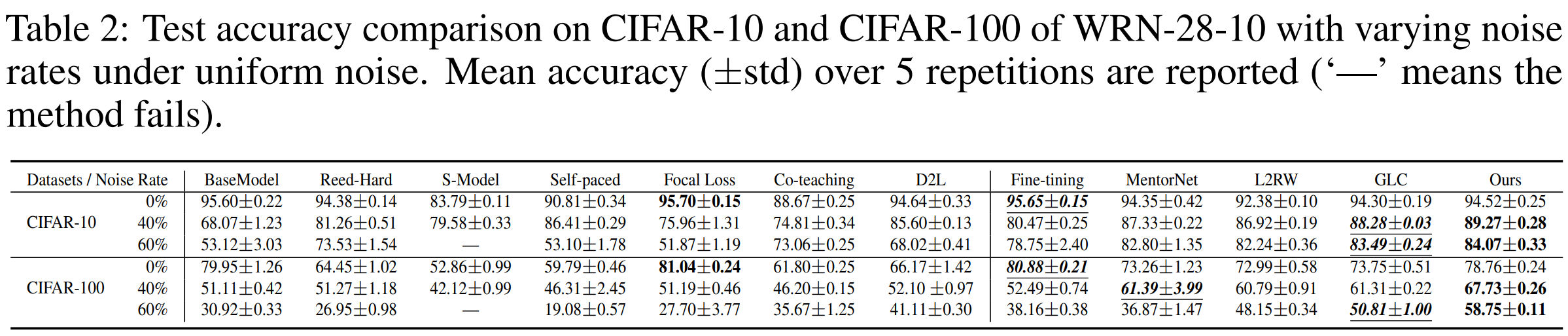
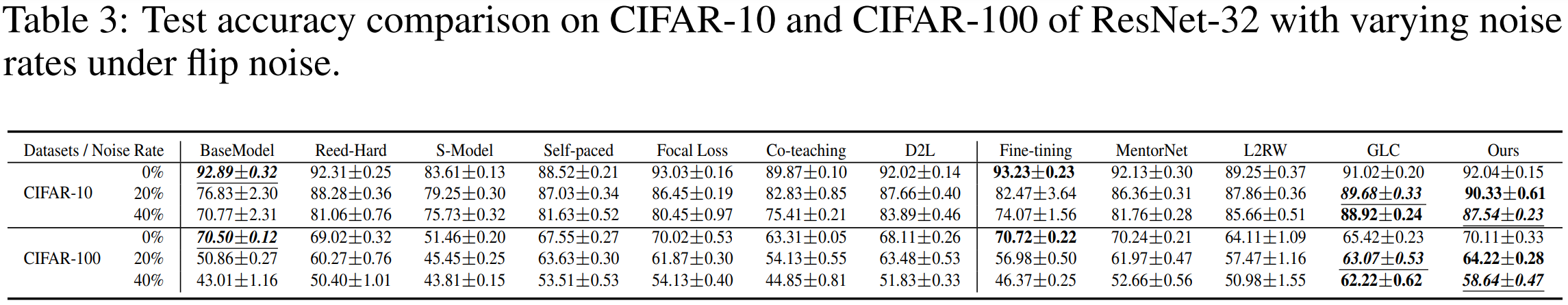
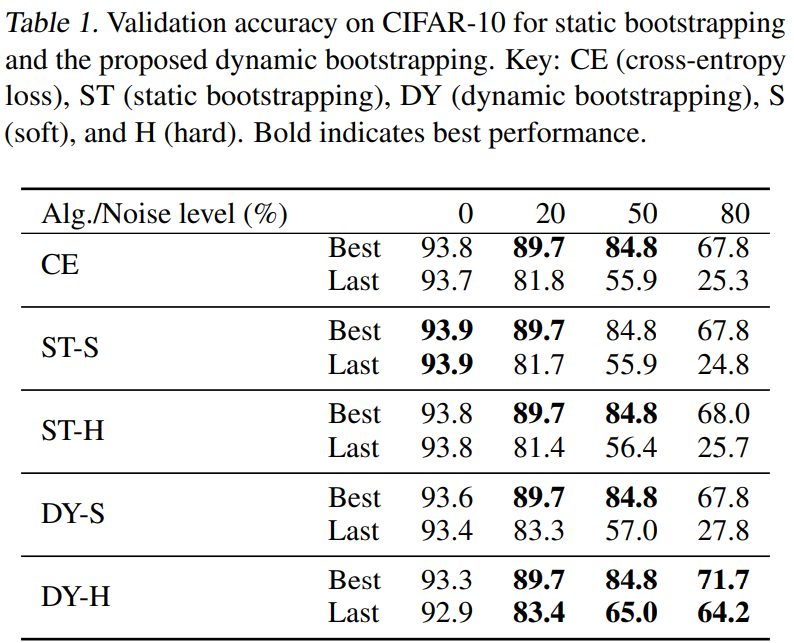
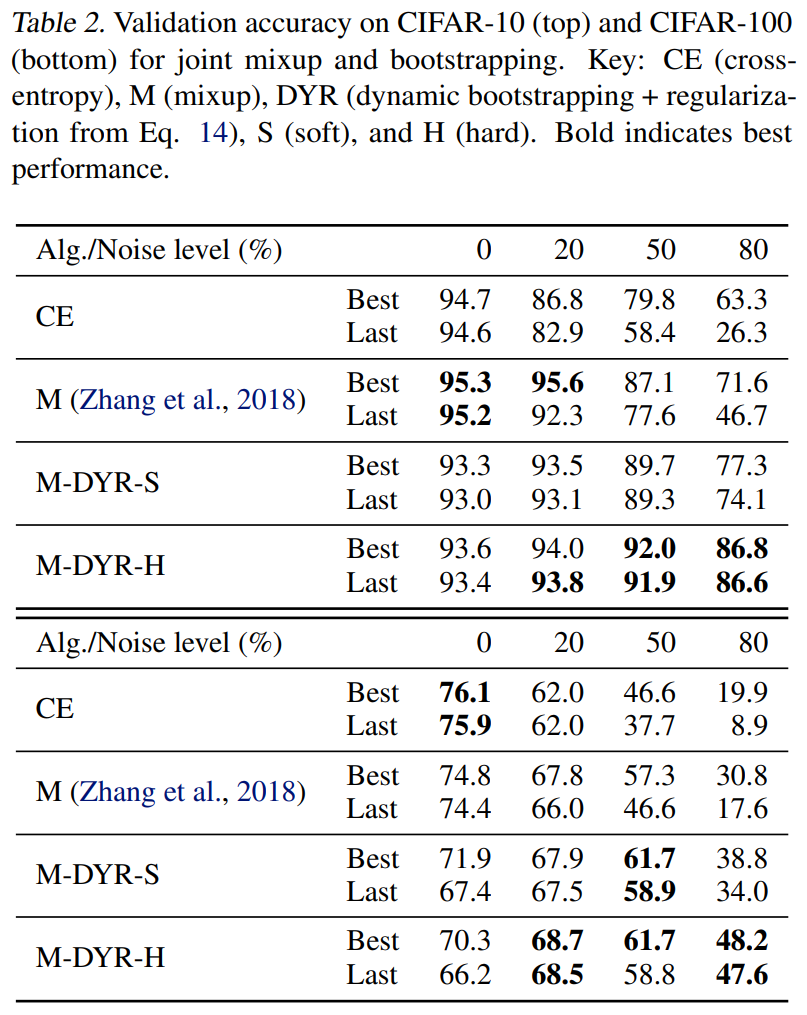
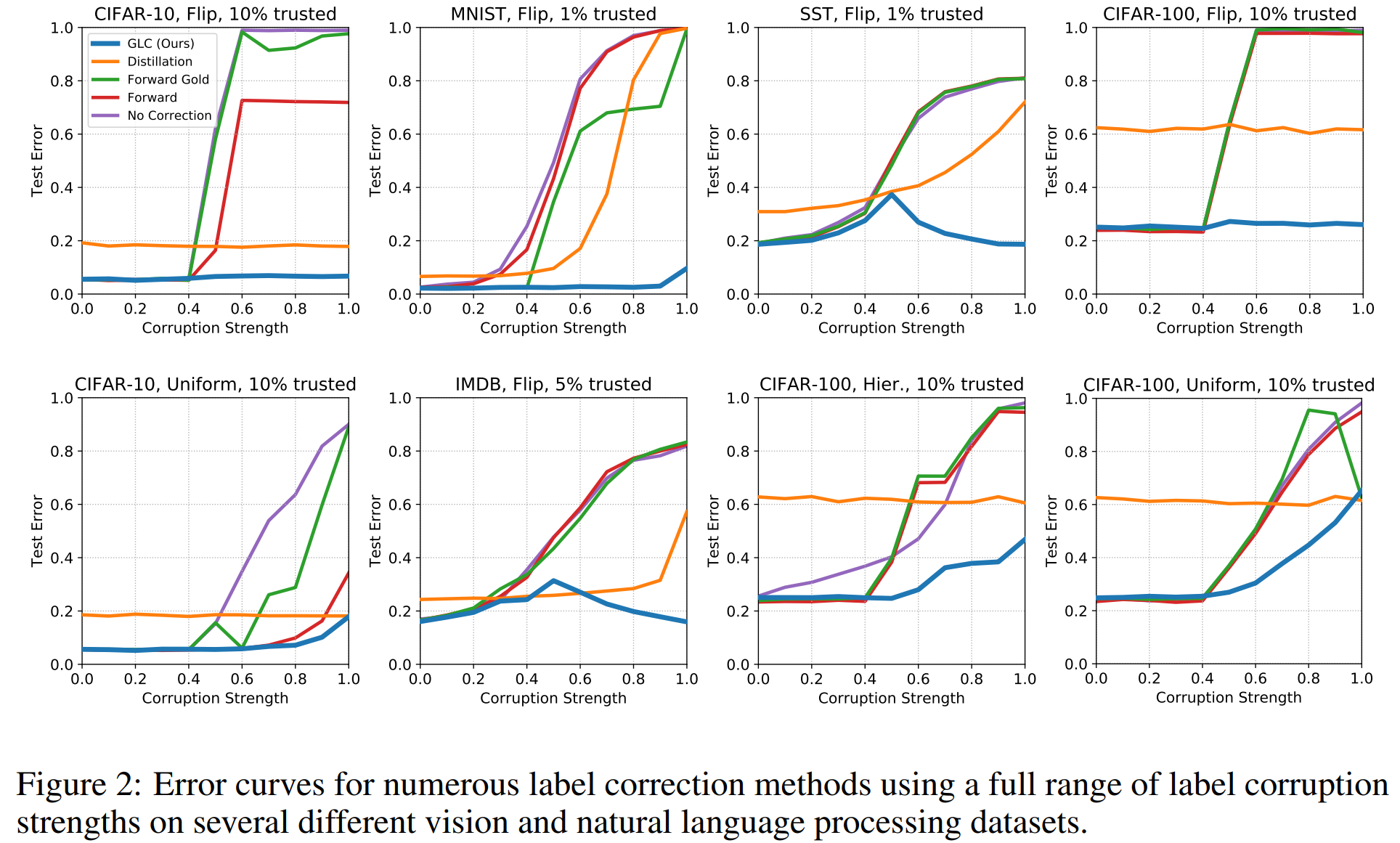
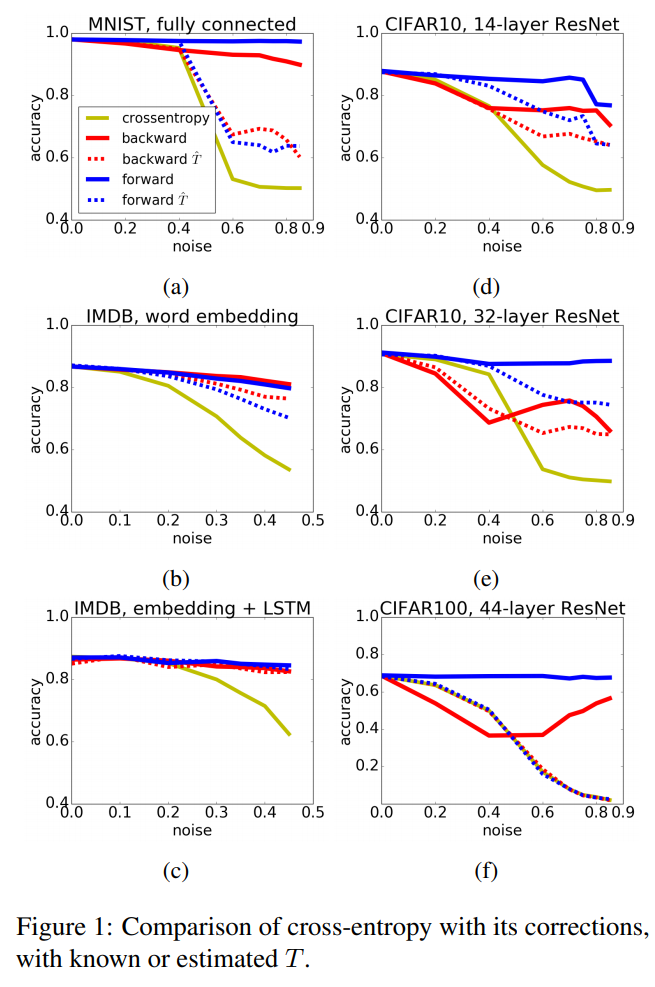
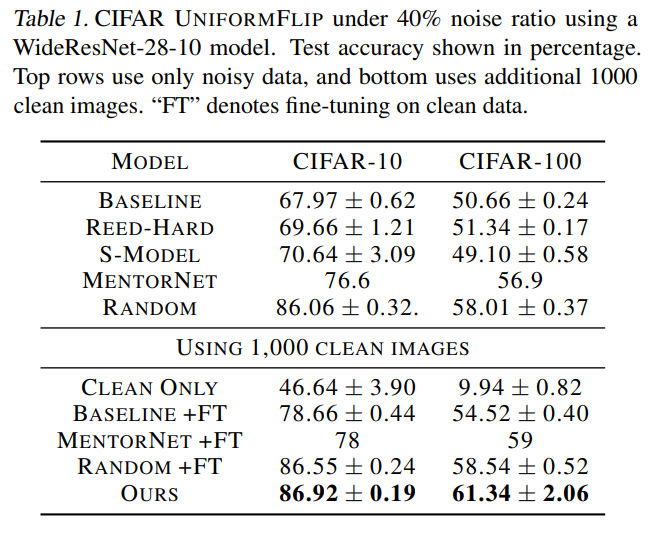
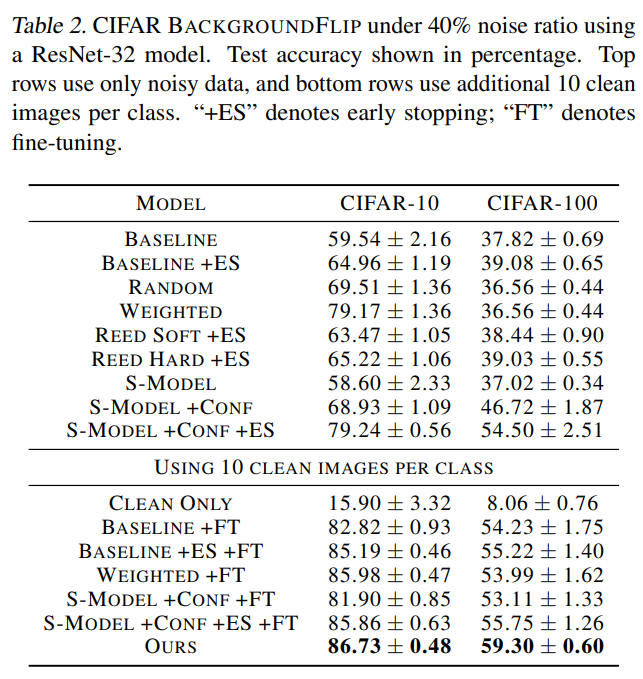
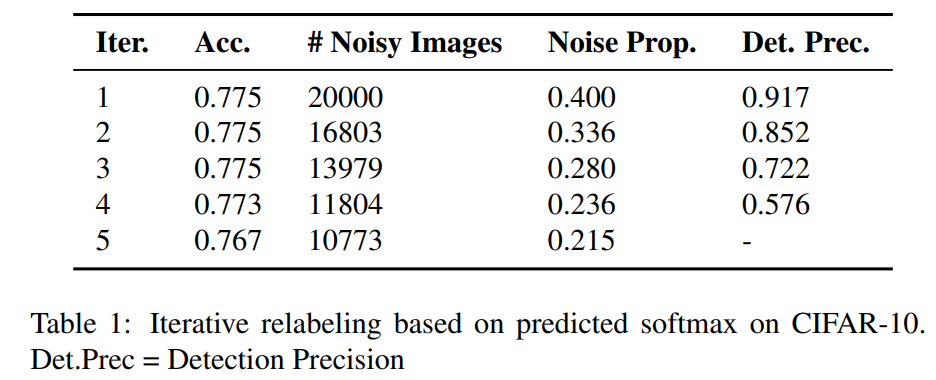
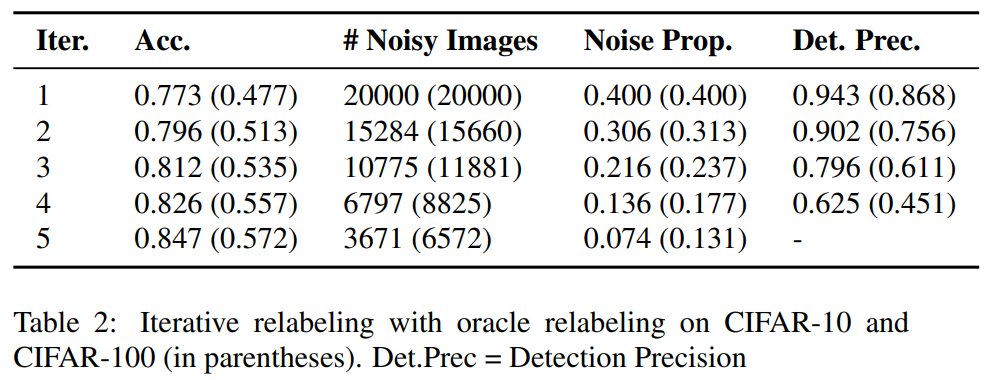
## Loss related reweighting

1. Meta-Weight-Net: Learning an Explicit Mapping For Sample Weighting (2019 NIPS)
   1. Reference code: <https://github.com/xjtushujun/meta-weight-net>(pytorch)
   2. 训练准备:
      1. 数据集处理
         1. Imbalanced dataset实验暂时不考虑
         2. Noised dataset: 待复现
   3. 训练超参
      1. Backbone:
         1. Uniform noise: WideResNet28-10
         2. Flip noise: ResNet32
      2. Optimizer:
         1. Optimizer: SGD
         2. Initial Lr: main network 0.1, meta hidden layer fixed 0.001
         3. Batch size: 100(32 for clothing 1M dataset)
         4. Weight decay: 0.0005
         5. Lr scheduler: (meta hidden layer fixed)
            1. Multistep factor=10, steps: 36, 38, total 40 for uniform
            2. Multistep factor=10, steps: 40, 50, total 60 for flip
      3. Dropout rate: 0
      4. repeat for result: 5
   4. Accuracy
      1. Uniform: 
      2. Flip: 
2. Unsupervised Label Noise Modeling and Loss Correction (2019 ICML)
   1. Reference code: <https://github.com/PaulAlbert31/LabelNoiseCorrection>(pytorch)
   2. 训练准备
      1. 图片处理：random horizontal flip, 32×32 random crops after zero padding with 4 pixels on each side, normalization
      2. 数据增强：mixup
   3. 训练超参
      1. Backbone: PreAct ResNet-18
      2. Optimizer:
         1. Optimizer: SGD
         2. Initial Lr: 0.1
         3. Batch size: 128
         4. Weight decay: 0.0001
         5. Momentum: 0.9
         6. Lr scheduler:
            1. Multistep factor=10, steps: 30, 80, 110 total 120 for no mixup, bootstrapping starts at 31 epoch
            2. Multistep factor=10, steps: 100, 250, total 300 for with mixup, bootstrapping starts at 106 epoch, mixup alpha=32
      3. Dropout: none
      4. BMM parameter: 10 EM iterations
   4. Accuracy
      1. Without mixup 
      2. With mixup 
3. Using Trusted Data to Train Deep Networks on Labels Corrupted by Severe Noise (2018 NIPS)
   1. Reference code: <https://github.com/mmazeika/glc>(pytorch)
   2. 训练准备
      1. 图片处理：random horizontal flip, 32×32 random crops after zero padding with 4 pixels on each side, normalization
   3. 训练超参
      1. Backbone: WideResNet40-2
      2. Optimizer:
         1. Optimizer: SGD
         2. Initial Lr: 0.1
         3. Batch size: 128
         4. Weight decay: 0.0005
         5. Momentum: Nesterov 0.9
         6. Lr scheduler: Cosine lr schedular, with no SGDR(cyclic cosine lr schedular, or warm restart)
      3. Dropout rate: 0.3
      4. Gold fraction(clean validation set fraction): 0.1
   4. Accuracy 
4. Making Deep Neural Networks Robust to Label Noise: a Loss Correction Approach (2017 CVPR)
   1. Reference code: <https://github.com/GarrettLee/label_noise_correction>(unofficial tf)
   2. 训练准备
      1. 图片处理：random horizontal flip, 32×32 random crops after zero padding with 4 pixels on each side, normalization
   3. 训练超参
      1. Backbone: ResNet with “pre-activation”, depth 14 or 32 for CIFAR10, 44 for CIFAR100
      2. Optimizer: (all for CIFAR)
         1. Optimizer: SGD
         2. Initial Lr: 0.01
         3. Batch size: 128
         4. Weight decay: 0.0001
         5. Momentum: 0.9
         6. Lr scheduler:
            1. Multistep factor=10, steps: 40, 80, total 120 for CIFAR10
            2. Multistep factor=10, steps: 80, 120, total 150 for CIFAR100
      3. Dropout: None
      4. Validation set 10%
   4. Accuracy 

## Meta learning

1. Meta-Weight-Net: Learning an Explicit Mapping For Sample Weighting (2019 NIPS)
   1. 见上
2. Learning to Reweight Examples for Robust Deep Learning (2018 ICML)
   1. Reference code:
      1. <https://github.com/danieltan07/learning-to-reweight-examples>(unofficial pytorch)
      2. <https://github.com/uber-research/learning-to-reweight-examples>(official tf)
   2. 训练准备
      1. 图片处理：random horizontal flip, 32×32 random crops after zero padding with 4 pixels on each side, normalization
   3. 训练超参
      1. Backbone:
         1. Uniform: WideResNet28-10
         2. BackGroundFlip: ResNet32
      2. Optimizer: (all for CIFAR)
         1. Optimizer: SGD
         2. Initial Lr: 0.1
         3. Batch size: 100
         4. Momentum: 0.9
         5. Lr scheduler:
            1. Multistep factor=10, steps: 40K step, 60K step, total 80K step for ResNet32
            2. Multistep factor=10, steps: 40K step, 50K step, total 60K step for WRN and early stop version ResNet32
      3. Dropout rate: 0.3
      4. Validation set: clean, for meta data
         1. Uniform: 1000
         2. BackgroundFlip: 10 each class
      5. Hyper Validation set: also corrupted just checking the accuracy during training
   4. Accuracy
      1. Uniform
      2. BackGroundFlip

## Uncertainty related reweighting

1. Uncertainty Based Detection and Relabeling of Noisy Image Labels (2019)
   1. Reference code: None
   2. 训练准备
      1. No code
   3. 训练超参
      1. Backbone:
         1. Simple CNN network
      2. Optimizer: no code
      3. Dropout rate: no code
      4. Relabeling ratio: 10% of the sample with top uncertainty
   4. Accuracy
      1. Auto relabel using model in former epochs: not helping 
      2. Oracle relabel(all relabel are correct) is helping 
2. Deep Bayesian self training (2019)
   1. Reference code: <https://github.com/fabio-deep/Deep-Bayesian-Self-Training>(keras and tf)
   2. 训练准备: no CIFAR experiments
   3. 训练超参: no CIFAR experiments
   4. Accuracy: no CIFAR experiments
3. Proposed method