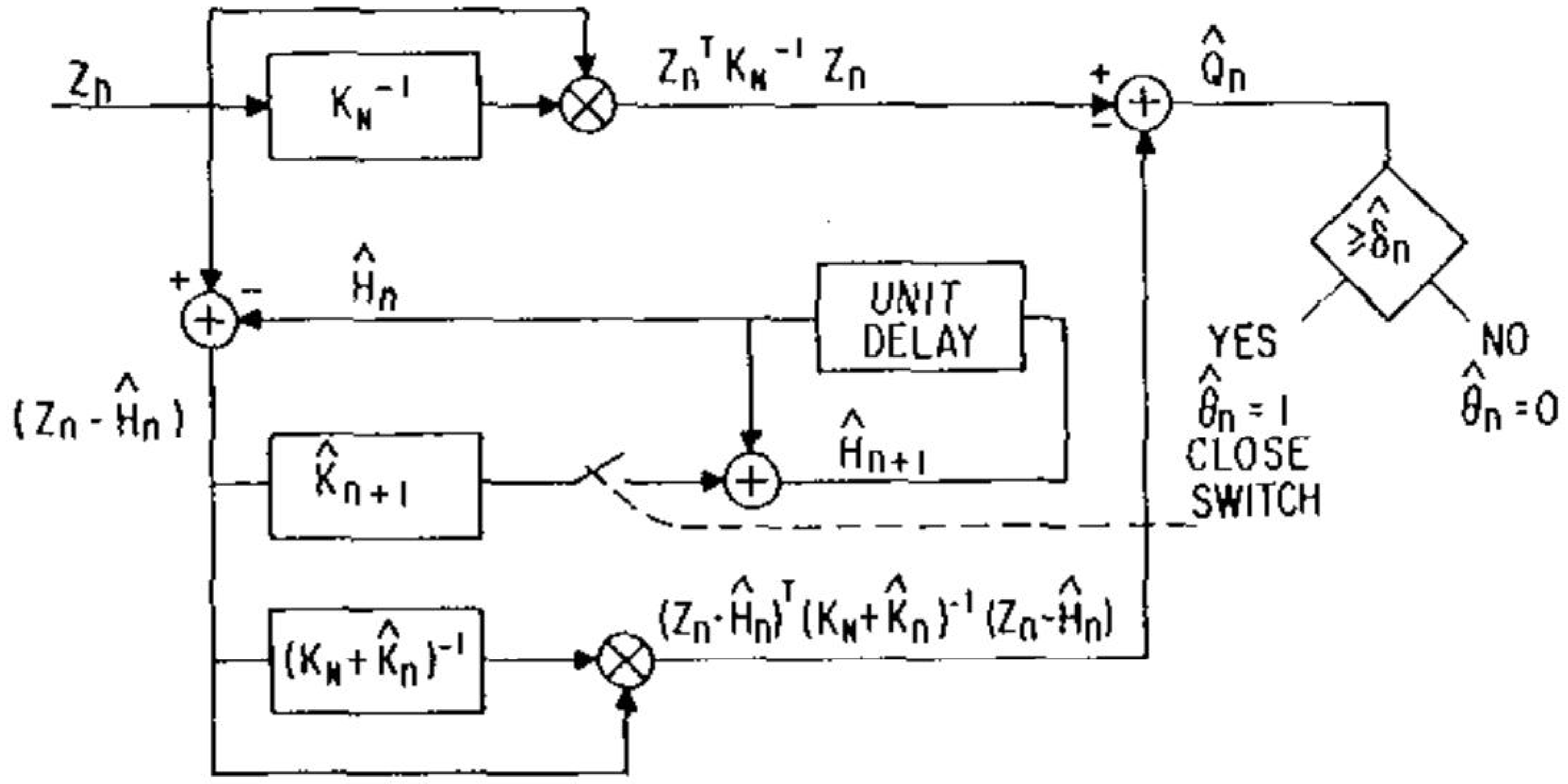
Self-Training for SE and ASR

Dong Wang

2020.06

Self-training for event detection



Scudder. Probability of error of some adaptive pattern-recognition machines.IEEE Transactions on InformationTheory, 11(3):363–371, 1965.

Noise2Noise

$$\underset{\theta}{\operatorname{argmin}} \sum_i L\left(f_\theta(\hat{x}_i), \hat{y}_i\right),$$

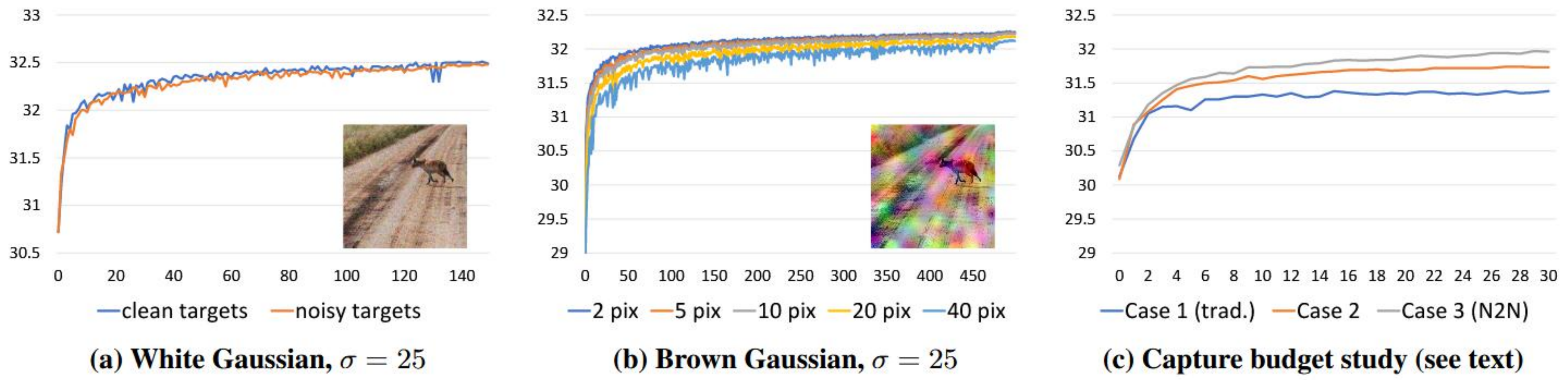


Figure 1. Denoising performance (dB in KODAK dataset) as a function of training epoch for additive Gaussian noise. (a) For i.i.d. (white) Gaussian noise, clean and noisy targets lead to very similar convergence speed and eventual quality. (b) For brown Gaussian noise, we observe that increased inter-pixel noise correlation (wider spatial blur; one graph per bandwidth) slows convergence down, but eventual performance remains close. (c) Effect of different allocations of a fixed capture budget to noisy vs. clean examples (see text).

Lehtinen J, Munkberg J, Hasselgren J, et al. Noise2noise: Learning image restoration without clean data[J]. arXiv preprint arXiv:1803.04189, 2018.

Noisy data only training

$$y = x + n$$

$$h(y) = y + g(y)$$

$$\eta(h(y)) = \sigma^2 + \frac{\|g(y)\|^2}{K} + \frac{2\sigma^2}{K}\sum_{i=1}^{K}\frac{\partial g_i(y)}{\partial y_i} = \frac{\|y - h(y)\|^2}{K} - \sigma^2 + \frac{2\sigma^2}{K}\sum_{i=1}^{K}\frac{\partial h_i(y)}{\partial y_i}$$

Theorem 1 ([23, 30]). *The random variable $\eta(h(y))$ is an unbiased estimator of*

$$\mathrm{MSE}(h(y)) = \frac{1}{K}\|x - h(y)\|^2$$

or

$$\mathbb{E}_{n\sim\mathcal{N}_{0,\sigma^2}}\left\{\frac{\|x - h(y)\|^2}{K}\right\} = \mathbb{E}_{n\sim\mathcal{N}_{0,\sigma^2}}\{\eta(h(y))\}$$

Soltanayev et al., Training deep learning based denoisers without ground truth data, NIPS 2018.

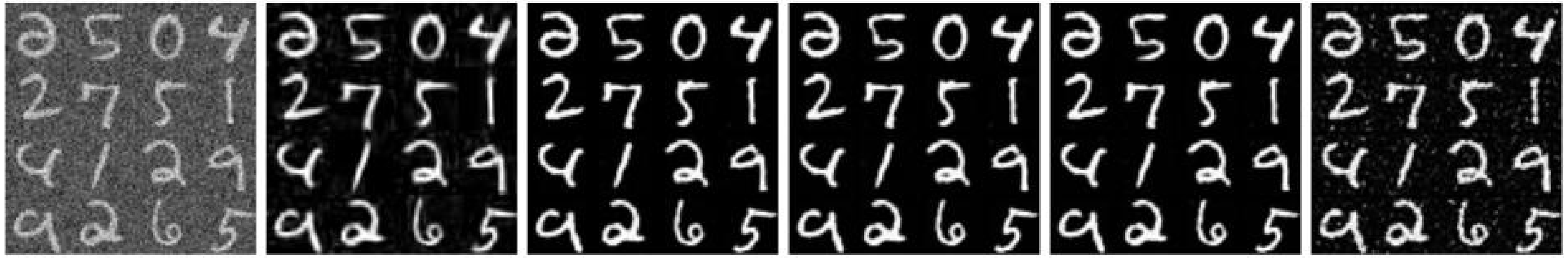

(a) Noisy image (b) BM3D (c) SDA-MSE-GT (d) SDA-SURE (e) SDA-SURE-T (f) SDA-REG

Figure 1: Denoising results of SDA with various methods for MNIST dataset at a noise level of σ=50.

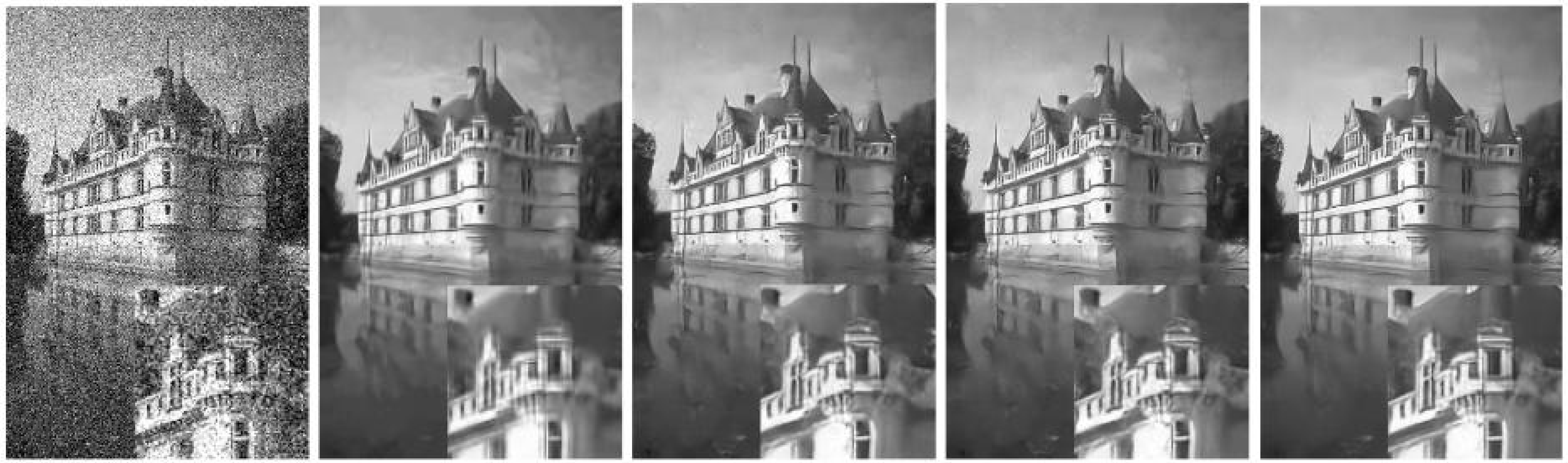

(a) Noisy image / 14.76dB (b) BM3D / 26.14dB (c) SURE / 26.46dB (d) SURE-T / 26.46dB (e) MSE / 26.85dB

Figure 5: Denoising results of an image from the BSD68 dataset for σ=50

Frame-by-Frame self-supervision

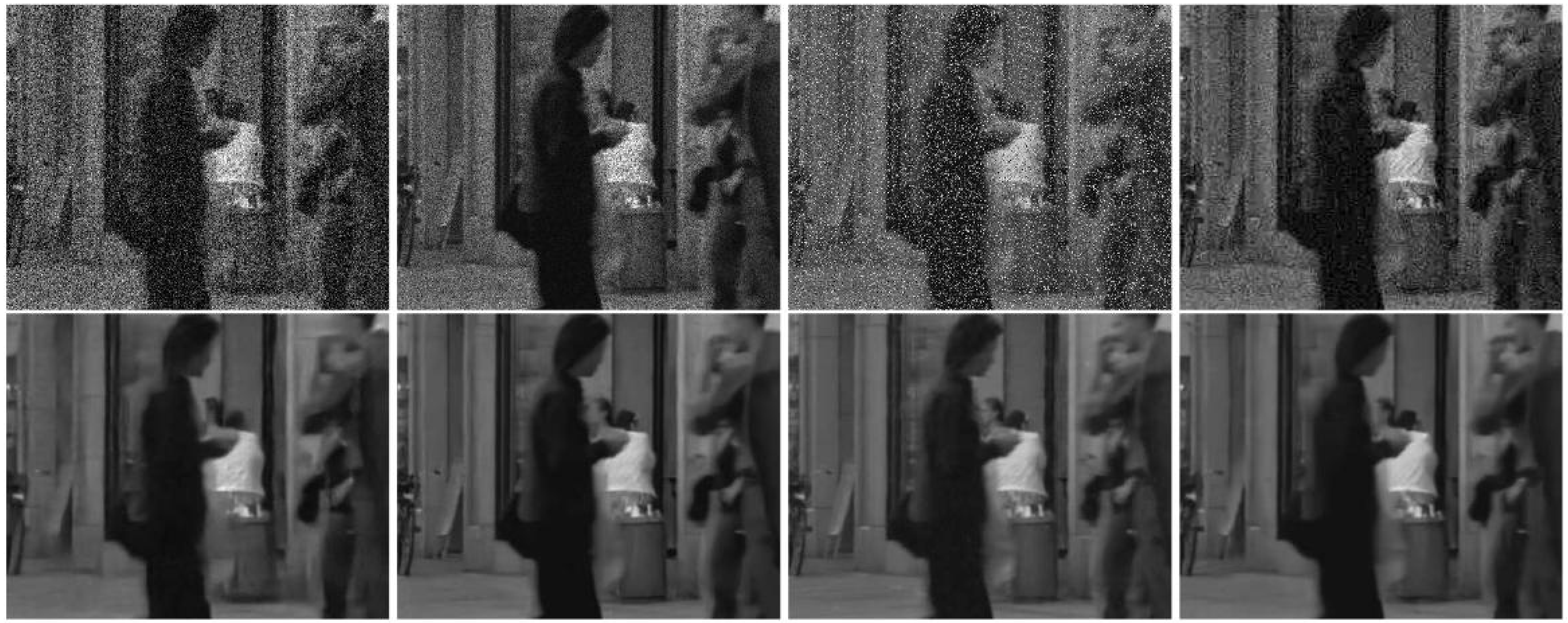


Figure 1: From the *same* starting point and *only* using the video, our fine-tuned network is able to denoise different noises without any artifact. The top images are the noisy and the bottom ones the denoised. From left to right: Gaussian noise, Poisson type noise, salt and pepper type noise and JPEG compressed Gaussian noise.

Ehret, Model-blind Video Denoising Via Frame-to-frame Training, CVPR 2019.

Noise mask training

- Mask center and predict the center

$$\underset{\theta}{\operatorname{argmin}} \sum_{i=1}^{n} -\ln\left(\int_{-\infty}^{\infty} p(s_i|\tilde{x}_{\mathrm{RF}(i)};\theta)p(x_i|s_i)ds_i\right).$$

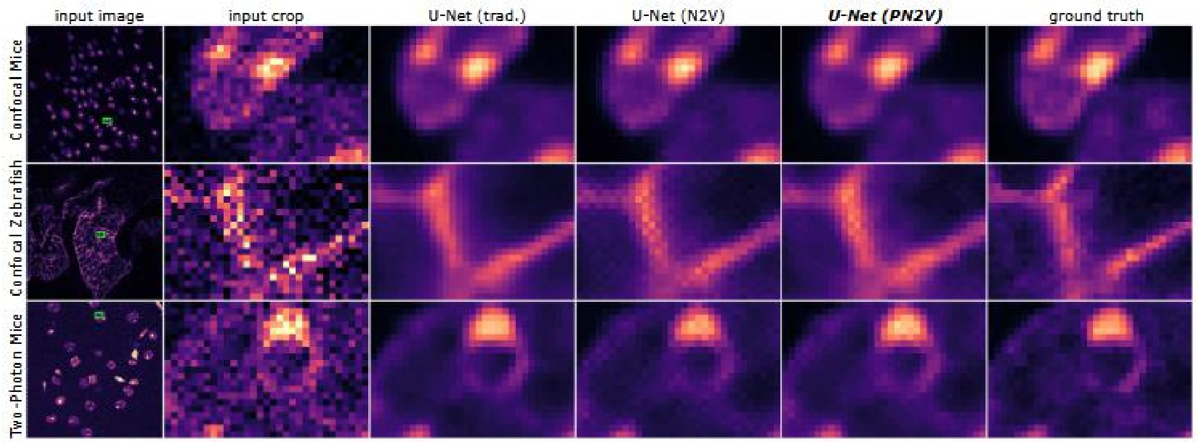


Fig. 2. Qualitative results for three images (rows) from the datasets we used in this manuscript. Left to right: raw image (NR1), zoomed inset, predictions by U-Net (trad.), U-Net (N2V), U-Net (PN2V), and ground truth data.

Krull et al, Noise2Void:Unsupervised Content-Aware Denoising, 2019.

More extension

$$\underbrace{p(\boldsymbol{y}|\Omega_y)}_{\text{Training data}} = \int \underbrace{p(\boldsymbol{y}|\boldsymbol{x})}_{\text{Noise model}} \underbrace{p(\boldsymbol{x}|\Omega_y)}_{\text{Unobserved}} \mathrm{d}\boldsymbol{x}$$

$$\underbrace{p(\boldsymbol{x}|\boldsymbol{y},\Omega_y)}_{\text{Posterior}} \propto \underbrace{p(\boldsymbol{y}|\boldsymbol{x})}_{\text{Noise model}} \underbrace{p(\boldsymbol{x}|\Omega_y)}_{\text{Prior}}$$

High-Quality Self-Supervised Deep Image Denoising, NIPS 2019

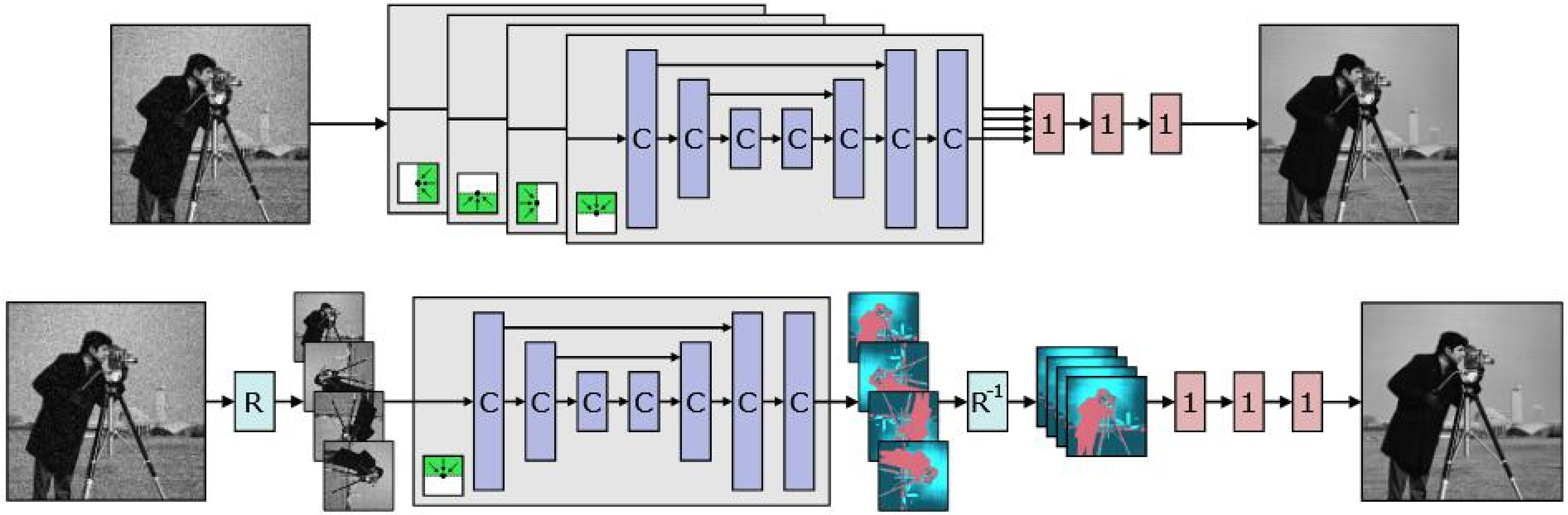


Figure 1: **Top:** In our blind-spot network architecture, we effectively construct four denoiser network branches, each having its receptive field restricted to a different direction. A single-pixel offset at the end of each branch separates the receptive field from the center pixel. The results are then combined by 1×1 convolutions. **Bottom:** In practice, we run four rotated versions of each input image through a single receptive field -restricted branch, yielding a simpler architecture that performs the same function. This also implicitly shares the convolution kernels between the branches and thus avoids the four-fold increase in the number of trainable weights.

Noise2Noise

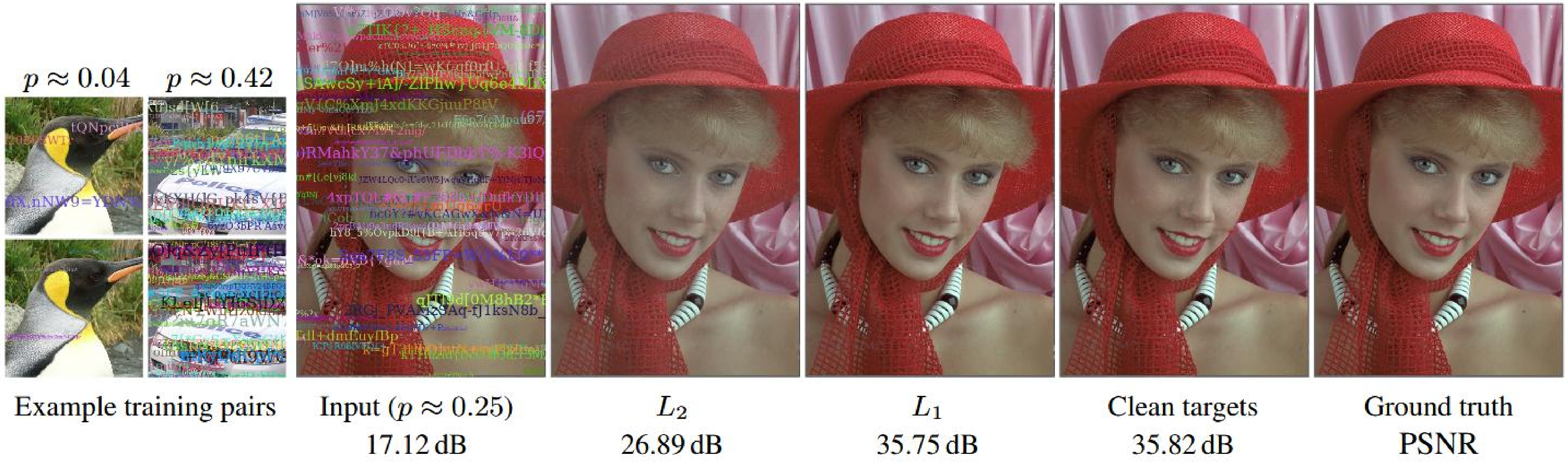
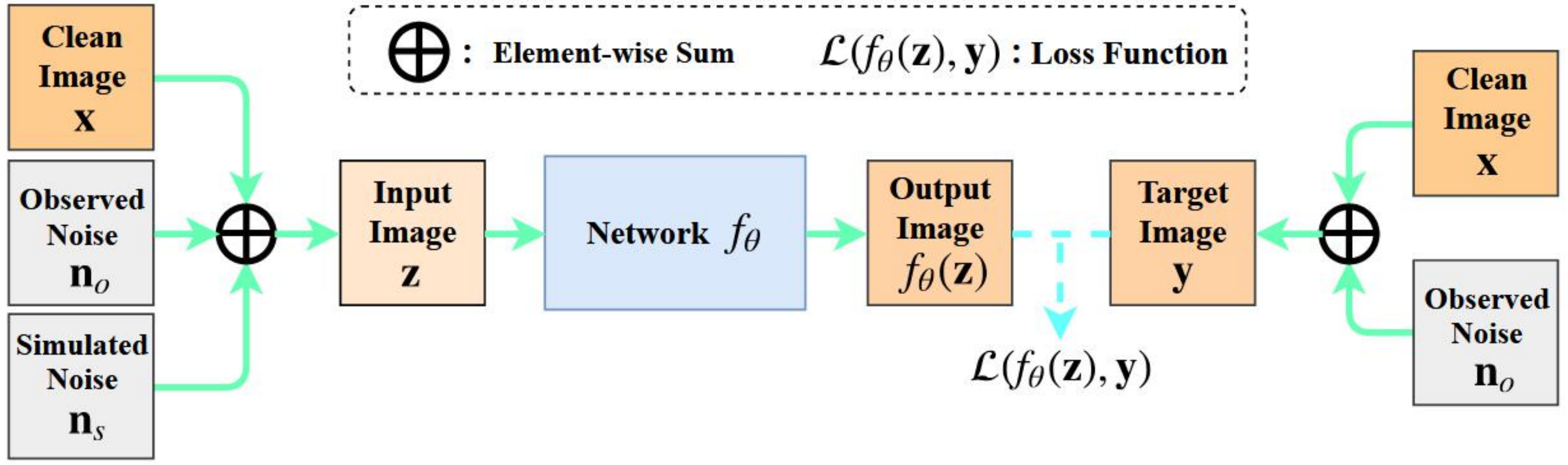


Figure 3. Removing random text overlays corresponds to seeking the median pixel color, accomplished using the L_1 loss. The mean (L_2 loss) is not the correct answer: note shift towards mean text color. Only corrupted images shown during training.

Noise as clean

- Double noise corruption



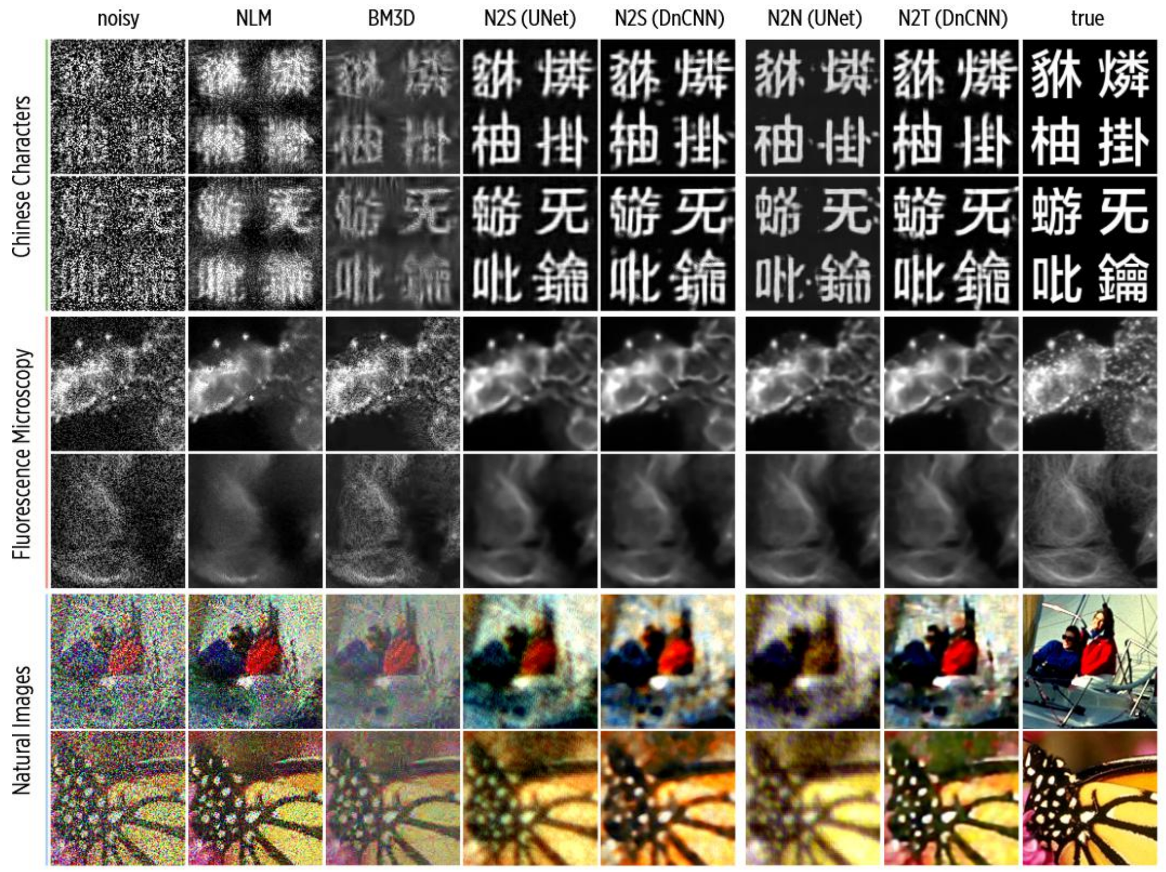
Clean Image $\mathbf{x}$
Observed Noise $\mathbf{n}_o$
Simulated Noise $\mathbf{n}_s$
$\oplus$: Element-wise Sum
$\mathcal{L}(f_\theta(\mathbf{z}), \mathbf{y})$: Loss Function
Input Image $\mathbf{z}$
Network f_θ
Output Image $f_\theta(\mathbf{z})$
Target Image $\mathbf{y}$
$\mathcal{L}(f_\theta(\mathbf{z}), \mathbf{y})$
Clean Image $\mathbf{x}$
Observed Noise $\mathbf{n}_o$

Fig. 2. **Proposed "Noisy-As-Clean" strategy for training self-supervised image denoising networks.** In our NAC strategy, we take the *observed* noisy image $\mathbf{y} = \mathbf{x} + \mathbf{n}_o$ as the "clean" target, and take the *simulated* noisy image $\mathbf{z} = \mathbf{y} + \mathbf{n}_s$ as the input. We do not regard the clean image $\mathbf{x}$ as target. After training, the inference is performed on the target noisy image $\mathbf{y} = \mathbf{x} + \mathbf{n}_o$.

Noisy-As-Clean: Learning Self-supervised Denoising from the Corrupted Image, 2020, May

Noise2Self

- Assuming noise at different dimensions are independent



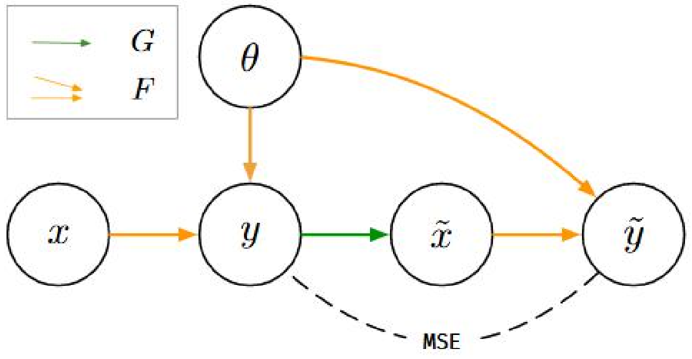
Batson J, Royer L. Noise2self: Blind denoising by self-supervision[J]. arXiv preprint arXiv:1901.11365, 2019.

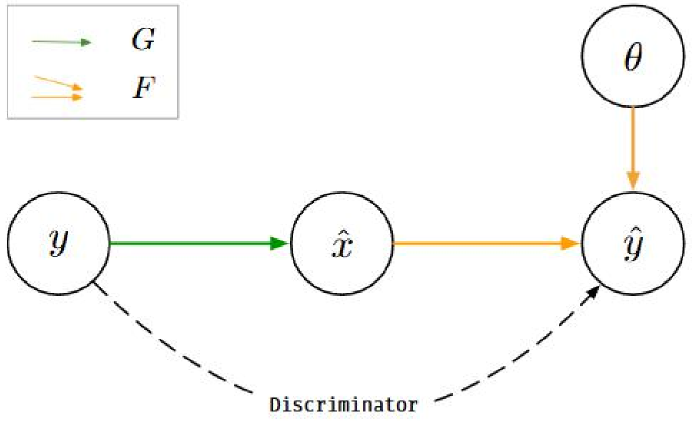
Denoise by corruption prior

$$Y = F(X;\Theta) + \mathcal{E}$$

$$x^* = \arg\max_x \log \mathrm{p}_{Y|X}(y|x) + \log \mathrm{p}_X(x)$$

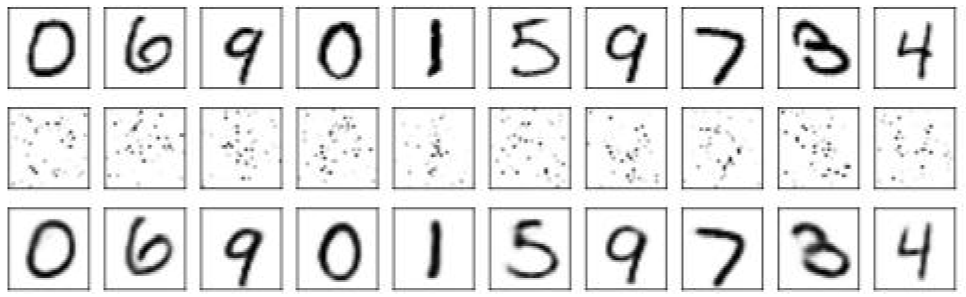
$$G^* = \arg\max_G \mathbb{E}_{\mathrm{p}_Y}\left\{\log \mathrm{p}_{Y|X}(y|G(y)) + \log \mathrm{p}_X(G(y))\right\}$$





Pajot et al., UNSUPERVISEDADVERSARIALIMAGERECONSTRUCTION, ICLR 2019.

SGD via Prior net


(a) We show original images (top row) and reconstructions by Lasso (middle row) and our algorithm (bottom row).

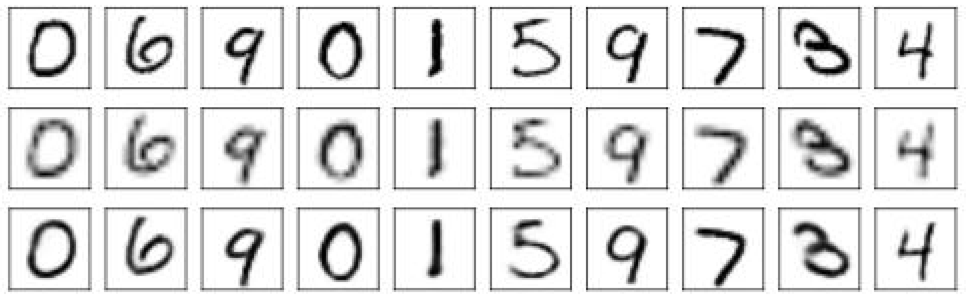

(b) We show original images (top row), low resolution version of original images (middle row) and reconstructions (last row).

Figure 2: Results on MNIST. Reconstruction with 100 measurements (left) and Super-resolution (right)

Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G Dimakis. Compressed sensing using generative models.arXiv preprint arXiv:1703.03208, 2017.

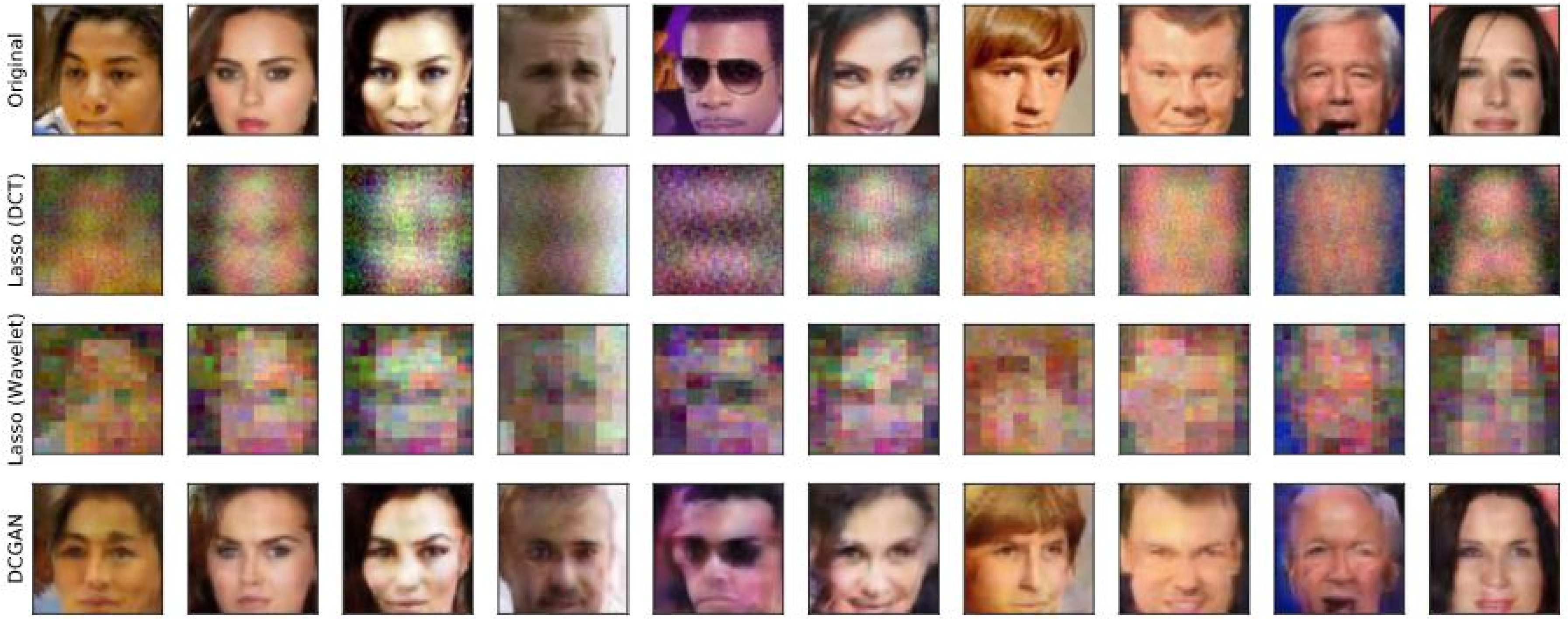
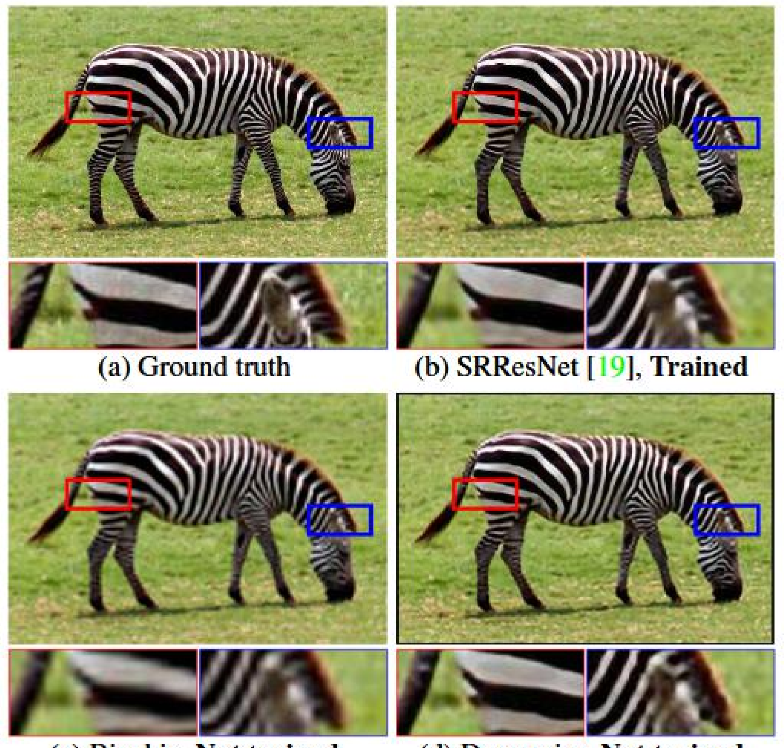


Figure 3: Reconstruction results on celebA with $m = 500$ measurements (of $n = 12288$ dimensional vector). We show original images (top row), and reconstructions by Lasso with DCT basis (second row), Lasso with wavelet basis (third row), and our algorithm (last row).

Deep image prior



(a) Ground truth (b) SRResNet [19], **Trained**

(c) Bicubic, **Not trained** (d) Deep prior, **Not trained**

Figure 1: **Super-resolution using the deep image prior.** Our method uses a randomly-initialized ConvNet to upsample an image, using its structure as an image prior; similar to bicubic upsampling, this method does not require learning, but produces much cleaner results with sharper edges. In fact, our results are quite close to state-of-the-art super-resolution methods that use ConvNets learned from large datasets. The deep image prior works well for all inverse problems we could test.

Ulyanov D, Vedaldi A, Lempitsky V. Deep image prior[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 9446-9454.

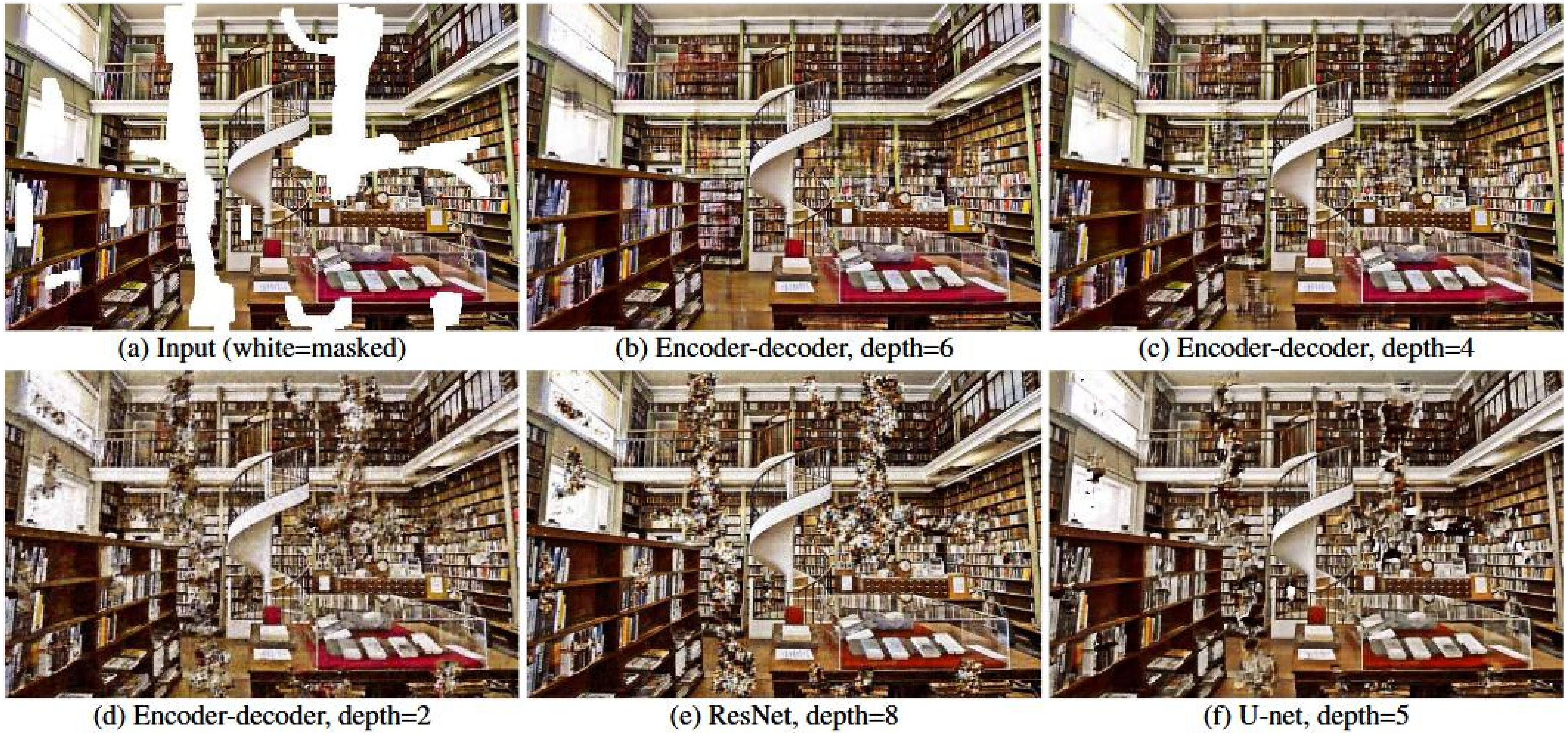
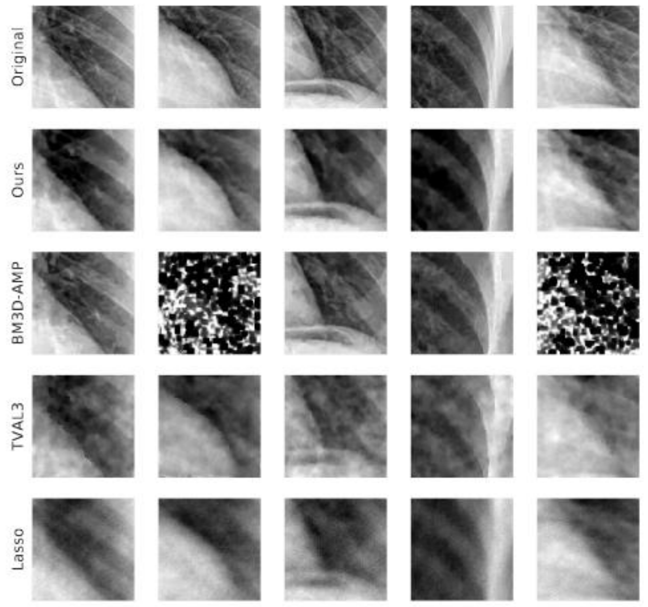
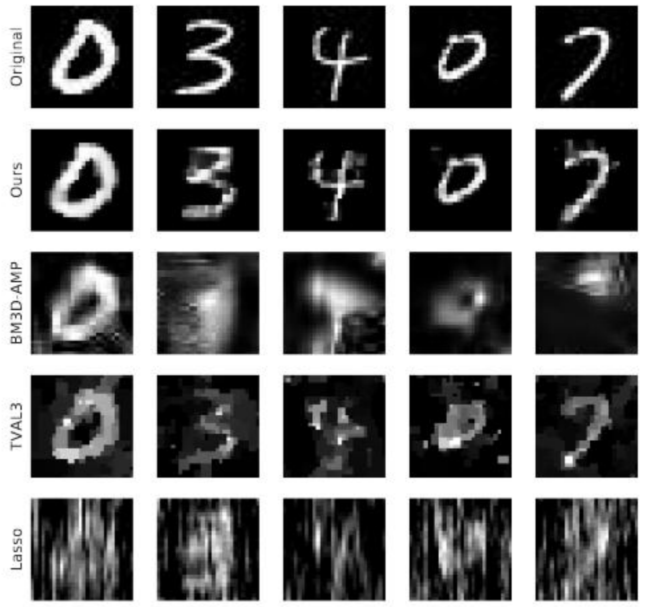

(a) Input (white=masked)
(b) Encoder-decoder, depth=6
(c) Encoder-decoder, depth=4
(d) Encoder-decoder, depth=2
(e) ResNet, depth=8
(f) U-net, depth=5

Figure 8: **Inpainting using different depths and architectures.** The figure shows that much better inpainting results can be obtained by using deeper random networks. However, adding skip connections to ResNet in U-Net is highly detrimental.

Regularization for DIP



(a) Reconstructions - Chest X-ray

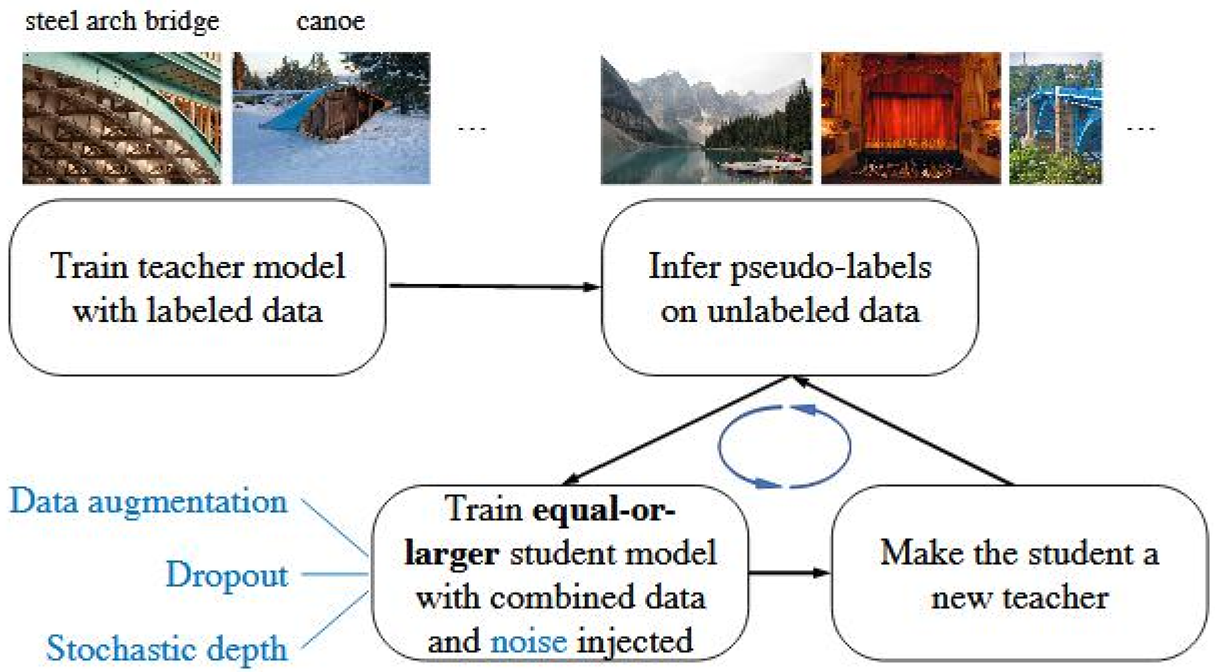


(b) Reconstructions - MNIST

Figure 2: Reconstruction results on x-ray images for m = 2000 measurements (of n = 65536 pixels) and MNIST for m = 75 measurements (of n = 784 pixels). From top to bottom row: original image, reconstructions by our algorithm, then reconstructions by baselines BM3D-AMP, TVAL3, and Lasso. For x-ray images the number of measurements obtained are 3% the number of pixels (i.e. $\frac{m}{n} = .03$), for which BM3D-AMP often fails to converge.

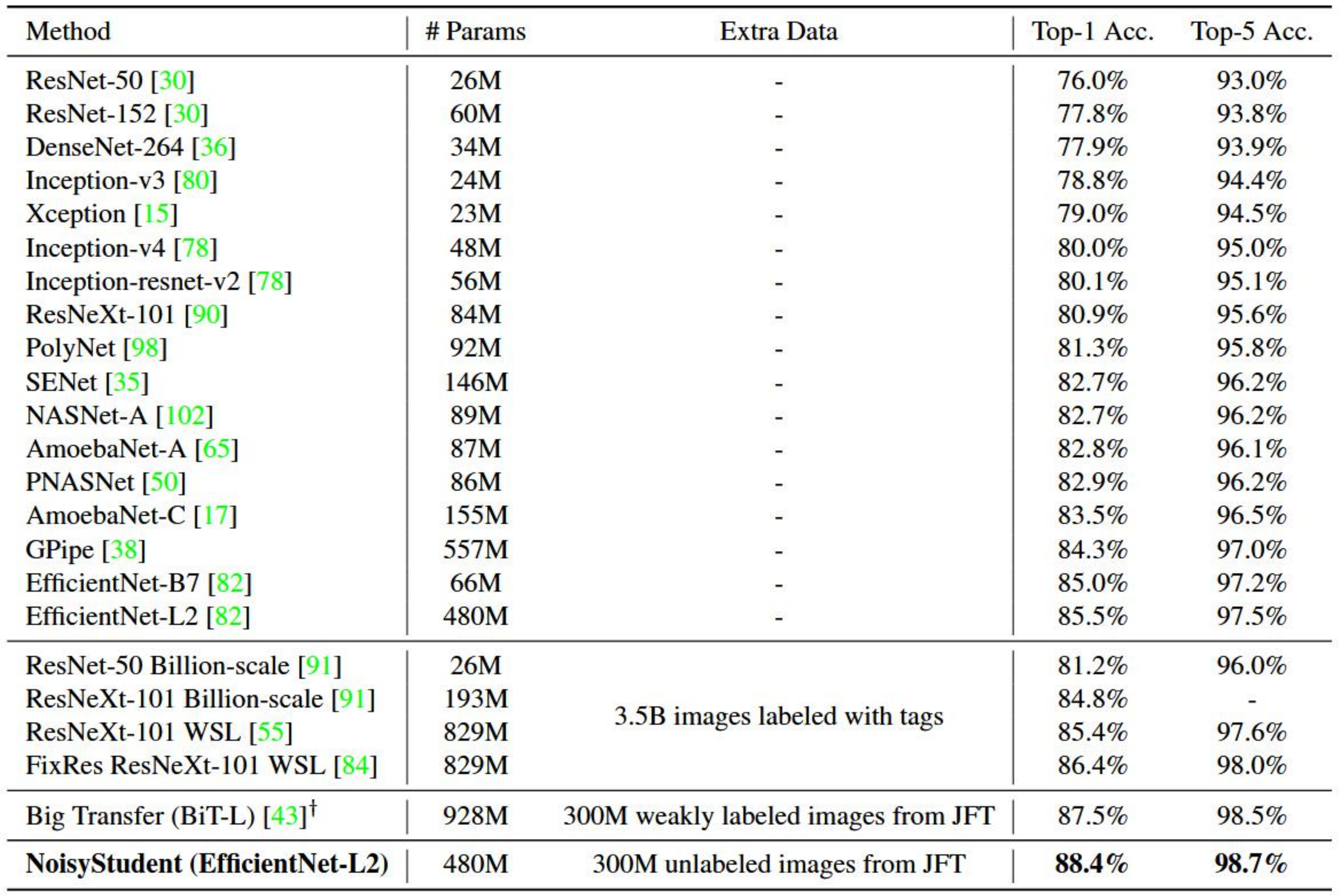
Veen et al., Compressed Sensing with Deep Image Prior andLearned Regularization, 2019.

Noisy student with self-learning


steel arch bridge
canoe
...
...
Train teacher model with labeled data
Infer pseudo-labels on unlabeled data
Data augmentation
Dropout
Stochastic depth
Train **equal-or-larger** student model with combined data and noise injected
Make the student a new teacher

Self-training with Noisy Student improves ImageNet classification, CVPR 2020.

Noisy student with self-learning



Method	# Params	Extra Data	Top-1 Acc.	Top-5 Acc.
ResNet-50 [30]	26M	-	76.0%	93.0%
ResNet-152 [30]	60M	-	77.8%	93.8%
DenseNet-264 [36]	34M	-	77.9%	93.9%
Inception-v3 [80]	24M	-	78.8%	94.4%
Xception [15]	23M	-	79.0%	94.5%
Inception-v4 [78]	48M	-	80.0%	95.0%
Inception-resnet-v2 [78]	56M	-	80.1%	95.1%
ResNeXt-101 [90]	84M	-	80.9%	95.6%
PolyNet [98]	92M	-	81.3%	95.8%
SENet [35]	146M	-	82.7%	96.2%
NASNet-A [102]	89M	-	82.7%	96.2%
AmoebaNet-A [65]	87M	-	82.8%	96.1%
PNASNet [50]	86M	-	82.9%	96.2%
AmoebaNet-C [17]	155M	-	83.5%	96.5%
GPipe [38]	557M	-	84.3%	97.0%
EfficientNet-B7 [82]	66M	-	85.0%	97.2%
EfficientNet-L2 [82]	480M	-	85.5%	97.5%
ResNet-50 Billion-scale [91]	26M	3.5B images labeled with tags	81.2%	96.0%
ResNeXt-101 Billion-scale [91]	193M		84.8%	-
ResNeXt-101 WSL [55]	829M		85.4%	97.6%
FixRes ResNeXt-101 WSL [84]	829M		86.4%	98.0%
Big Transfer (BiT-L) [43][†]	928M	300M weakly labeled images from JFT	87.5%	98.5%
NoisyStudent (EfficientNet-L2)	480M	300M unlabeled images from JFT	**88.4%**	**98.7%**

Self-training is better than pretraining

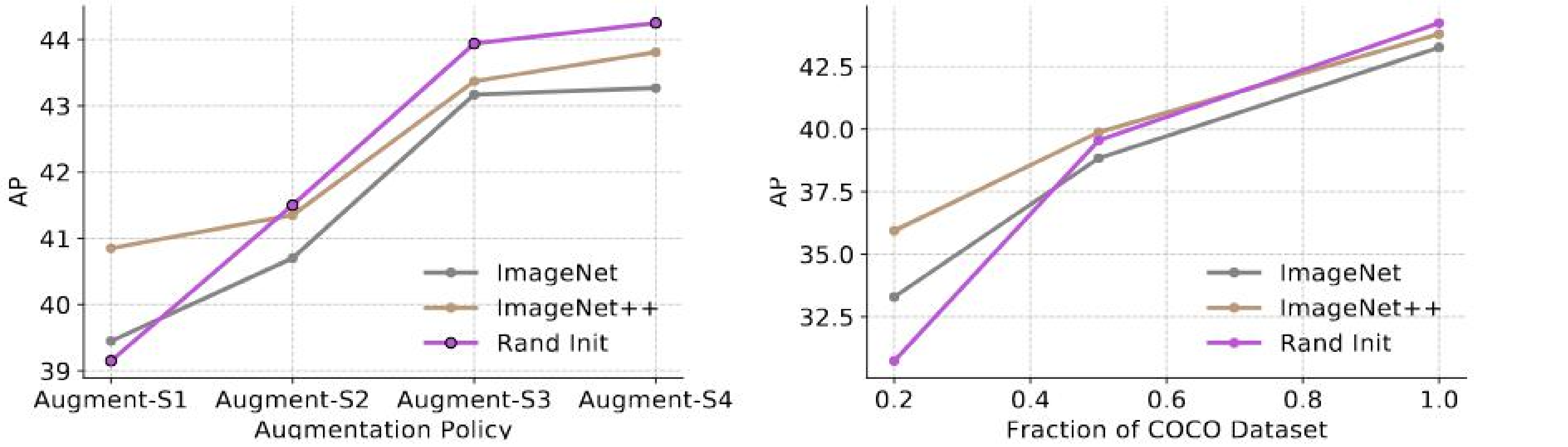
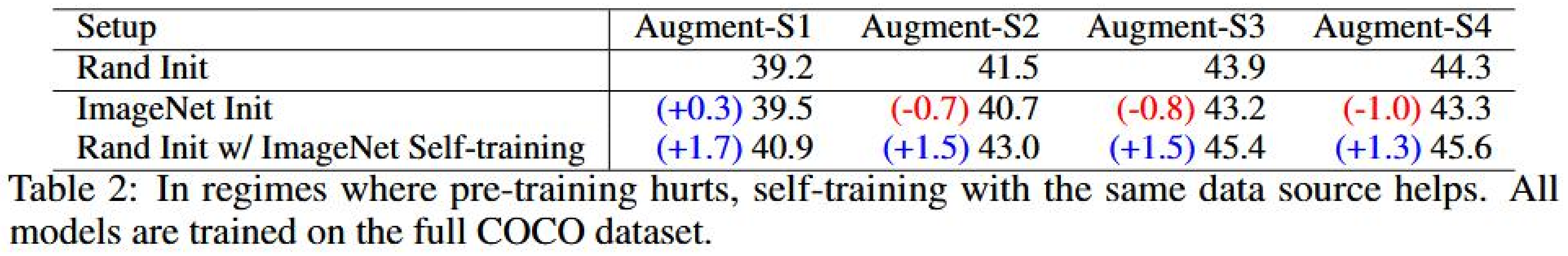


Figure 1: The effects of data augmentation and dataset size on pre-training. **Left**: Supervised object detection performance under various ImageNet pre-trained checkpoint qualities and data augmentation strengths on COCO. **Right**: Supervised object detection performance under various COCO dataset sizes and ImageNet pre-trained checkpoint qualities. All models use Augment-S4 (for similar results with other augmentation methods see Appendix C).

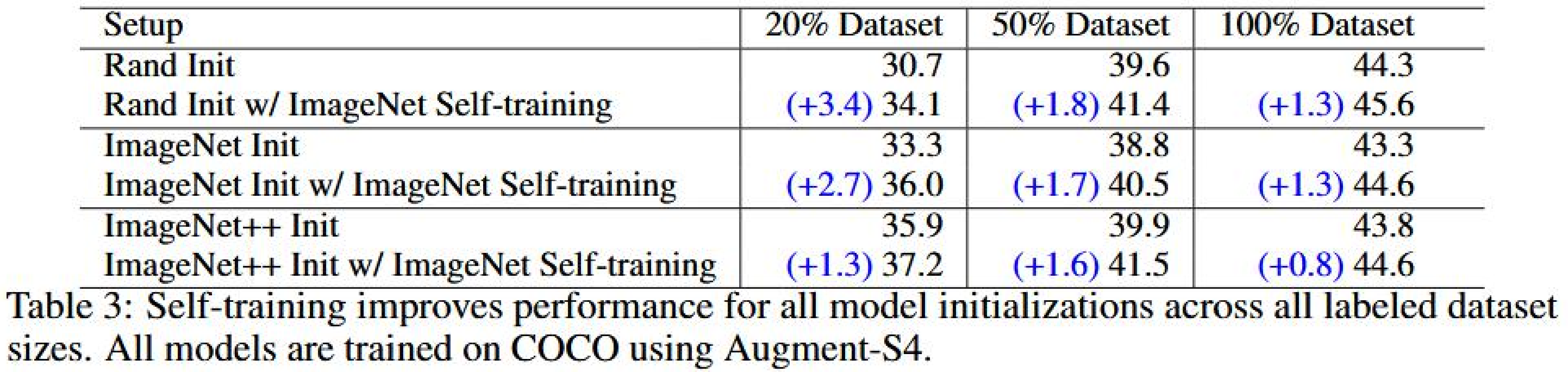
Barret Zoph∗ , Golnaz Ghiasi∗ , Tsung-Yi Lin∗ ,Yin Cui, Hanxiao Liu, Ekin Cubuk, Quoc V. Le, Rethinking Pre-training and Self-training, Google Brain, 2020.

Self-training is better than pretraining



Setup	Augment-S1	Augment-S2	Augment-S3	Augment-S4
Rand Init	39.2	41.5	43.9	44.3
ImageNet Init	(+0.3) 39.5	(-0.7) 40.7	(-0.8) 43.2	(-1.0) 43.3
Rand Init w/ ImageNet Self-training	(+1.7) 40.9	(+1.5) 43.0	(+1.5) 45.4	(+1.3) 45.6

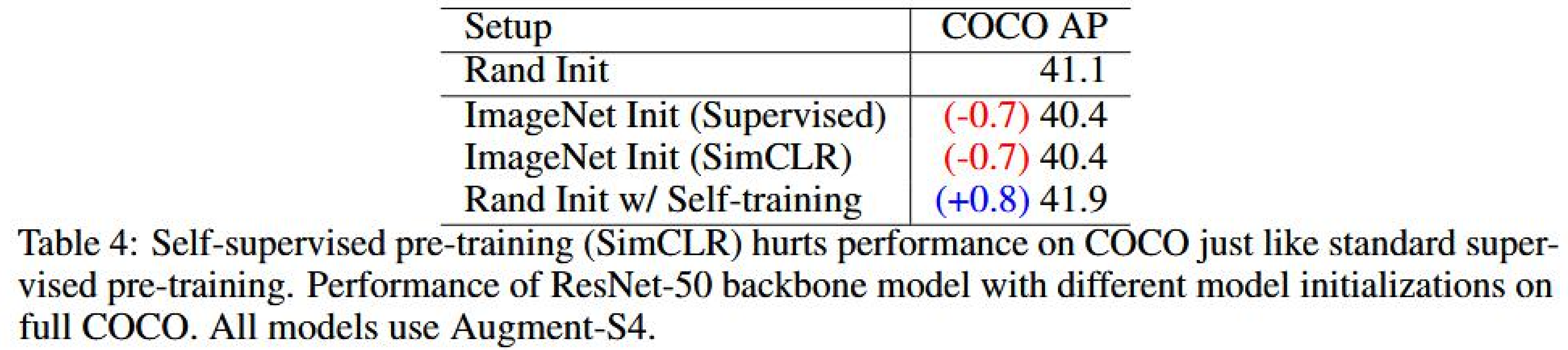
Table 2: In regimes where pre-training hurts, self-training with the same data source helps. All models are trained on the full COCO dataset.



Setup	20% Dataset	50% Dataset	100% Dataset
Rand Init	30.7	39.6	44.3
Rand Init w/ ImageNet Self-training	(+3.4) 34.1	(+1.8) 41.4	(+1.3) 45.6
ImageNet Init	33.3	38.8	43.3
ImageNet Init w/ ImageNet Self-training	(+2.7) 36.0	(+1.7) 40.5	(+1.3) 44.6
ImageNet++ Init	35.9	39.9	43.8
ImageNet++ Init w/ ImageNet Self-training	(+1.3) 37.2	(+1.6) 41.5	(+0.8) 44.6

Table 3: Self-training improves performance for all model initializations across all labeled dataset sizes. All models are trained on COCO using Augment-S4.

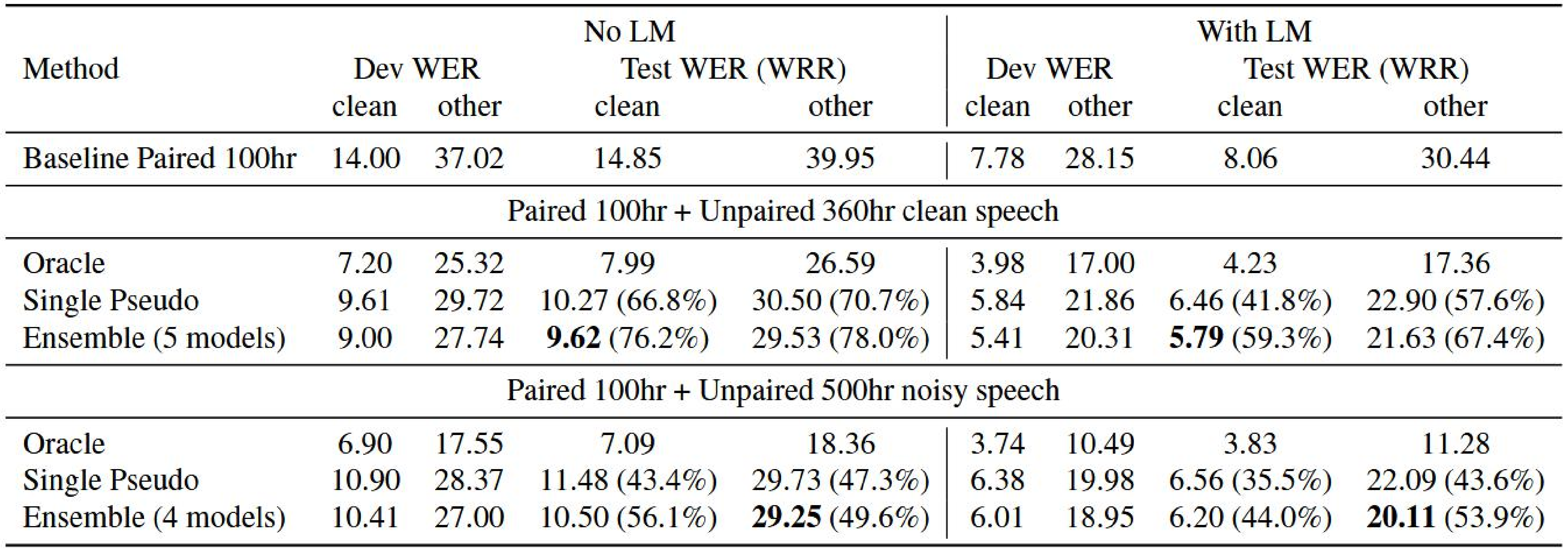
Self-training is better than pretraining



Setup	COCO AP
Rand Init	41.1
ImageNet Init (Supervised)	(-0.7) 40.4
ImageNet Init (SimCLR)	(-0.7) 40.4
Rand Init w/ Self-training	(+0.8) 41.9

Table 4: Self-supervised pre-training (SimCLR) hurts performance on COCO just like standard supervised pre-training. Performance of ResNet-50 backbone model with different model initializations on full COCO. All models use Augment-S4.

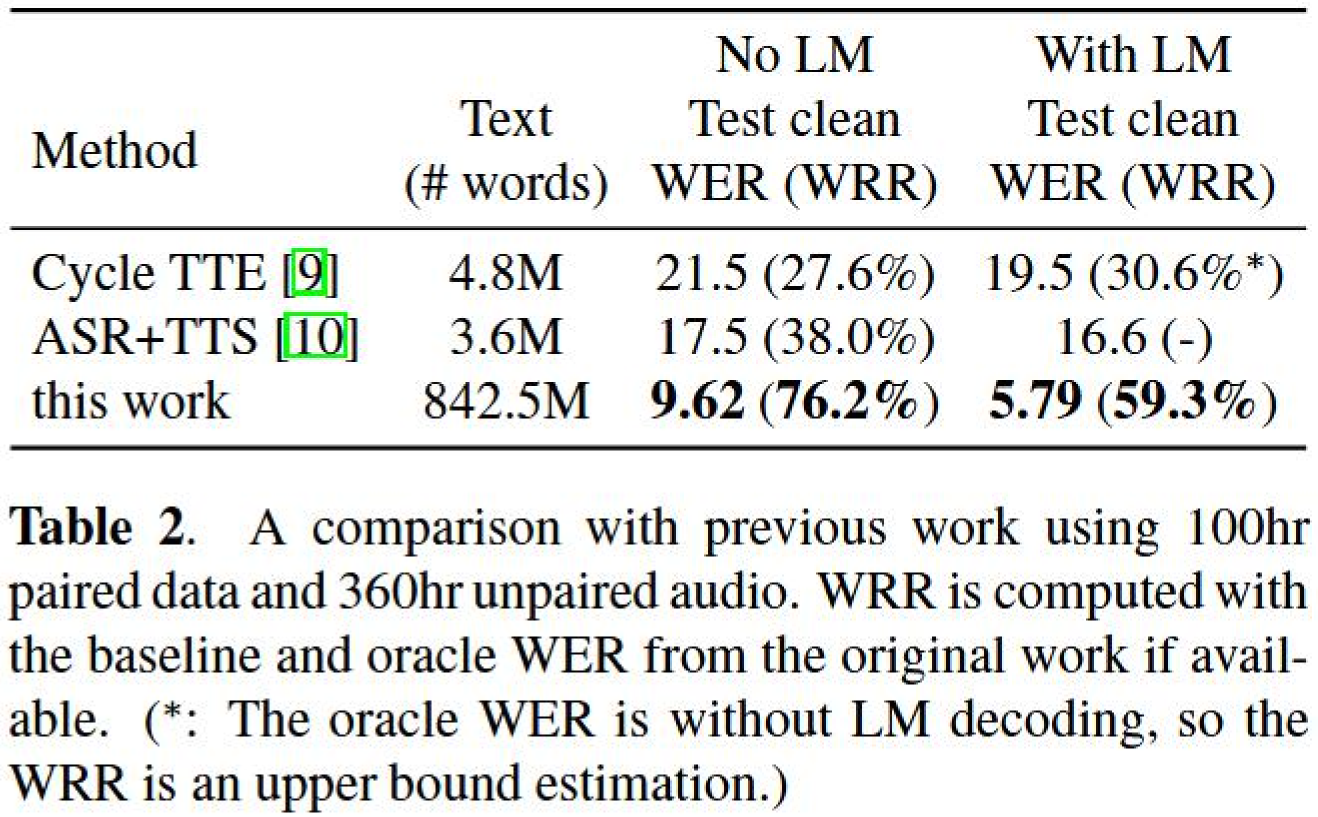
Speech recognition with self-training



Method	No LM				With LM			
	Dev WER		Test WER (WRR)		Dev WER		Test WER (WRR)	
	clean	other	clean	other	clean	other	clean	other
Baseline Paired 100hr	14.00	37.02	14.85	39.95	7.78	28.15	8.06	30.44
Paired 100hr + Unpaired 360hr clean speech								
Oracle	7.20	25.32	7.99	26.59	3.98	17.00	4.23	17.36
Single Pseudo	9.61	29.72	10.27 (66.8%)	30.50 (70.7%)	5.84	21.86	6.46 (41.8%)	22.90 (57.6%)
Ensemble (5 models)	9.00	27.74	**9.62** (76.2%)	29.53 (78.0%)	5.41	20.31	**5.79** (59.3%)	21.63 (67.4%)
Paired 100hr + Unpaired 500hr noisy speech								
Oracle	6.90	17.55	7.09	18.36	3.74	10.49	3.83	11.28
Single Pseudo	10.90	28.37	11.48 (43.4%)	29.73 (47.3%)	6.38	19.98	6.56 (35.5%)	22.09 (43.6%)
Ensemble (4 models)	10.41	27.00	10.50 (56.1%)	**29.25** (49.6%)	6.01	18.95	6.20 (44.0%)	**20.11** (53.9%)

Table 1. Best results from single runs tuned on the dev sets. The best filtering setup found in Section 4.3.1 is applied.

Speech recognition with self-training



Method	Text (# words)	No LM Test clean WER (WRR)	With LM Test clean WER (WRR)
Cycle TTE [9]	4.8M	21.5 (27.6%)	19.5 (30.6%*)
ASR+TTS [10]	3.6M	17.5 (38.0%)	16.6 (-)
this work	842.5M	**9.62 (76.2%)**	**5.79 (59.3%)**

Table 2. A comparison with previous work using 100hr paired data and 360hr unpaired audio. WRR is computed with the baseline and oracle WER from the original work if available. (*: The oracle WER is without LM decoding, so the WRR is an upper bound estimation.)

Jacob Kahn, Ann Lee, and Awni Hannun. Self-training for end-to-end speech recognition. FaceBook, CASSP,2019.