Ideas in Models

이현수, 권태희

Contents

Environment Setup

- 사용 IDE: 제공받은 GPU server 및 Google Colab Pro+
- Training data: 1~280, Validation data: 281~407 (7:3 비율)
- Test data: leaderboard data 1~58

Model 1: Attention U-Net (Modified)

- Attention U-Net[1]
  - ✓ 기본 U-Net 에 Attention Gate 가 추가된 구조
  - ✓ Attention Gate 를 통해, Image 에서 Attention Coefficients 가 계산됨으로써 Image 에서 집중할 부분에 대한 정보가 추출됨
  - ✓ CT pancreas segmentation Problem 을 위해 제안되었지만, 직접 시도해 본 결과 뇌 MRI 데이터에 대해서도 좋은 성능을 나타내었음
  - ✓ 논문에서 제시된 Model 구조를 변형하여 사용
  - ✓ 기본 U-Net[2] 에 비해 더 좋은 성능을 나타냄

- Original Attention U-Net

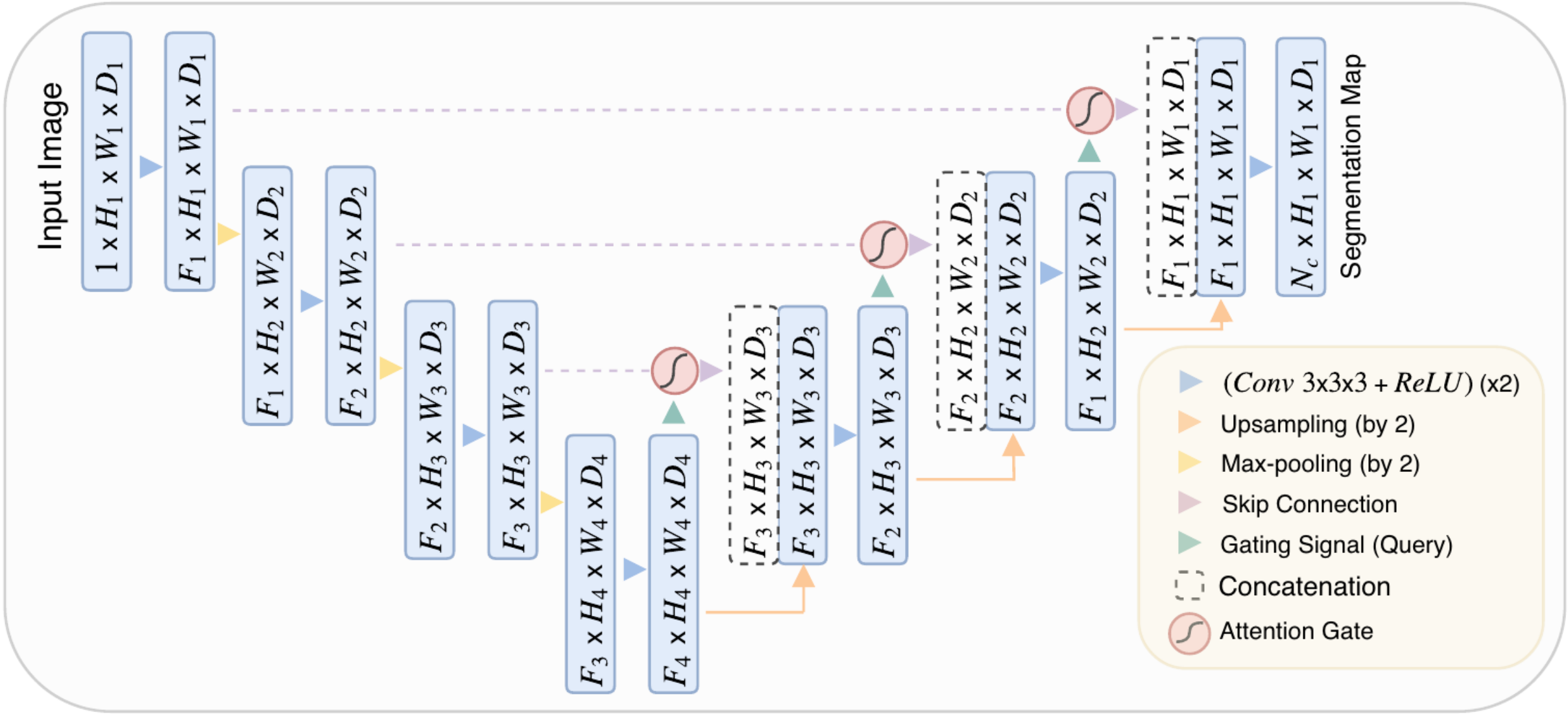
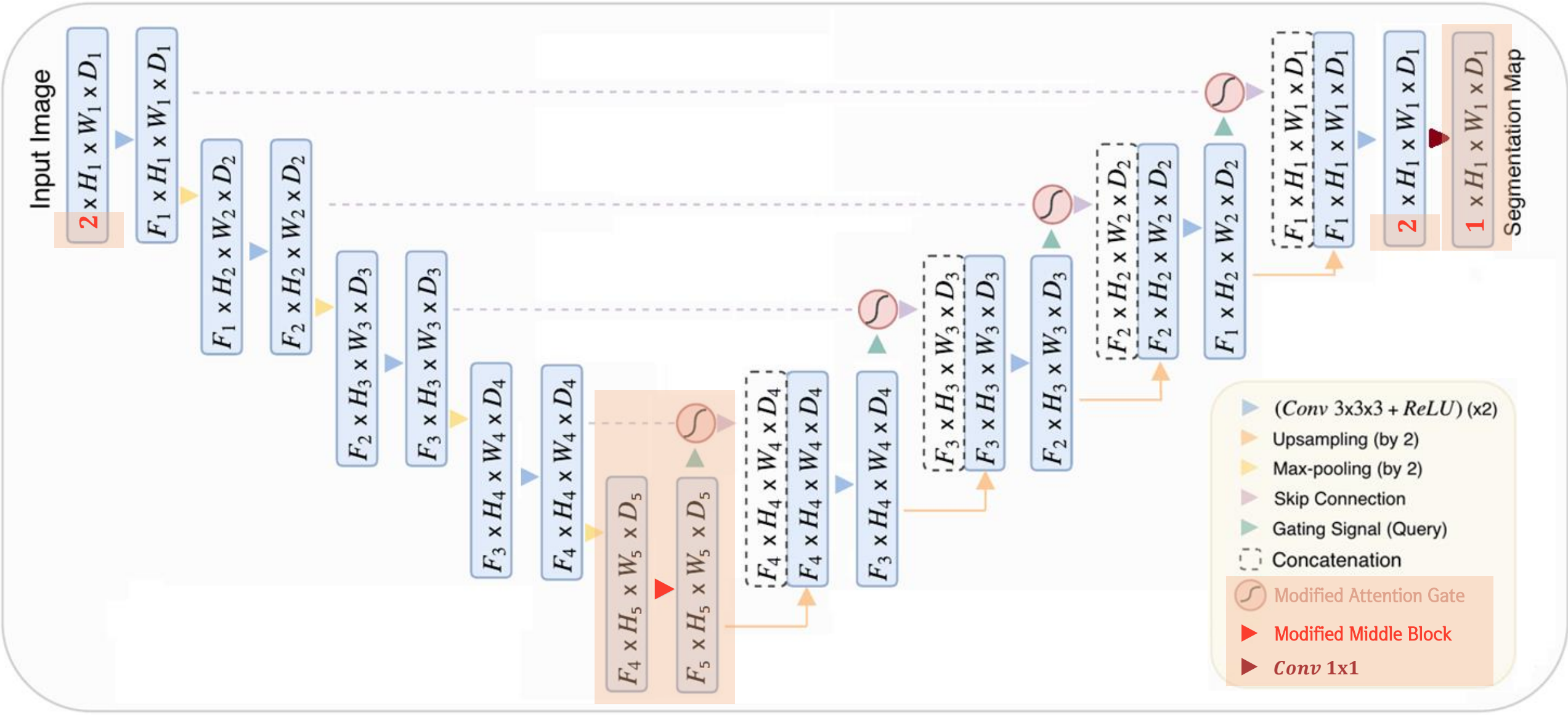


그림 출처 : Reference [1]

- Modified Attention U-Net



- Modified Attention U-Net

  1) Image + Grappa 2 Channel 로 사용

  2) Attention Gate 수정

  3) Middle Block 수정

  4) 출력 시 Channel 수를 맞춰주기 위해 1x1 Convolution 이용[3]

  * Attention Gate 만 변화시킨 경우 : ValLoss 0.03979 / 63epoch

  * 위 4가지를 모두 적용시킨 경우 : ValLoss 0.0337 / 45epoch

- Original Attention Gate

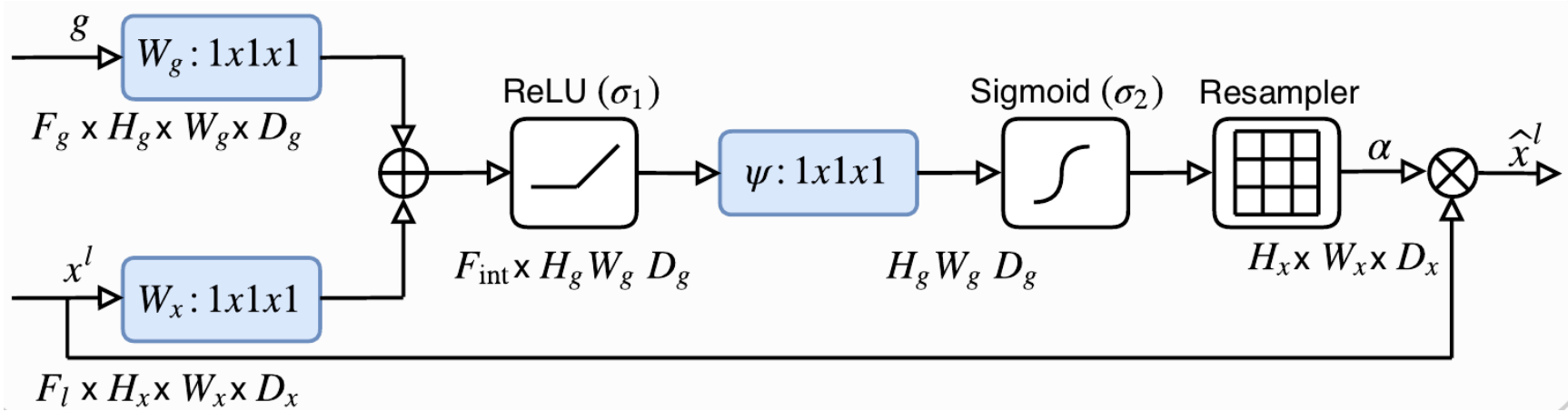
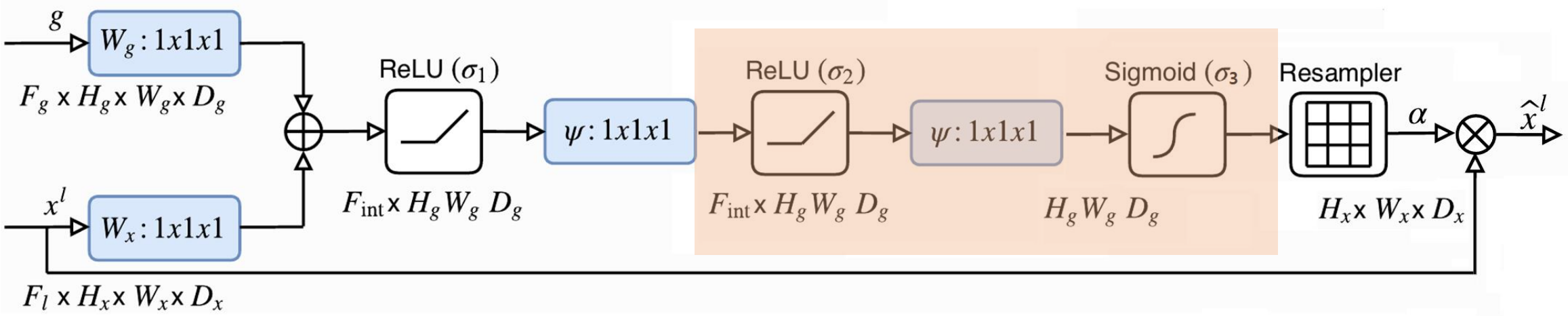


그림 출처 : Reference [1]

- Modified Attention Gate



- Original Middle Block

  Conv 3x3x3 + BatchNorm + ReLU

  Conv 3x3x3 + BatchNorm + ReLU

  ConvTranspose2D + BatchNorm + ReLU

- Modified Middle Block

  Conv 3x3x3 + InstanceNorm + ReLU

  Conv 3x3x3 + InstanceNorm + ReLU

  Max-Pooling (by 12)

  FC Layer 1 (4096 → 2048)

  FC Layer 2 (2048 → 4096)

  ConvTranspose2D

  ConvTranspose2D + InstanceNorm + ReLU

- FC Layer

  Linear

  LeakyReLU (negative slope : 0.2)

  Dropout (ratio : 0.3)

Model 1: Attention U-Net (Modified)

- Data Preprocessing
  - ✓ 논문 및 작년 대회 우승자의 영상을 참고
  - ✓ Data Mixup 을 사용하려고 시도
    - Paper "*mixup* : Beyond Empirical Risk Minimization" (2018)[4]

$$\tilde{x} = \lambda x_i + (1-\lambda)x_j,$$
$$\tilde{y} = \lambda y_i + (1-\lambda)y_j,$$
$$\text{where } \lambda \sim \beta(0.4)$$

⟶

```
for (x1, y1), (x2, y2) in zip(loader1, loader2):
    lam = numpy.random.beta(alpha, alpha)
    x = Variable(lam * x1 + (1. - lam) * x2)
    y = Variable(lam * y1 + (1. - lam) * y2)
    optimizer.zero_grad()
    loss(net(x), y).backward()
    optimizer.step()
```

그림 출처 : Reference [4]

  - ✓ 그러나 Computing resource 제한으로 인해 Batch Size = 1 로 설정하였고,
    따라서 실질적으로 Mixup 에 의한 효과를 적용하지 못함.

Model 2: End-to-End Varnet (Modified)

- End-to-End Varnet[5]

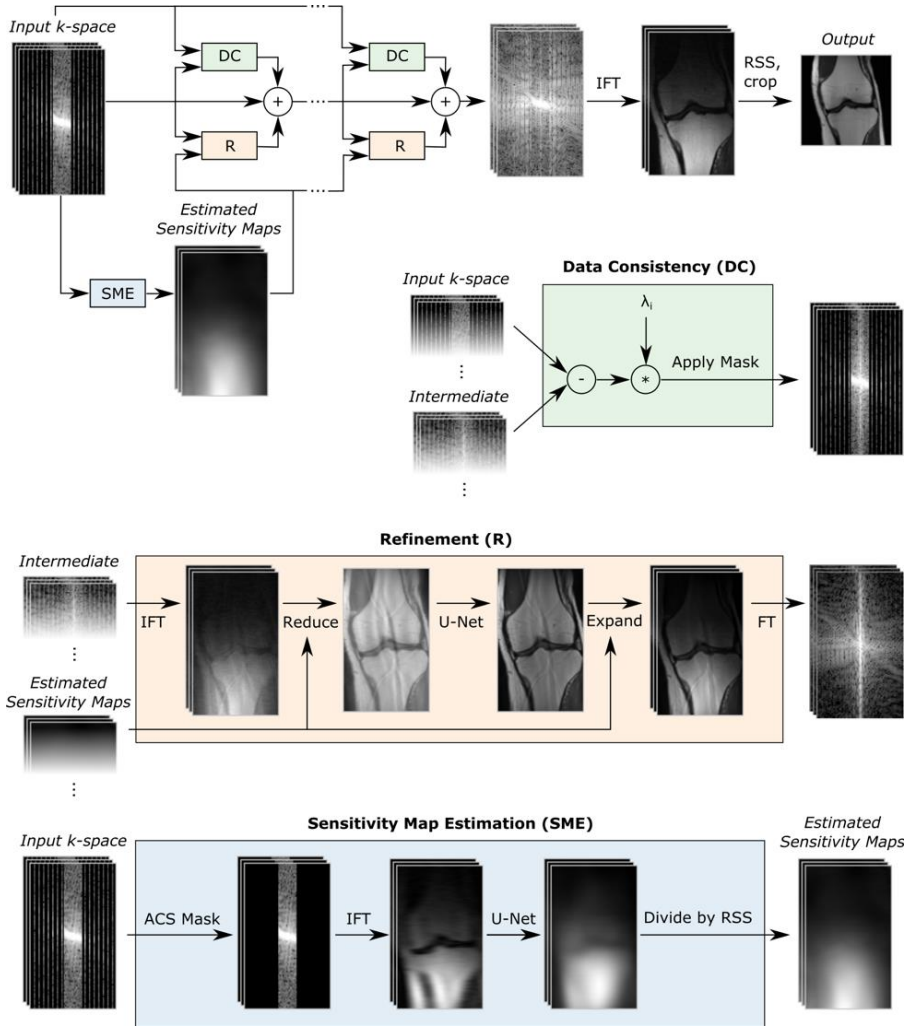


그림 출처 : Reference [5]

Model 2: End-to-End Varnet (Modified)

- End-to-End Varnet[5]
  1) 변형한 모델에서의 Computing Resource 한계 → Cascade # 을 6으로 감소
  2) Fine-tuning : Pretrained Weights 를 이용해 Fine-tuning 진행
  3) U-Net 구조 변경 (작년 우승자의 영상 참고)

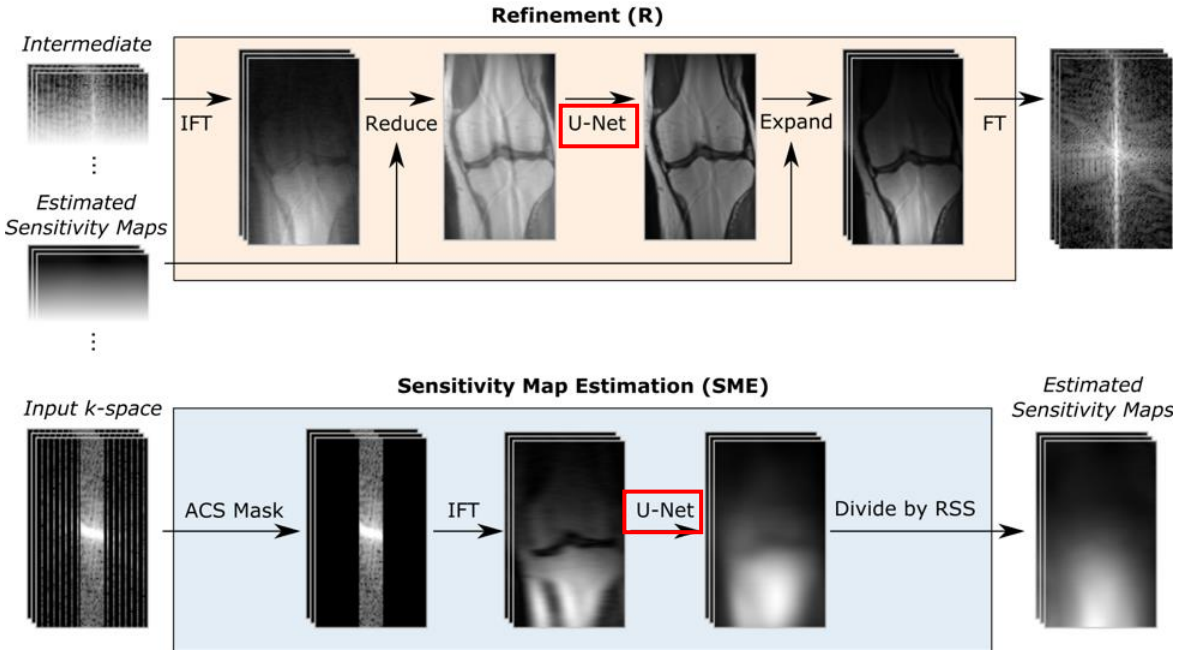
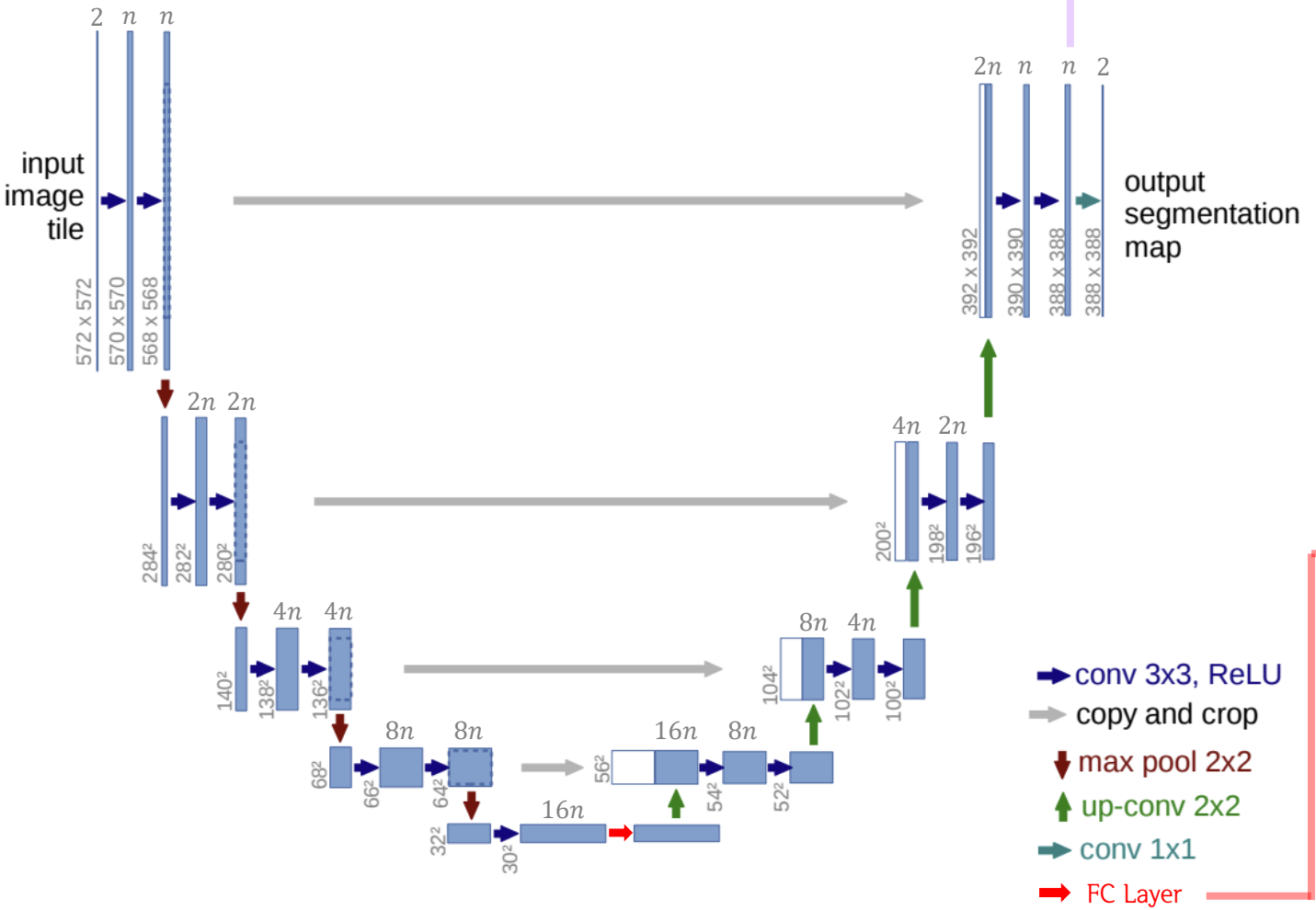


그림 출처 : Reference [5]

Model 2: End-to-End Varnet (Modified)

- U-Net 구조 변경 방식



In Sensitivity Map : 8
In Refinement : 18

1) Max pooling (computing resource 한계로 인해, FC Layer 의 Parameter 수를 감소시키기 위함)

2) Vectorization

3) Linear Layer + ReLU

4) Dropout(0.5)

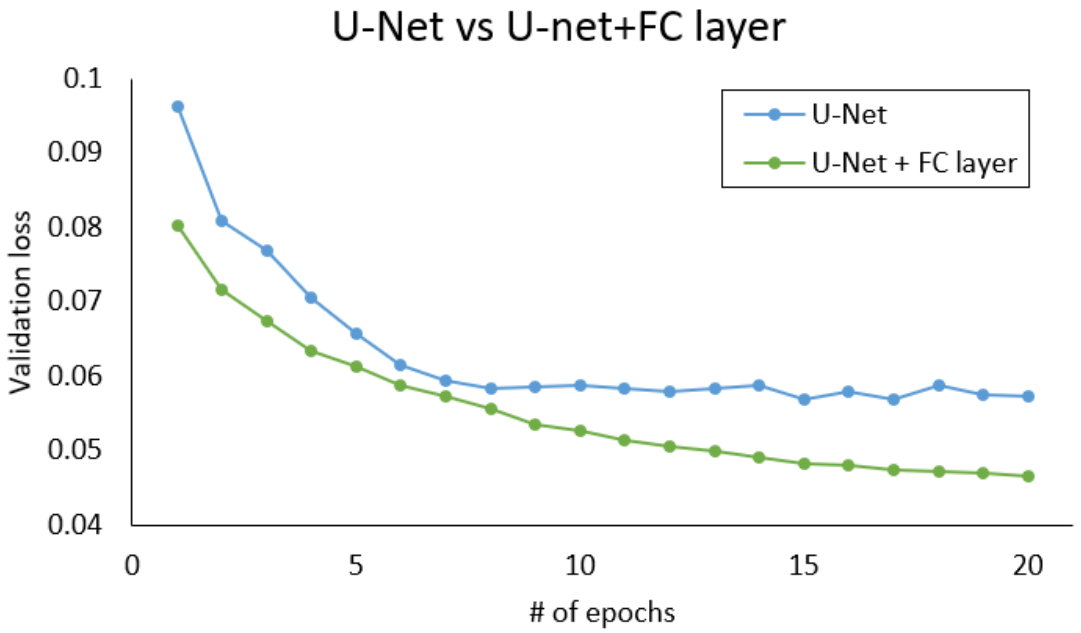
5) Reshape (Convolution 을 적용할 수 있는 shape 로)

6) ConvTranspose2D (Max pooling 해서 줄인 Image Size 를 다시 원래대로 복원시켜 주는 역할)

그림 출처 : Reference [2]

Model 2: End-to-End Varnet (Modified)

- U-Net 맨 아래 부분에 FC Layer 를 추가함으로써, Feature Extraction 이 더욱 효율적으로 진행될 수 있을 것이라 판단
- 기본 U-Net 과 FC Layer 를 추가한 U-Net 을 비교해 본 결과, 후자의 Val. Loss 가 더 작게 나타남

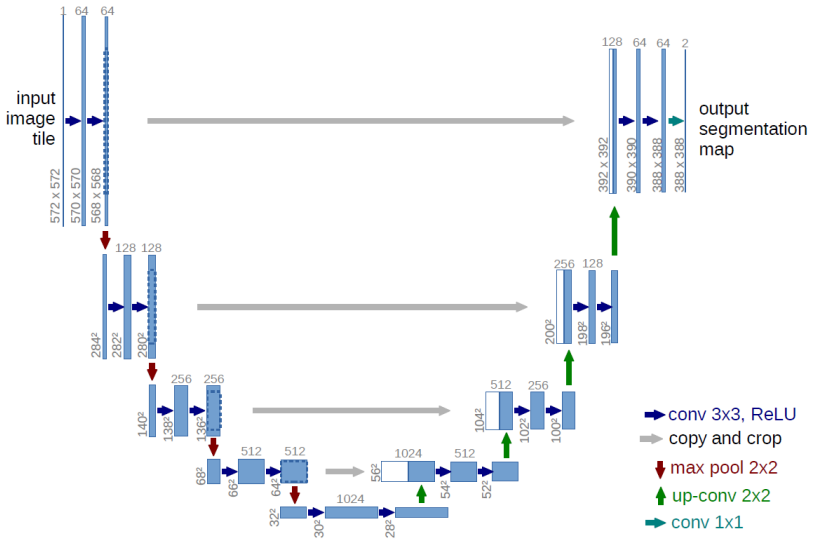


- 따라서, E2E Varnet 의 U-Net 도 변경

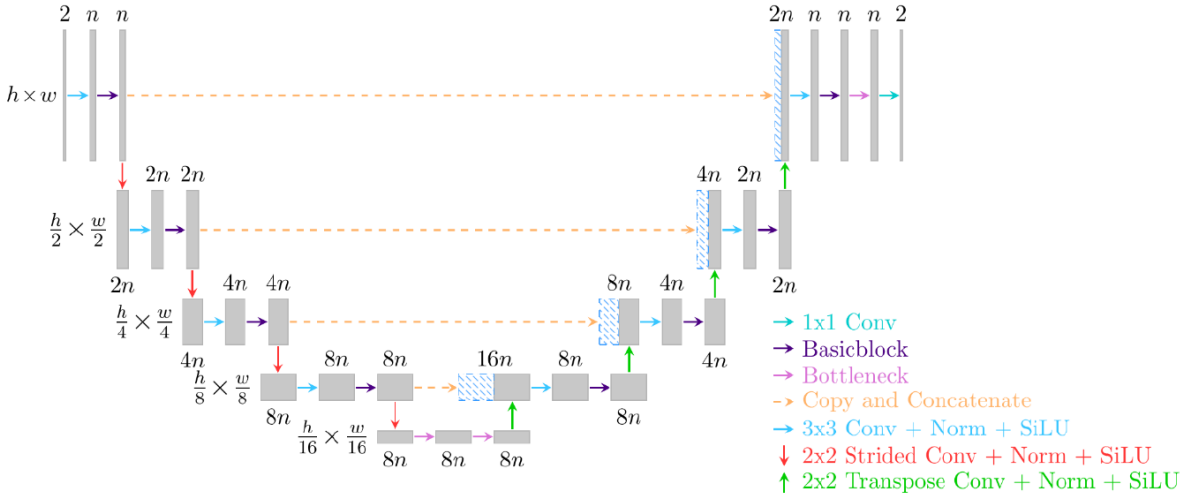
Model 3&4: E2E Varnet extension - DIRCN (Modified)

- DIRCN(Densely Interconnected Residual Cascading Network)[6]
  - ✓ Baseline model로 End-to-End Varnet을 사용, 세 가지의 아이디어 적용

1) Refinement of CNNS: ResXUnet



U-Net



ResXUnet

그림 출처 : Reference [2]

그림 출처 : Reference [6]

Model 3&4: E2E Varnet extension - DIRCN (Modified)

1) Refinement of CNNS: ResXUnet

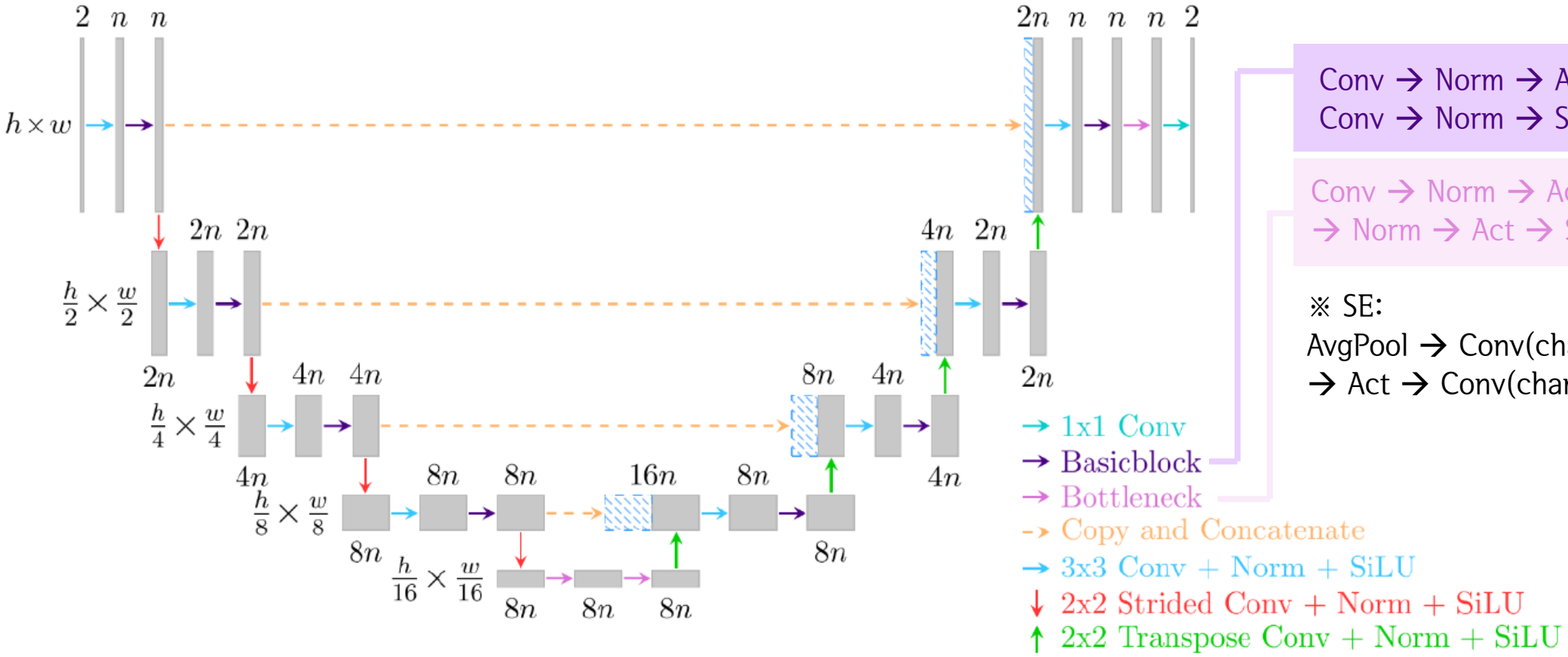
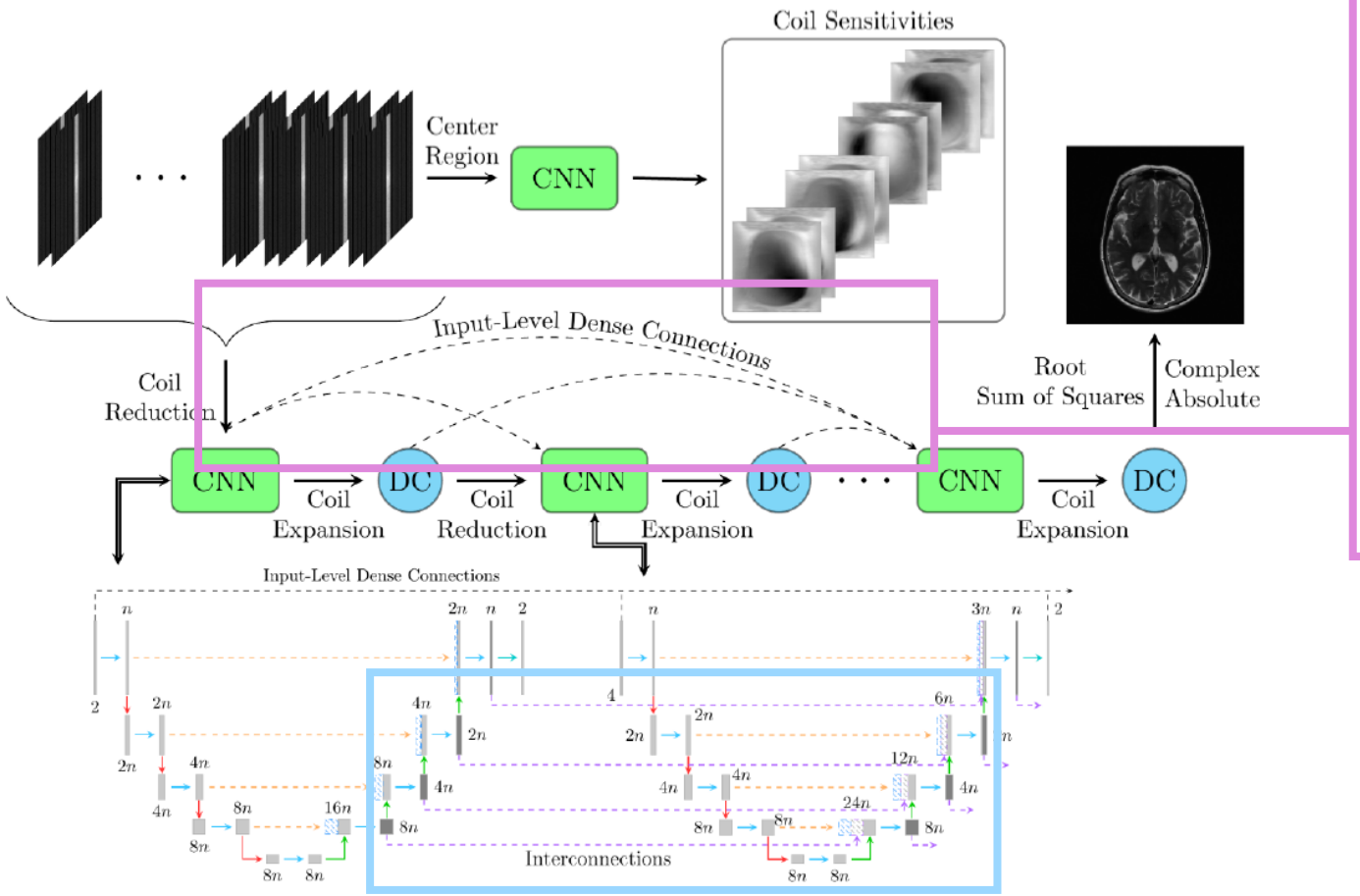


그림 출처 : Reference [6]

Model 3&4: E2E Varnet extension - DIRCN (Modified)

2) Input-level Dense Connections[7]

3) Long Range Skip Connections



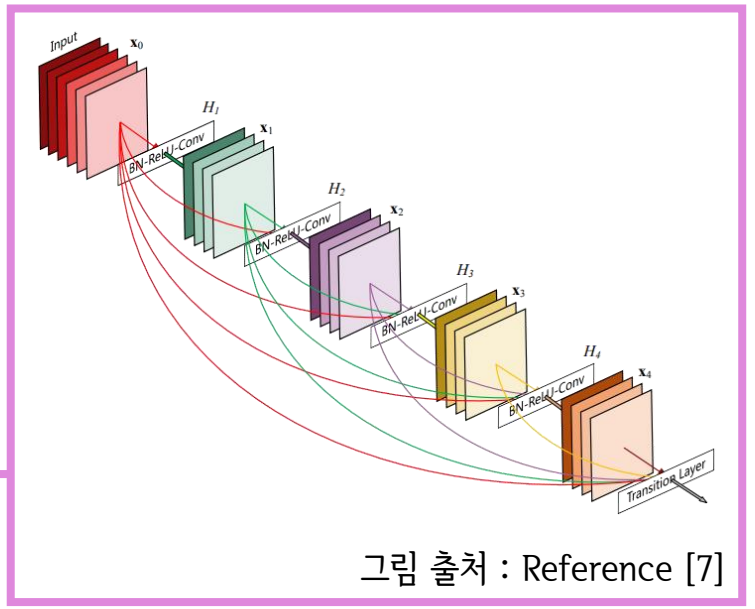


그림 출처 : Reference [7]

그림 출처 : Reference [6]

Model 3&4: E2E Varnet extension - DIRCN (Modified)

- Computing resource 제한으로 인해, Train 가능한 모델의 cascade는 최대 2
- 제약조건 하에서, deeper network를 구축하는 것을 목표로

				Four-fold acceleration			Eight-fold acceleration		
Architecture	#Params	Inference time [ms]	Modality	SSIM	NMSE	PSNR	SSIM	NMSE	PSNR
Baseline	45M	148 ± 3	T1	0.9626	0.0033	41.5	0.9466	0.0070	38.2
			T2	0.9556	0.0044	40.0	0.9399	0.0096	36.6
			FLAIR	0.9357	0.0055	39.1	0.9123	0.0111	36.1
			ALL	0.9560	0.0041	40.4	0.9395	0.0088	37.0
Baseline + Dense	45M	153 ± 2	T1	0.9637	0.0031	41.7	0.9486	0.0064	38.6
			T2	0.9569	0.0042	40.2	0.9420	0.0085	37.0
			FLAIR	0.9378	0.0053	39.3	0.9154	0.0100	36.5
			ALL	0.9574	0.0039	40.6	0.9417	0.0080	37.5
Baseline + ResXUNet	41M	394 ± 4	T1	0.9635	0.0031	41.7	0.9485	0.0065	38.6
			T2	0.9565	0.0042	40.2	0.9417	0.0087	36.9
			FLAIR	0.9370	0.0053	39.3	0.9149	0.0101	36.5
			ALL	0.9569	0.0040	40.6	0.9415	0.0081	37.4
Baseline + Interconnections	49M	159 ± 2	T1	0.9638	0.0030	41.8	0.9482	0.0065	38.6
			T2	0.9567	0.0041	40.3	0.9419	0.0085	37.0
			FLAIR	0.9375	0.0053	39.3	0.9144	0.0101	36.5
			ALL	0.9573	0.0039	40.7	0.9415	0.0080	37.5
DIRCN	47 M	387 ± 5	T1	0.9658	0.0028	42.2	0.9529	0.0054	39.3
			T2	0.9588	0.0038	40.7	0.9460	0.0072	37.7
			FLAIR	0.9408	0.0048	39.8	0.9216	0.0085	37.2
			ALL	**0.9594**	**0.0035**	**41.1**	**0.9460**	**0.0068**	**38.2**

표 출처 : Reference [6]

Model 3&4: E2E Varnet extension - DIRCN (Modified)

- DIRCN의 세 idea 중, ResXUnet 사용을 다른 U-Net 계열 모델로 대체
  - ✓ Baseline model에 ResXUnet 사용의 아이디어만 적용한 경우, SSIM이 가장 낮음
  - ✓ CNN 내부에서는 Parameter #의 조절이 간편 → Computing resource 제한에 직접적으로 접근

- Model 3: ResXUnet을 본래대로, U-Net+FC layer 으로 대체
  - ✓ Parameter 수의 급격한 감소를 기대 → Cascade #를 6까지 증가 가능

- Model 4: ResXUnet을 Attention U-Net 으로 대체
  - ✓ 마찬가지로, Parameter 수의 급격한 감소 → Cascade #를 5까지 증가 가능
  - ✓ 직접 train한 U-Net 계열 model 중 우수한 성능을 확인

Model 3&4: E2E Varnet extension - DIRCN (Modified)

- Model 3: ResXUnet 을 본래대로, U-Net+FC layer 으로 대체
  - ✓ Parameter 수의 급격한 감소를 기대 → Cascade #를 6까지 증가 가능

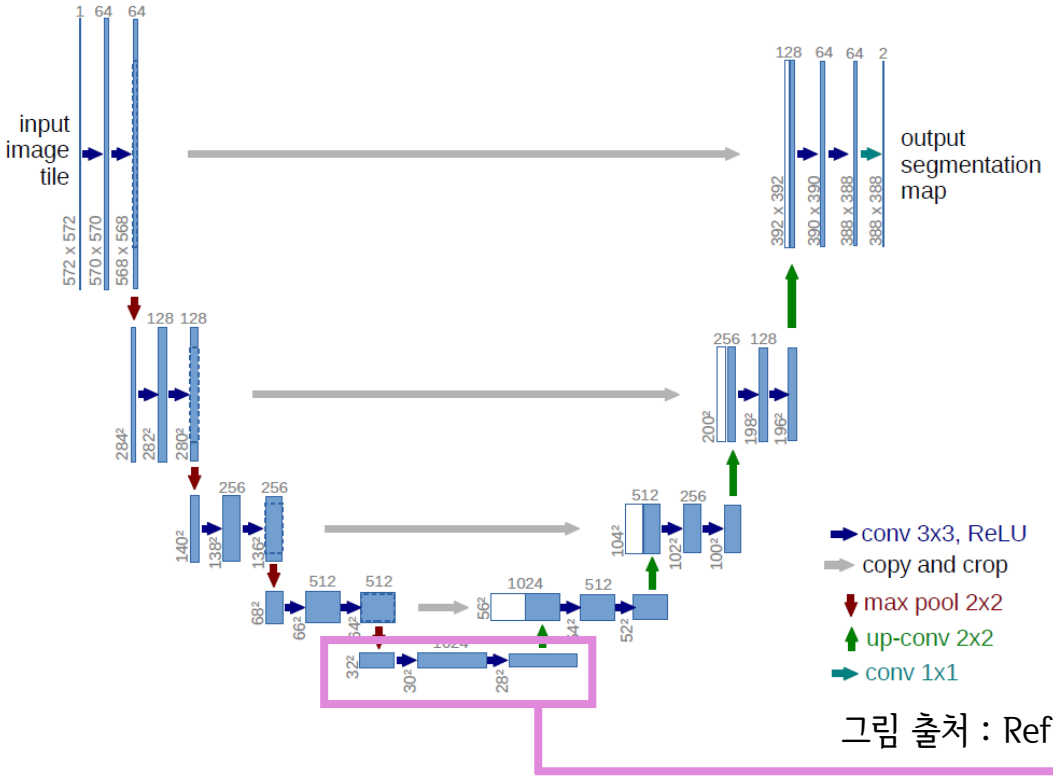


그림 출처 : Reference [2]

- ✓ 첫 번째 layer 통과 이후의 channel 수: 18, 8 (sensitivity)
- ✓ Fc layer: 가장 깊은 layer에서 효과적인 feature 추출

Max-Pooling(파라미터 수 감소를 위함) → Flatten

→ FC layer → Reshape (이후 up-conv 과정을 이어가기 위함)

Model 3&4: E2E Varnet extension - DIRCN (Modified)

- Model 4: ResXUnet 을 Attention U-Net 으로 대체
  - ✓ 마찬가지로, Parameter 수의 급격한 감소 → Cascade #를 5까지 증가 가능
  - ✓ 직접 train한 U-Net 계열 model 중 우수한 성능을 확인
  - ✓ 추가적으로, Parameter 수를 감소시키기 위해
    - ✓ Model 1에서 추가한 FC layer 중 하나를 없앰
    - ✓ 층의 깊이를 4에서 3으로 변경

Training Method

- Common Method
  - ✓ Loss Function: $1 - SSIM$ (Skeleton Code의 함수 사용)
  - ✓ Optimizer: Adam
  - ✓ 각 epoch마다 checkpoint 저장, 가장 작은 validation loss 를 가지는 ‘best model’ 에 대해서는 별도로 저장
  - ✓ Colab 사용량 제한 시, 저장한 checkpoint를 불러와 이어서 train 진행
  - ✓ Test 과정에서는 best model 을 사용

Training Method

- Model 1: Attention U-Net (Modified)
  - ✓ Batch Size: 1
  - ✓ Learning rate: 초깃값 1e-3
  - ✓ Learning rate scheduler: MultiStepLR

    (10, 15, 20, 25 epoch 에 Learning Rate 가 절반씩 감소하게끔 사용)

- Model 2: End-to-End Varnet (Modified)
  - ✓ Batch Size: 1
  - ✓ Cascade #: 6
  - ✓ Learning rate: 초깃값 1e-3
  - ✓ Learning rate scheduler: 별도의 scheduler 없이, validation loss의 감소가 멈추거나 더딘 경우 감소시킴
  - ✓ Pretrained Model 을 이용해 Fine-tuning

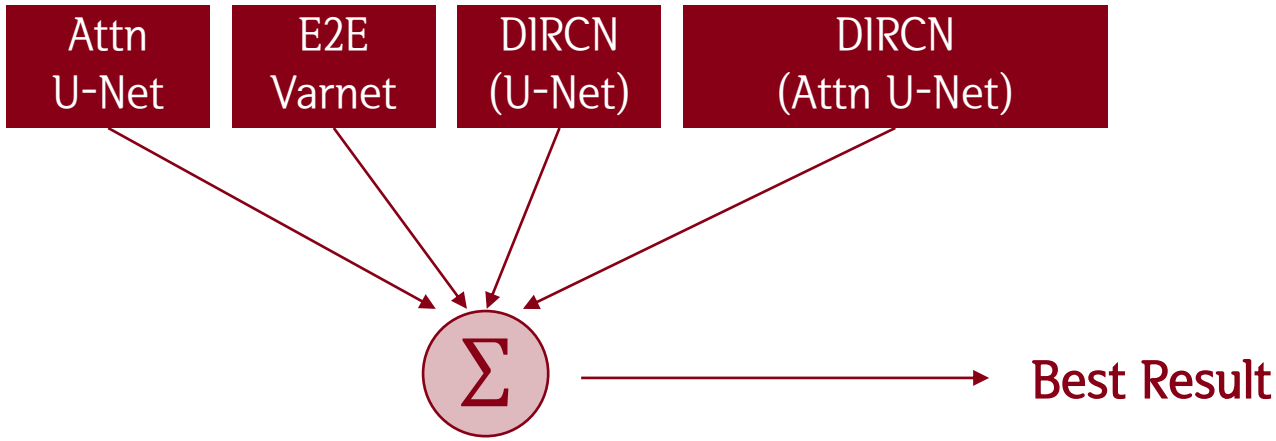
Training Method

- Model 3: DIRCN with U-Net
  - ✓ Batch Size: 1
  - ✓ Cascade #: 6
  - ✓ Learning rate: 초깃값 1e-3
  - ✓ Learning rate scheduler: 별도의 scheduler 없이, validation loss의 감소가 멈추거나 더딘 경우 감소시킴

- Model 4: DIRCN with Attention U-Net
  - ✓ Batch Size: 1
  - ✓ Cascade #: 5
  - ✓ Learning rate: 초깃값 1e-3
  - ✓ Learning rate scheduler: 별도의 scheduler 없이, validation loss의 감소가 멈추거나 더딘 경우 감소시킴

Model Ensemble

- 4개의 모델을 Ensemble 해서 최종 Reconstruction 도출
  - 각 모델에서 생성된 Reconstruction 들을 가중합
  - 이 결과를 바탕으로 SSIM 산출



Model Ensemble

- 4개의 모델의 최종 Validation Loss

Attn U-Net	0.0337 / 45 Epoch
E2E Varnet	0.02831 / 30 Epoch
DIRCN (U-Net)	0.0293 / 14 Epoch
DIRCN (Attn U-Net)	0.0313 / 16 Epoch

- Ensemble 비율은 1 : 1 : 1 : 1 로 설정
  - 4개 모델의 Validation Loss 차이가 크지 않아, 1 : 1 : 1 : 1 근처에서 비율을 바꾸어 가며 적절한 비율을 찾음
  - 여러 가지 비율을 시도해 본 결과, 가장 높은 SSIM 값을 나타내는 비율을 찾음

Result

- SSIM of 4-model Ensemble (1 : 1 : 1 : 1) → **0.9841**

References

[1] Oktay, Ozan & Schlemper, Jo & Folgoc, Loic & Lee, Matthew & Heinrich, Mattias & Misawa, Kazunari & Mori, Kensaku & McDonagh, Steven & Hammerla, Nils & Kainz, Bernhard & Glocker, Ben & Rueckert, Daniel. (2018). Attention U-Net: Learning Where to Look for the Pancreas.

[2] Ronneberger, O., Fischer, P. and Brox, T. (2015) U-net: Convolutional Networks for Biomedical Image Segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, Cham, 234-241. https://doi.org/10.1007/978-3-319-24574-4_28

[3] Lin, M., Chen, Q., & Yan, S. (2013). Network in network. arXiv preprint arXiv:1312.4400.

[4] Zhang, H., Cissé, M., Dauphin, Y., & Lopez-Paz, D. (2018). mixup: Beyond Empirical Risk Minimization. ArXiv, abs/1710.09412.

References

[5] Sriram, Anuroop & Zbontar, Jure & Murrell, Tullie & Defazio, Aaron & Zitnick, C. & Yakubova, Nafissa & Knoll, Florian & Johnson, Patricia. (2020). End-to-End Variational Networks for Accelerated MRI Reconstruction. 10.1007/978-3-030-59713-9_7.

[6] Ottesen, Jon & Caan, Matthan & Groote, Inge & Bjørnerud, Atle. (2022). A Densely Interconnected Network for Deep Learning Accelerated MRI. 10.48550/arXiv.2207.02073.

[7] Huang G, Liu Z, Van Der Maaten L, Weinberger KQ (2017) Densely connected convolutional networks. Proc - 30th IEEE Conf Comput Vis Pattern Recognition, CVPR 2017. doi: 10.1109/CVPR.2017.243