 output stride 16뿐만 아니라 8에도 확장하여 인식 성능을 강화하며, 각 DSPP 모듈을 직렬로 배치하여 높은 복잡도의 base network 대비 성능을 향상시키는 효과가 있다.
- [0070] 본 발명의 실시예에 따른 방법을 대표적인 semantic segmentation network인 ESPNet V2와 ResNet-50을 baes network로 사용한 DeepLab 3+에 적용하여 효과를 검증하였으며, 종래 기술에 따른 변형 합성곱 대비 0.55%의 메모리만 사용하면서 동등한 인식 성능을 제공하는 것이 가능하다.
- [0071] 본 발명의 실시예에 따르면, 동등한 인식 성능을 확보하면서도 메모리 및 연산량 사용량을 최적화하는 것이 가능하다.
- [0073] 도 1의 (a)는 격자 구역으로 나눈 전방 카메라 영상을 도시하고, 도 1의 (b)는 DeepLab V3+ 특징을 사용한 TSNE 결과를 도시한다.

- [0074] 차량용 전방 카메라 영상은 영역에 따라 보여지는 객체가 다르다.
- [0075] 영상의 유사도를 TSNE를 사용하여 분석해보면, 도 1의 (b)에 도시한 바와 같이 상, 중, 하로 특징을 나눌 수 있고, 왼쪽과 오른쪽은 유사하나 중간은 쉽게 구분할 수 있다.
- [0076] 따라서 변형 합성곱(deformable convolution)을 사용하면 위치에 따라 다르게 나타나는 객체를 처리할 수 있으므로 인식 성능 향상에 유리한 면이 있다.
- [0077] 변형 합성곱은 학습과 추론 과정에서 많은 GPU 메모리를 사용하기 때문에, 입력 영상의 크기를 줄여서 다수를 사용해야 하지만, 자율 주행 시스템은 객체의 고정밀 위치 해상도가 필요하기 때문에 입력 영상의 크기를 줄일 수 없다.
- [0078] 이하, 도 2를 참조하여 종래 기술에 따른 변형 합성곱의 메모리 증가 원인을 설명한다.
- [0079] 일반적인 convolution은 한 위치의 출력을 연산하기 위하여 주변 격자 위치의 데이터를 사용한다.
- [0080] 따라서 한 출력 연산에 사용된 데이터 중 일부는 주변 데이터를 처리할 때 재사용 되므로, 연산 중에 임시 특징 맵(feature map)의 크기가 증가하지 않는다.
- [0081] 변형 합성곱(Deformable convolution)은 입력에 따라 사용하는 데이터의 위치를 변경하기 위하여 convolution을 사용하여 좌표 offset을 먼저 구한다.
- [0082] 한 출력 위치의 convolution에 사용되는 데이터는 기준점으로부터 offset 만큼 변위를 갖게 되므로, 임의의 위치에 분포하게 된다.
- [0083] 따라서 출력 위치 별로 연산에 사용되는 데이터가 다를 수 있으므로, effective feature map의 크기는 kernel 크기만큼 증가하게 된다.
- [0084] 따라서 3 x 3 convolution을 사용할 경우, 도 2의 (a)에 도시한 바와 같이, 임시 feature map의 크기가 9배 증가한다.
- [0085] 변형 합성곱(Deformable convolution) 과정 중에 구한 좌표 offset은 임의의 실수 값을 가지고, 변경된 위치의 데이터를 구하기 위하여 bilinear interpolation을 사용한다.
- [0086] Bilinear interpolation은 변경된 좌표를 감싸는 고정된 4개의 데이터를 선형 보간함으로써 목표 위치의 데이터를 추정할 수 있다.
- [0087] 즉, kernel 한 위치의 데이터를 추정하기 위하여 4개의 데이터를 추가로 사용하게 되므로, 임시 특징 맵(feature map)의 크기가 도 2의 (b)에 도시한 바와 같이 4배 증가하게 된다.
- [0088] 이하에서는 본 발명의 실시예에 대한 당업자의 이해를 돕기 위하여, 3 x 3 convolution 일 때 변형 합성곱의 메모리 사용량을 분석하고, 메모리 증가 원인을 상쇄시키는 압축 변형 합성곱(squeezed deformable convolution)을 제안한다.
- [0089] 본 발명의 다른 실시예에 따르면 고효율 메모리 구조를 가진 압축 변형 합성곱을 사용하여 high-end context를 처리할 수 있는 DSPP 모듈을 제안하고, 본 발명의 또다른 실시예에 따르면 DSPP 모듈을 효과적으로 확장 적용하는 구성에 대해 제안한다.
- [0091] 도 3은 본 발명의 실시예에 따른 영상 의미 분할 시스템을 도시한다.
- [0092] 본 발명의 실시예에 따른 영상 의미 분할 시스템은 영상 데이터를 수신하는 입력부(110)와, 영상 데이터에 대한 의미 분할을 수행하는 프로그램이 저장된 메모리(120) 및 프로그램을 실행시키는 프로세서(130)를 포함하고, 프로세서(130)는 입력되는 영상 데이터에 대한 합성곱 연산에 있어, 채널 수를 압축하는 압축된 변형 합성곱 연산을 수행하며, 이러한 본 발명의 실시예에 대하여는, 이하 도 4를 참조하여 상세히 설명한다.
- [0093] 프로세서(130)는 오프셋을 구하는 합성곱의 출력 채널 수의 증가를 고려하여, 커널의 크기만큼 채널 수를 압축하여 압축된 변형 합성곱 연산을 수행한다.
- [0094] 프로세서(130)는 채널 수 압축을 통해 임시 특징 맵의 크기 증가를 상쇄시킴으로써 샘플링과 재배치 과정에서 연산량을 감소시키고, 재배치된 데이터를 사용하여 상기 출력 채널 수를 생성하여 정보 손실을 방지한다.

- [0095] 프로세서(130)는 추론 과정에서 정수형 오프셋을 사용하여 메모리 사용량을 감소시킨다.
- [0097] 본 발명의 다른 실시예에 따른 영상 의미 분할 시스템은 영상 데이터를 수신하는 입력부(110)와, 영상 데이터에 대한 의미 분할을 수행하는 프로그램이 저장된 메모리(120) 및 프로그램을 실행시키는 프로세서(130)를 포함하고, 프로세서(130)는 영상 데이터에 대한 입력 채널 수를 압축하는 압축된 변형 합성곱 연산을 제1 경로 및 제2 경로에 따라 병렬로 수행하되, 제2 경로에 대해서는 풀링하여 입력시키는 것을 특징으로 하며, 이러한 본 발명의 다른 실시예에 대하여는, 이하 도 5를 참조하여 상세히 설명한다.
- [0098] 압축된 변형 합성곱은 출력 채널 수의 증가를 고려하여 커널 크기만큼 상기 입력 채널 수를 압축한다.
- [0099] 프로세서(130)는 추론 과정에서 정수형 오프셋을 사용하여 메모리 사용량을 감소시킨다.
- [0100] 프로세서(130)는 상기 제1 경로 및 제2 경로에 따른 오프셋 추정 결과의 유효도를 고려하여 의미 분할에 반영한다.
- [0101] 제1 경로 및 제2 경로에서 오프셋을 구하는 합성곱과 특징을 처리하는 합성곱은 각각 공유되는 커널을 사용한다.
- [0103] 본 발명의 또 다른 실시예에 따른 영상 의미 분할 시스템은 영상 데이터를 수신하는 입력부(110)와, 영상 데이터에 대한 의미 분할을 수행하는 프로그램이 저장된 메모리(120) 및 프로그램을 실행시키는 프로세서(130)를 포함하고, 프로세서(130)는 제1 output stride에서 입력 채널 수를 압축하는 압축된 변형 합성곱 연산을 제1 경로 및 풀링 후 입력하는 제2 경로에 따라 병렬로 수행하고, 제1 output stride에 적용된 모듈을 직렬로 확장하여 제2 output stride에 적용하되, 제2 output stride에서는 제1 output stride 대비 풀링 후 입력하는 병렬 경로를 추가적으로 구성하여 압축된 변형 합성곱 연산을 수행하며, 이러한 본 발명의 또 다른 실시예에 대하여는, 이하 도 6 및 도 7을 참조하여 상세히 설명한다.
- [0105] 압축된 변형 합성곱은 출력 채널 수의 증가를 고려하여 커널 크기만큼 상기 입력 채널 수를 압축한다.
- [0106] 프로세서(130)는 추론 과정에서 정수형 오프셋을 사용하여 메모리 사용량을 감소시킨다.
- [0107] 제1 output stride 및 제2 output stride 적용 시, 오프셋을 구하는 합성곱과 특징을 처리하는 합성곱은 각각 공유되는 커널을 사용한다.
- [0108] 프로세서(130)는 제1 output stride 및 제2 output stride에 적용되는 변형 가능한 공간 피라미드 풀링 모듈을 직렬로 중복 확장하여 연산을 수행한다.
- [0110] 도 4는 본 발명의 실시예에 따른 압축 변형 합성곱의 구조를 도시한다.
- [0111] 본 발명의 실시예에 따르면, 변형 합성곱의 메모리 사용량을 감소시키기 위해 압축 변형 합성곱을 제안한다.
- [0112] 본 발명의 실시예에 따르면, 채널 수를 감소시킨 후 데이터 위치를 변경한다.
- [0113] 변형 합성곱은 출력 위치 별로 다른 오프셋을 구하지만, 입력 채널 별로 다른 오프셋 값을 사용하지 않는다. 만약, 입력 채널 별로 오프셋을 다르게 구하려면, 오프셋을 구하는 합성곱의 출력 채널 수가 입력 채널 수의 2×3^2 배로 증가해야 한다. 따라서, 단일 채널에 적용되는 오프셋을 구한 후 모든 채널로 확장한다. Xception의 가설과 같이 cross-channels correlation과 spatial correlation을 분리할 수 있다면, 채널 연산 보다는 data reallocation 특성이 변형 합성곱의 네트워크 표현력 향상에 기여하는 특성이 된다.
- [0114] 따라서 ResNet의 bottleneck과 같이 spatial convolution 전에 채널 압축의 결과로 얻은 자원이 잃어버린 정보량 보다 많을 경우, 결과적으로는 네트워크 표현력을 효율적으로 향상시키는 것이 가능하다.
- [0115] 본 발명의 실시예에 따르면, 도 4에 도시한 바와 같이 압축 변형 합성곱을 이용하여 커널 크기만큼 채널 수를 압축하여 임의 특징 맵 크기 증가를 상쇄시킨다.
- [0116] sampling과 relocation 과정에서 일반적인 convolution과 동일한 특징 맵 크기를 사용하며, 특징 맵 크기 증가

가 상쇄됨에 따라 연산량도 감소하게 된다.

- [0117] 또한, 재배치된 데이터를 사용하여 기존과 동일한 출력 채널 수를 생성함으로써 정보 손실을 최소화하는 것이 가능하다.
- [0118] 본 발명의 실시예에 따르면, 에지 네트워크와 같이 최소 자원이 필요한 분야를 위해 정수형 오프셋을 사용한다.
- [0119] 오프셋 값은 0으로 시작하고, 학습이 진행되면서 바뀌게 된다.
- [0120] Bilinear interpolation에 의하여 매 반복(iteration)마다 선형적으로 변화하던 오프셋 영향성은 정수 값을 지날 때 사용하는 데이터가 변경되면서 비선형적인 특성을 가지게 된다.
- [0121] 따라서 정수형 오프셋 값을 사용할 경우, 매 반복마다 참조하는 데이터가 변할 정도의 출력이 없으면, 오프셋 연산 경로에 그래디언트(gradient)가 전달되지 않으므로 학습이 되지 않는데, 추론(inference) 과정에서는 그래디언트의 전달이 필요하지 않다.
- [0122] 본 발명의 실시예에 따르면, 메모리 사용량 감소가 필요할 경우, 추론 과정에서만 정수형 오프셋을 사용하여 메모리 사용량을 감소시킨다.
- [0123] 추론 과정에서 bilinear interpolation을 사용하지 않음에 따른 위치 추정 오차의 증가가 발생하지만, 주변 데이터 추가 사용에 따른 메모리 증가가 발생되지 않는다.
- [0125] 도 5는 본 발명의 다른 실시예에 따른 변형 가능한 공간 피라미드 풀링 모듈 구조를 도시한다.
- [0126] 본 발명의 실시예에 따른 압축된 변형 합성곱은 입력 채널 수를 압축함에 따라 정보량 손실이 발생되고, 정수형 오프셋을 추론 과정에서 사용함에 따라 오차가 추가되지만, 메모리 사용량을 크게 줄이는 것이 가능하다.
- [0127] 본 발명의 다른 실시예에 따르면, 입력 특징(feature)의 크기가 큰 경우에도 high-end context를 처리할 수 있는 모듈을 구성함으로써, 단점을 보완하는 것이 가능하다.
- [0128] 본 발명의 다른 실시예에 따르면, 보다 풍부한 컨텍스트 추출을 위해, 변형 가능한 공간 피라미드 풀링(deformable spatial pyramid pooling, DSPP) 모듈을 제안한다.
- [0129] 변형 합성곱은 입출력이 공간적으로 동일한 정보를 가지고 있지 않으므로, skip connection을 사용하는 직렬 구조의 모듈은 그 효율이 좋지 않은 문제점이 있는 바, 본 발명의 다른 실시예에 따르면 도 5에 도시한 바와 같이, 압축된 변형 합성곱을 병렬로 배치하고, 입력을 풀링하여 병렬 경로에 입력한다.
- [0130] 입력에 대해 압축된 변형 합성곱 연산이 수행되고(S503), 입력에 대한 에버리지 풀링(S501)을 거쳐 압축된 변형 합성곱 연산이 수행된다(S502).
- [0131] S502 단계에 후속하여 업샘플링이 수행되고(S504), 이어서 Concatenate(S505) 과정 및 컨볼루션(S50) 이후 출력이 수행된다.
- [0132] SPP나 ASPP는 다양한 크기의 객체에 맞는 effective FOV를 제공하기 위해 병렬로 구성하지만, 변형 합성곱은 단일 layer만으로도 객체에 맞는 FOV 제공이 가능하므로 병렬 구조는 불필요한 중복이 될 수 있다.
- [0133] 본 발명의 다른 실시예에 따르면, KSAC와 같이 학습에 사용되는 유효 샘플 수를 증가시키기 위한 목적과, 오프셋 학습을 가이드 하기 위한 목적에서 병렬 구조가 사용된다.
- [0134] KSAC와 같이 학습에 사용되는 유효 샘플 수를 증가시키기 위한 목적에 대해 설명하면, 변형 합성곱은 학습 초기에 위치 정확도가 높지 않으므로, 오프셋 학습에 영향을 주는 유효 샘플 수가 적은 반면, 본 발명의 다른 실시예에 따르면 크기가 다른 입력을 병렬로 사용함으로써, 오프셋 추정이 부정확하더라도 다양한 크기의 객체에 대응하는 것이 가능하다.
- [0135] 오프셋 학습을 가이드 하기 위한 목적에 대해 설명하면, 풀링(Pooling)이 사용된 경로에는 크기가 반으로 줄어든 객체가 입력되므로, 정상적으로 오프셋 학습이 수행되었다면 오프셋 값도 반으로 줄어들어야 한다.
- [0136] 만약 두 경로의 오프셋 추정이 다르다면, 유효도가 더 높은 경로의 결과가 해당 iteration에서 더 반영된다.
- [0137] 병렬 경로에 입력되는 객체 크기 상관 관계로부터 서로 오프셋 추정 과정을 도와줄 수 있으며, 특정 경로에서 학습된 결과를 공유하기 위하여, 도 5에 도시한 바와 같이 오프셋을 구하는 합성곱과 특징을 처리하는 합성곱은

공유되는 커널(shared kernel)을 사용한다.

- [0138] 공간적 처리를 하지 않는 압축된 변형 합성곱은 독립적으로 사용되어, 네트워크 표현력을 향상시키는 것이 가능하다.
- [0140] 도 6은 본 발명의 또 다른 실시예에 따른 확장된 변형 가능한 공간 피라미드 풀링 모듈 구조를 도시하고, 도 7은 본 발명의 또 다른 실시예에 따른 직렬 중복 확장 구조를 도시한다.
- [0141] 종래 기술에 따른 일반적인 semantic segmentation network는 ILSVRC2012 등에서 학습된 높은 복잡도의 분류 네트워크(classification network)를 엔코더(encoder)로 사용하므로, 전체 복잡도를 낮추기 위하여 단순한 디코더(decoder)를 사용하여야 하고, SPP나 ASPP도 특징 맵(feature map) 크기가 작은 output stride 16에 적용된다.
- [0142] 하지만 MMANet과 같이 high-end context를 처리하는 모듈을 output stride 8에도 적용하여 디코더(decoder)를 구성함으로써 인식 성능을 보다 향상시키는 것이 가능하다.
- [0143] 본 발명의 또 다른 실시예에 따르면, 변형 가능한 공간 피라미드 풀링 모듈을 output stride 8까지 적용한 확장된 변형 가능한 공간 피라미드 풀링 모듈(extended DSPP module)을 제안한다.
- [0144] 도 6을 참조하면, output stride 16에 적용된 모듈을 직렬로 확장하여 output stride 8에도 적용하였다.
- [0145] 크기가 1/64 이하인 특징은 추상화 수준이 높아서 데이터 위치를 조정하는 오프셋 학습 가이드 능력이 낮으므로, output stride 8에 적용된 module은 병렬 경로를 더 추가하여 학습 유효 샘플 수를 증가시키고, 오프셋 가이드 능력을 향상시킨다.
- [0146] S601 단계에서는 전술한 바와 같이, 입력에 대해 압축된 변형 합성곱 연산이 수행되고, 입력에 대한 에버리지 풀링을 거쳐 압축된 변형 합성곱 연산이 수행된다.
- [0147] 또한, 업 샘플링 후 Concatenate, 컨볼루션이 수행되고, 그 출력은 S602 단계에서 업 샘플링된다.
- [0148] 이 때, S601 단계에서는 보라색과 녹색으로 표시한 바와 같이, 오프셋을 구하는 합성곱(보라색)과 특징을 처리하는 합성곱(녹색)은 공유되는 커널(shared kernel)을 사용한다.
- [0149] S602 단계 이후, S603 단계는 제1 풀링(average pooling by 2) 및 제2 풀링(average pooling by 4)을 거쳐 각각 압축된 변형 합성곱 연산 과정을 거치고, 업 샘플링 후 Concatenate, 컨볼루션이 수행된다.
- [0150] 또한, 본 발명의 또 다른 실시예에 따르면, 도 7에 도시한 바와 같이, 디코더 강화 효과 확인을 위해 각각 stride에 적용된 변형 가능한 공간 피라미드 풀링 모듈을 직렬로 중복 확장한다.
- [0151] 즉, output stride 16에 적용된 변형 가능한 공간 피라미드 풀링이 중복적으로 수행되고(S601-1, S601-2), 업 샘플링(S602) 후, output stride 8에 적용된 변형 가능한 공간 피라미드 풀링이 중복적으로 수행된다(S603-1, S603-2).
- [0153] 도 8은 본 발명의 실시예에 따른 메모리 사용량 실험 결과를 도시한다.
- [0154] Df ConV는 변형 합성곱을, Sdf Conv는 본 발명의 실시예에 따른 압축된 변형 합성곱을 의미한다.
- [0155] 각 모듈이 소모하는 메모리 양과 연산량을 비교하기 위하여 2개의 1 x 1 convolution을 사용하여 384 채널을 처리하는 모듈에 입력을 공급하고 출력을 처리하였다.
- [0156] 48 x 48 또는 96 x 96의 영상을 사용하여 output stride 16과 8일 때의 각 모듈의 입력 크기를 모사하였다.
- [0157] 첫 번째로 CUDA 10.2, CuDNN 7.6.5 그리고 PyTorch 1.6.0 환경에서 NVIDIA TITAN RTX GPU를 사용하여 메모리 사용량을 측정하였다.
- [0158] 변형 합성곱은 도 8에서 확인할 수 있듯이, 모든 모듈 중에서 가장 많은 GPU 메모리를 사용한다.
- [0159] Output stride가 16일 때에는 일반적인 convolution 보다 메모리 사용량이 2배 정도 많이 소모하나, output stride가 8일 때에는 4배 이상 메모리 사용량이 증가한다.

- [0160] 재배치 과정에서 특징 맵 크기를 증가시킨 후 연산을 수행하므로, 메모리 총 사용량이 특징 맵 크기의 증가만큼 증가하게 되는 것을 확인할 수 있다.
- [0161] 본 발명의 실시예에 따른 압축된 변형 합성곱은 입력 채널을 384에서 42로 압축하므로 일반적인 convolution과 유사한 메모리를 사용하는 것을 확인할 수 있으며, 특히 정수형 오프셋을 사용할 경우, bilinear interpolation을 사용하지 않으므로 output stride가 16일 때에는 1.02배 그리고 output stride가 8일 때에는 1.11배의 메모리만 사용하는 것을 확인할 수 있다.
- [0162] 즉, 본 발명의 실시예에 따르면 객체의 기하학적 변환에 강건하면서 일반적인 convolution대비 동등한 메모리 사용량을 가지게 되는 효과가 있다.
- [0163] 모듈의 메모리 사용량을 비교해 보면, 병렬 경로에서 convolution 연산을 하지 않는 SPP(Spatial Pyramid Pooling)가 최소의 메모리 사용량을 갖는다.
- [0164] 본 발명의 다른 실시예에 따른 변형 가능한 공간 피라미드 풀링(DSPP)은 output stride가 16일 때 두 개의 병렬 경로를 가지므로, 학습과 추론(inference) 시에 모두 ASPP 보다 작은 메모리를 사용한다.
- [0165] 하지만 output stride가 8일 경우에는 3개의 병렬 경로를 사용하므로 ASPP(Atrous Spatial Pyramid Pooling)보다 다수의 메모리를 사용한다.
- [0166] 오프셋 오차에 의한 인식 성능 감소를 감수할 경우, 추론 시 정수 오프셋을 사용하게 되면 ASPP 보다 10% 적은 메모리 사용량이 되도록 구현이 가능하다.
- [0167] 두 번째로 연산을 수행하기 위하여 사용된 곱셈 수를 계산하였다.
- [0168] 본 발명의 실시예에 따른 압축된 변형 합성곱은 전체 모듈 중 최소의 곱셈 수를 사용한다.
- [0169] 이는 입력 채널 수를 압축하여 일반적인 convolution의 20% 이하의 곱셈 수만 사용한다.
- [0170] 그리고 ASPP와 변형 합성곱은 일반적인 convolution과 거의 동등한 연산 수를 사용하는 것을 확인하였다.
- [0171] ASPP는 depthwise separable convolution을 사용하여 병렬 경로 수 증가에 따른 연산량 증가를 상쇄시켰다.
- [0172] 그리고 변형 합성곱은 오프셋을 구하기 위한 convolution의 출력 채널 수가 2×3^2 으로 작기 때문에 곱셈 수 증가량이 작았다.
- [0173] 본 발명의 다른 실시예에 따른 변형 가능한 공간 피라미드 풀링(DSPP)도 모든 모듈 중에서 최소 곱셈 수를 사용한다.
- [0174] 압축된 변형 합성곱이 사용하는 곱셈 수가 매우 작으므로, output stride가 8인 경우에도 SPP와 동일한 연산 수를 갖는다.
- [0175] 이를 통해, 본 발명의 실시예에 따르면 모듈을 구성하는 경우에도 변형 합성곱 대비 메모리 뿐 아니라 연산량도 적게 사용하는 것을 알 수 있다.
- [0177] 도 9는 본 발명의 실시예에 따른 정수 오프셋(offset) 영향성을 도시한다.
- [0178] Integer offset을 사용하는 것은 weight quantization과 유사하여 정수화 하는 과정에서 최대 0.5 샘플의 오차 발생이 가능하다.
- [0179] 따라서, 다수의 압축된 변형 합성곱을 사용하는 경우, 오차 누적에 따라 인식 성능을 더 낮추게 된다.
- [0180] 본 발명의 실시예에 따르면, 정수 오프셋 사용에 따른 인식 성능 하락을 보상하기 위해, fine tuning을 수행하였다.
- [0181] ESPNet V2를 Camvid 데이터셋에 학습시킨 후 정수화 하고 500 epochs 동안 추가 학습을 진행하였다.
- [0182] 정수화를 진행하였을 때 도 9에 도시한 바와 같이, 압축된 변형 합성곱, 변형 가능한 공간 피라미드 풀링 모듈, 확장된 변형 가능한 공간 피라미드 풀링 모듈은 각각 0.5%, 0.7%, 1.4% mIoU가 감소하였으나, fine tuning 후에는 각각 감소량이 0.3%, 0.5, 0.9% mIoU로 변화하였다.
- [0183] 도 10은 본 발명의 실시예에 따른 ESPNet V2 기반 Cityscapes 검증 집합 인식 성능을 도시하고, 도 11은 본 발

명의 실시예에 따른 DeepLab V3+(ResNet-50) 기반 Cityscapes 검증 집합 인식 성능을 도시하고, 도 12는 본 발명의 실시예에 따른 ESPNet V2 기반 Camvid 평가 집합 인식 성능을 도시하며, 도 13은 본 발명의 실시예에 따른 ESPNet V2 기반 Cityscapes 검증 집합 인식 결과, DeepLab V3+(ResNet-50) 기반 Cityscapes 검증 집합 인식 결과 및 ESPNet V2 기반 Camvid 평가 집합 인식 결과를 도시한다.

- [0184] 제안한 방법의 효과를 검증하기 위하여 ESPNet V2에 적용하여 Cityscapes를 사용하여 평가하였다.
- [0185] 본 발명의 실시예에 따른 압축된 변형 합성곱은 입력 채널을 압축할 때 발생하는 정보 손실로 인하여 도 10에 도시한 바와 같이, 변형 합성곱 대비 0.4% mIoU가 낮다.
- [0186] 하지만 절약한 자원을 사용하여 모듈을 구성한 변형 가능한 공간 피라미드 풀링(DSPP)의 경우 1.3% mIoU를 향상시킬 수 있다.
- [0187] 모듈 사용을 확장하여 output stride 8에도 적용한 확장된 변형 가능한 공간 피라미드 풀링의 경우, 원본 네트워크 보다 3.3% mIoU를 향상시킬 수 있다.
- [0188] 디코더를 보강한 효과를 확인하기 위하여 output stride 8과 16 위치에 변형 가능한 공간 피라미드 풀링(DSPP) 두 개를 직렬로 배치한 경우, 원본 ESPNet V2 보다 6.0% mIoU를 향상시켜서 DeepLab 3+ with ResNet-50 보다 높은 인식 성능 제공이 가능함을 확인하였다.
- [0189] 도 13의 (a)는 ESPNet V2의 inference 결과를 나타낸다.
- [0190] 모듈 중에서는 본 발명의 다른 실시예에 따른 변형 가능한 공간 피라미드 풀링(DSPP)가, 모듈 사용을 확장한 경우에는 변형 가능한 공간 피라미드 풀링이 객체 경계선을 가장 잘 인식하는 것을 확인할 수 있다.
- [0191] 본 발명의 실시예에 따른 방법이 다른 네트워크에도 효과가 있는지 확인하기 위하여 도 11과 같이 ResNet-50을 base network로 사용한 DeepLab 3+에도 적용해 보았다.
- [0192] 이 실험에서는 압축된 변형 합성곱이 변형 합성곱 대비 0.1% mIoU 좋은 결과를 보여준다.
- [0193] 전술한 바와 마찬가지로, 변형 가능한 공간 피라미드 풀링(DSPP)이 모듈 중에서 가장 높은 인식 성능을 제공하며, 확장된 변형 가능한 공간 피라미드 풀링(DSPP)을 통해 모듈 사용을 확장한 경우 가장 높은 77.9% mIoU를 제공한다.
- [0194] 이 수치는 base network가 Xception 일 때와 유사한 인식 성능으로 낮은 복잡도의 base network를 사용하여도 효율적으로 context를 추출하여 디코더를 강화하며 인식 성능을 향상시키는 효과를 보여준다.
- [0195] 도 13의 (b)는 DeepLab 3+ with ResNet-50의 inference 결과를 보여준다.
- [0196] ESPNet V2와 마찬가지로 본 발명의 실시예에 따른 방법을 적용한 경우 가장 ground truth와 유사한 추정을 하는 것을 확인할 수 있다.
- [0197] 본 발명의 실시예에 따른 방법이 다른 dataset에도 효과가 있는지 확인하기 위하여, 도 12와 같이 ESPNet V2에 적용하여 Camvid를 사용하여 평가하였다.
- [0198] DeepLab 3+ with ResNet-50에 적용한 결과와 마찬가지로, 본 발명의 실시예에 따른 압축된 변형 합성곱과 변형 가능한 공간 피라미드 풀링(DSPP)이 layer와 모듈 중에서 가장 높은 인식 성능을 제공하는 것을 확인하였다.
- [0199] 확장된 변형 가능한 공간 피라미드 풀링을 적용할 경우, 원본 보다 3.2% 높은 78.4% mIoU를 달성하는 것을 확인하였다.
- [0200] 도 13의 (c)는 ESPNet V2의 inference 결과를 보여준다. 본 발명의 실시예에 따른 방법을 적용하였을 때, 경계선이 명확하게 추정되는 것을 확인할 수 있다.
- [0202] 본 발명의 실시예에 따른 영상 의미 분할 방법은 (a) 영상 데이터를 수신하고, 채널 수를 압축하는 압축된 변형 합성곱 연산을 수행하는 단계 및 (b) 추론 과정에서 정수형 오프셋을 사용하는 단계를 포함한다.
- [0203] (a) 단계는 오프셋을 구하는 합성곱의 출력 채널 수의 증가를 고려하여, 커널 크기만큼 상기 채널 수를 압축한다.
- [0204] (a) 단계는 상기 채널 수의 압축을 통해 임시 특징 맵의 크기 증가를 상쇄시키고, 재배치된 데이터를 사용하여

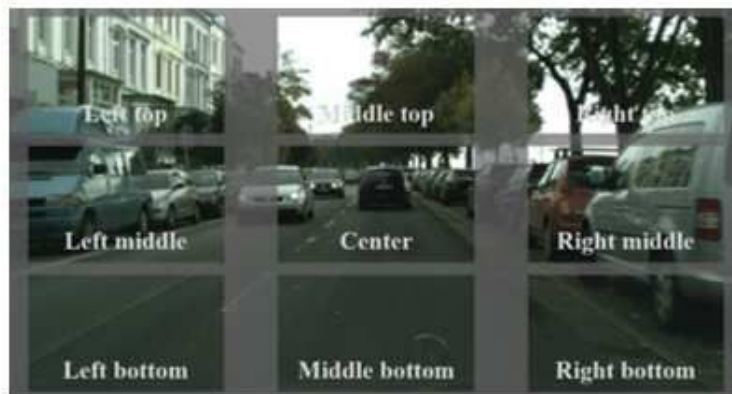
상기 출력 채널 수를 생성한다.

- [0205] (b) 단계는 메모리 사용량에 대한 추가 감소가 필요한 것으로 판단되면, 추론 과정에서 선형 보간법을 사용하지 않고 상기 정수형 오프셋을 사용하여 주변 데이터 추가 사용에 따른 메모리 증가 발생을 방지한다.
- [0207] 본 발명의 다른 실시예에 따른 영상 의미 분할 방법은 (a) 영상 데이터를 수신하고, 입력 채널 수를 압축하는 압축된 변형 합성곱 연산을 제1 경로에 따라 수행하는 단계 및 (b) 제1 경로와 병렬 구조인 제2 경로에서, 입력에 대한 풀링 후 상기 압축된 변형 합성곱 연산을 수행하는 단계를 포함한다.
- [0208] (a) 및 (b) 단계에서의 압축된 변형 합성곱 연산은 출력 채널 수 증가를 고려하여 커널 크기만큼 상기 입력 채널 수를 압축한다.
- [0209] (a) 및 (b) 단계에서의 압축된 변형 합성곱 연산은 추론 과정에서 정수형 오프셋을 사용하여 메모리 사용량을 감소시킨다.
- [0210] (a) 단계가 수행되는 제1 경로 및 (b) 단계가 수행되는 제2 경로에서, 오프셋을 구하는 합성곱과 특징을 처리하는 합성곱은 각각 공유되는 커널을 사용한다.
- [0212] 본 발명의 또 다른 실시예에 따른 영상 의미 분할 방법은 (a) 제1 output stride에 적용되는 변형 가능한 공간 피라미드 풀링 모듈을 이용하여, 입력 채널 수를 압축하는 압축된 변형 합성곱 연산을 병렬 경로에 따라 수행하는 단계 및 (b) 상기 제1 output stride의 지수와 상이한 제2 output stride에 적용되는 변형 가능한 공간 피라미드 풀링 모듈을 직렬로 확장하여, 상기 압축된 변형 합성곱 연산을 병렬 경로에 따라 수행하는 단계를 포함한다.
- [0213] 상기 (a) 및 (b) 단계에서 수행되는 압축된 변형 합성곱 연산은 출력 채널 수 증가를 고려하여 커널 크기만큼 상기 입력 채널 수를 압축한다.
- [0214] 상기 (a) 및 (b) 단계에 수행되는 압축된 변형 합성곱 연산은 추론 과정에서 정수형 오프셋을 사용하여 메모리 사용량을 감소시킨다.
- [0215] 상기 (a) 및 (b) 단계의 병렬 경로에서, 오프셋을 구하는 합성곱과 특징을 처리하는 합성곱은 각각 공유되는 커널을 사용한다.
- [0216] 상기 (b) 단계는 상기 (a) 단계에서의 병렬 경로 대비, 풀링 후 입력하는 병렬 경로를 추가 구성하여 압축된 변형 합성곱 연산을 수행한다.
- [0217] 상기 (a) 및 (b) 단계는 각각의 변형 가능한 공간 피라미드 풀링 모듈을 직렬로 중복 확장하여, 상기 압축된 변형 합성곱 연산을 중복 수행한다.
- [0219] 한편, 본 발명의 실시예에 따른 영상 의미 분할 방법은 컴퓨터 시스템에서 구현되거나, 또는 기록매체에 기록될 수 있다. 컴퓨터 시스템은 적어도 하나 이상의 프로세서와, 메모리와, 사용자 입력 장치와, 데이터 통신 버스와, 사용자 출력 장치와, 저장소를 포함할 수 있다. 전술한 각각의 구성 요소는 데이터 통신 버스를 통해 데이터 통신을 한다.
- [0220] 컴퓨터 시스템은 네트워크에 커플링된 네트워크 인터페이스를 더 포함할 수 있다. 프로세서는 중앙처리 장치 (central processing unit (CPU))이거나, 혹은 메모리 및/또는 저장소에 저장된 명령어를 처리하는 반도체 장치일 수 있다.
- [0221] 메모리 및 저장소는 다양한 형태의 휘발성 혹은 비휘발성 저장매체를 포함할 수 있다. 예컨대, 메모리는 ROM 및 RAM을 포함할 수 있다.
- [0222] 따라서, 본 발명의 실시예에 따른 영상 의미 분할 방법은 컴퓨터에서 실행 가능한 방법으로 구현될 수 있다. 본 발명의 실시예에 따른 영상 의미 분할 방법이 컴퓨터 장치에서 수행될 때, 컴퓨터로 판독 가능한 명령어들이 본 발명에 따른 영상 의미 분할 방법을 수행할 수 있다.
- [0223] 한편, 상술한 본 발명에 따른 영상 의미 분할 방법은 컴퓨터로 읽을 수 있는 기록매체에 컴퓨터가 읽을 수 있는

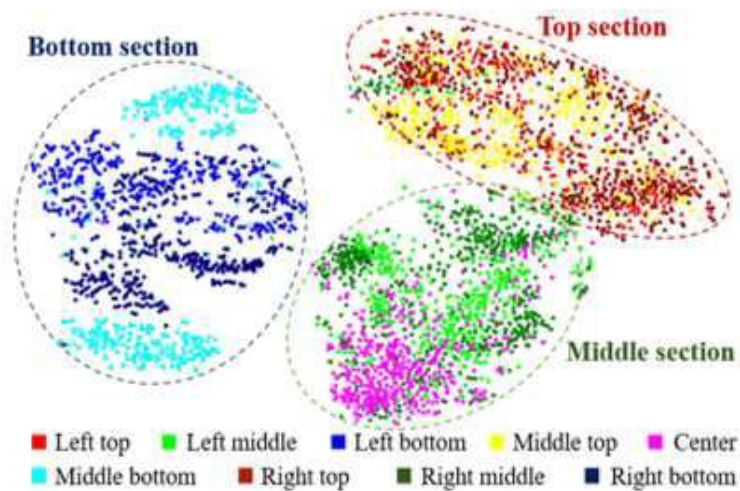
코드로서 구현되는 것이 가능하다. 컴퓨터가 읽을 수 있는 기록 매체로는 컴퓨터 시스템에 의하여 해독될 수 있는 데이터가 저장된 모든 종류의 기록 매체를 포함한다. 예를 들어, ROM(Read Only Memory), RAM(Random Access Memory), 자기 테이프, 자기 디스크, 플래시 메모리, 광 데이터 저장장치 등이 있을 수 있다. 또한, 컴퓨터로 판독 가능한 기록매체는 컴퓨터 통신망으로 연결된 컴퓨터 시스템에 분산되어, 분산방식으로 읽을 수 있는 코드로서 저장되고 실행될 수 있다.

도면

도면1

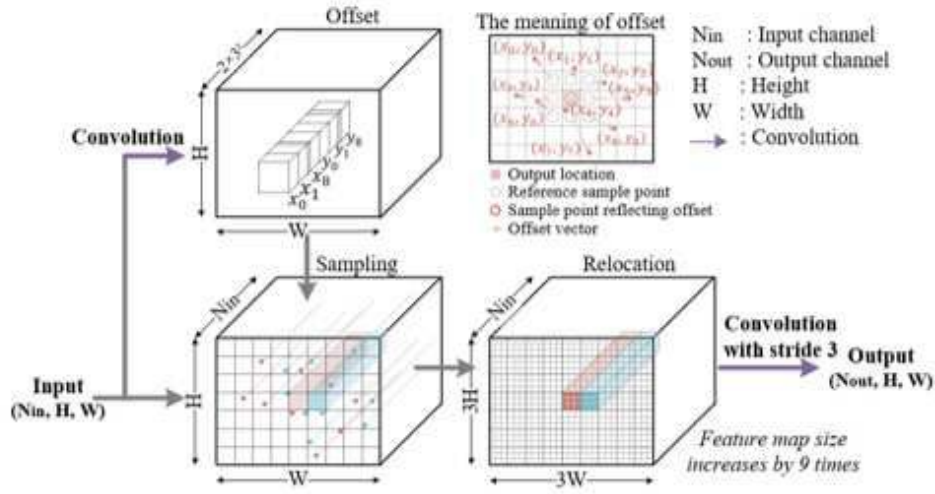


(a)

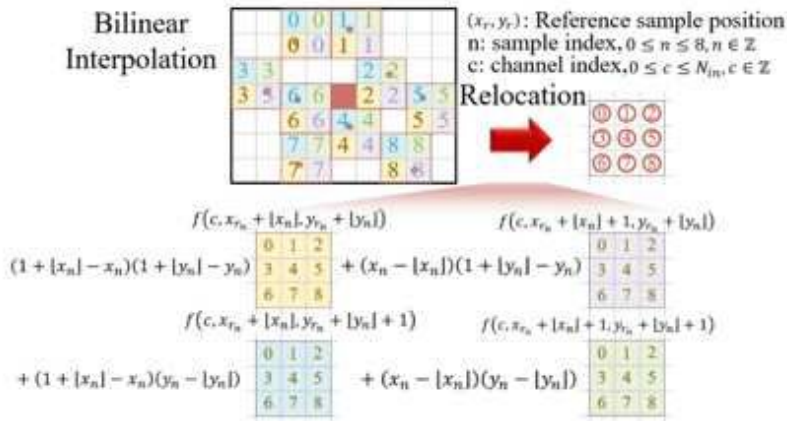


(b)

도면2

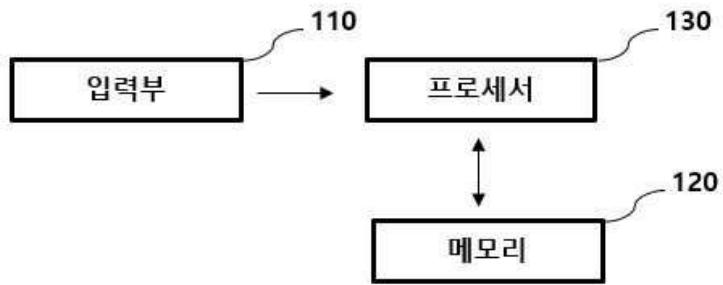


(a)

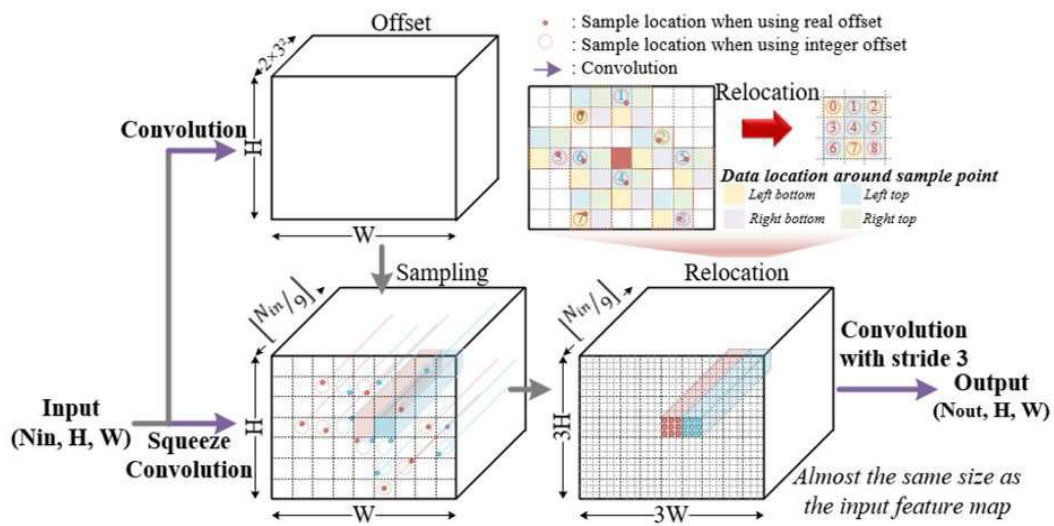


(b)

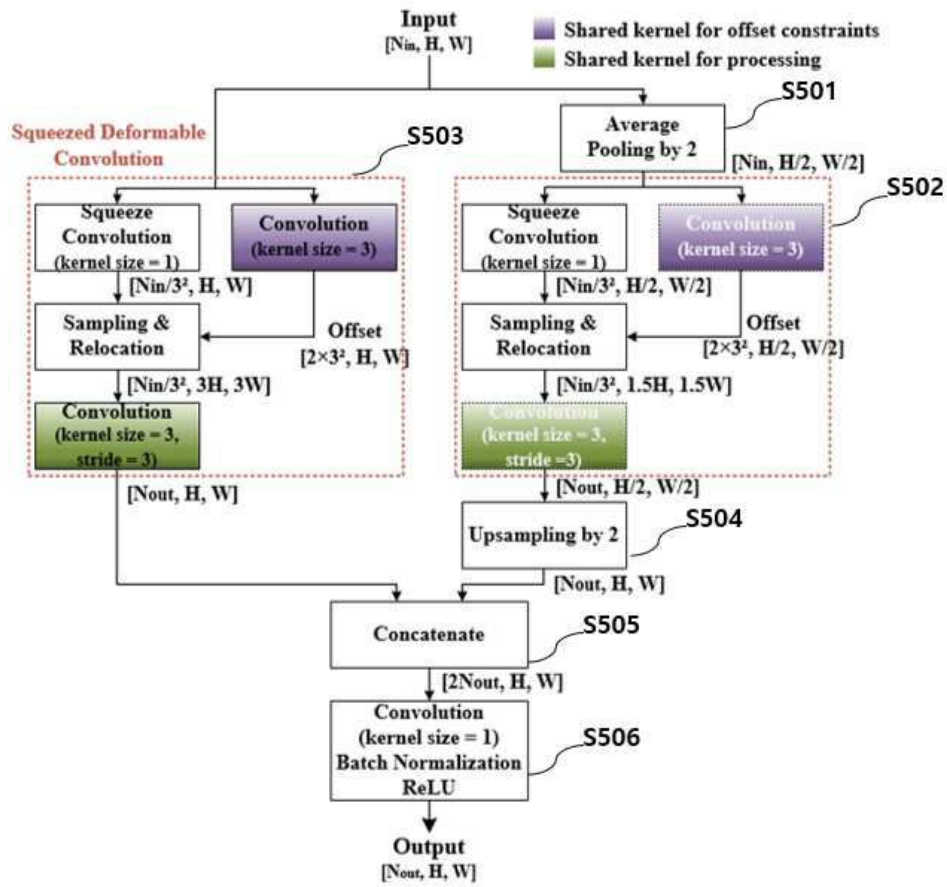
도면3



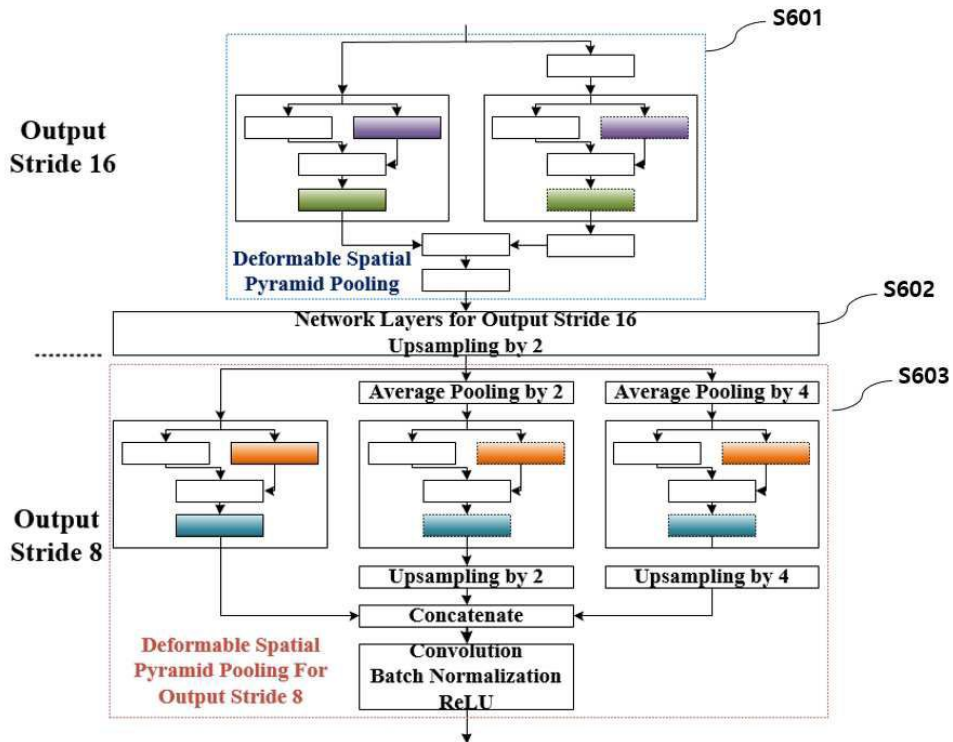
도면4



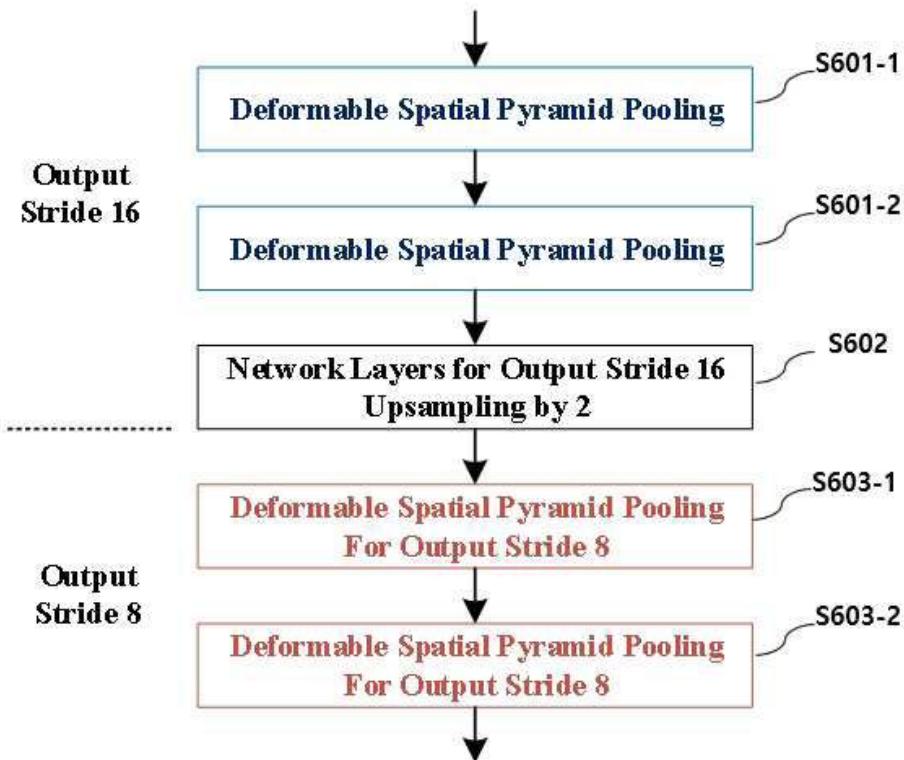
도면5



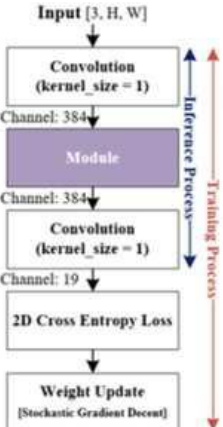
도면6



도면7



도면8

Network	Input Size	Module	Memory [MiB]		#Multi- plication [GOPs]
			Training	Inference	
	48 x 48 (Output Stride 16)	Conv	977	901	2.8
		ASPP	1,151	1,101	2.9
		SPP	1,023	979	1.6
		Df Conv	1,945	1,813	3
		SDF Conv (ours)	1,009	1,001 (923)	0.5
	96 x 96 (Output Stride 8)	DSPP (ours, branch 2EA)	1,099	1,075 (991)	1.2
		Conv	1,135	941	11.4
		ASPP	1,707	1,555	11.5
		SPP	1,409	1,265	6.3
		Df Conv	4,875	4,623	11.9
		SDf Conv (ours)	1,457	1,353 (1,045)	1.9
		DSPP (ours, branch 2EA)	1,957	1,783 (1,415)	6.3

도면9

Module	Training	Integer Offset	Fine Tuning
Squeezed Deformable Convolution	76.2	75.7	76.0
DSPP	76.6	75.9	76.4
Extended DSPP	78.4	77.0	77.9

도면10

Type	Name	Applied Position	mIoU (%)
Layer	Original	Output Stride 16	70.9
	Deformable Convolution	Output Stride 16	72.0
	Squeezed Deformable Convolution (ours)	Output Stride 16	71.6
Module	SPP	Output Stride 16	71.2
	ASPP	Output Stride 16	71.2
	DSPP (ours)	Output Stride 16	73.3
Extension	SPP Extension	Output Stride 8, 16	71.7
	ASPP Extension	Output Stride 8, 16	71.8
	Extended DSPP (ours)	Output Stride 8, 16	74.2
	Duplicate Extended DSPP (ours)	Output Stride 8, 16	76.9

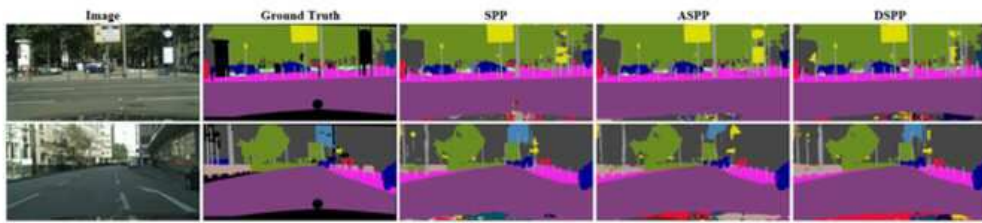
도면11

Type	Name	Apply Position	mIoU (%)
Layer	Deformable Convolution	Output Stride 16	76.0
	Squeezed Deformable Convolution (ours)	Output Stride 16	76.1
Module	SPP	Output Stride 16	76.6
	ASPP	Output Stride 16	75.4
	DSPP (ours)	Output Stride 16	76.9
Extension	SPP Extension	Output Stride 8, 16	76.8
	ASPP Extension	Output Stride 8, 16	76.4
	Extended DSPP (ours)	Output Stride 8, 16	77.9

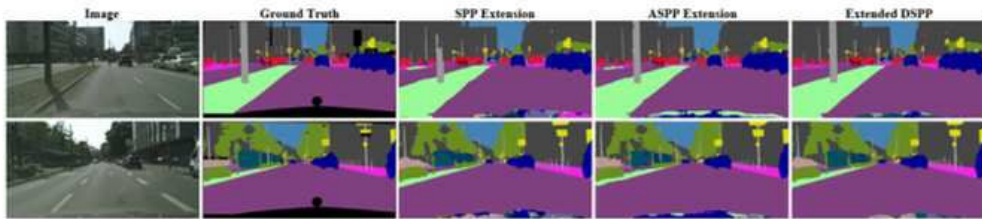
도면12

Type	Name	Apply Position	mIoU (%)
Layer	Original	Output Stride 16	75.2
	Deformable Convolution	Output Stride 16	76.1
	Squeezed Deformable Convolution (ours)	Output Stride 16	76.2
Module	SPP	Output Stride 16	75.1
	ASPP	Output Stride 16	75.9
	DSPP (ours)	Output Stride 16	76.6
Extension	SPP Extension	Output Stride 8, 16	77.3
	ASPP Extension	Output Stride 8, 16	77.2
	Extended DSPP (ours)	Output Stride 8, 16	78.4

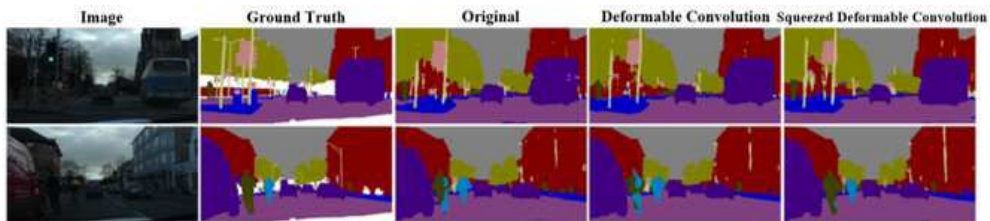
도면13



(a)



(b)



(c)