"How to measure the uncertainty"

↑ 사람 관점에서 X
모델 관점에서 O

Limitations of Active Learning

- Heuristic Approach
  - Highest entropy
  - Distance to Decision boundaries

) (-) Task specific design (only classification)

- Ensemble approach
  - 다양한 networks의 prediction의 variance ↗ 大 : uncertain ↘ 小 : certain

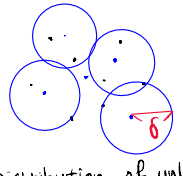
) (-) Not scale to large CNNs and data

- Bayesian Approach) (-) Not scale to large CNNs and data

- Distribution Approach
  : density based, diversity based
  unlabeled pool을 대표할 수 있는 sample을 가져오겠다.

) (-) Not considering hard examples

* Diversity : Core-set



$\{x\} = \min_{x} \delta$

Distribution of unlabeled pool

(+) can be task-agnostic as it only depends on feature space

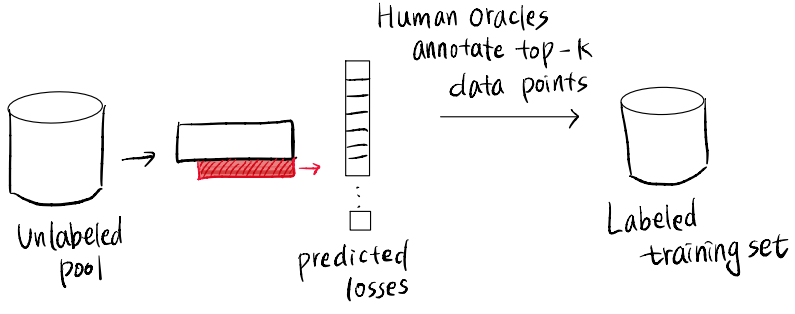
(-) not considering "hard" examples near the decision boundaries

(-) Expensive optimization for large pool

Active Learning : Our approach

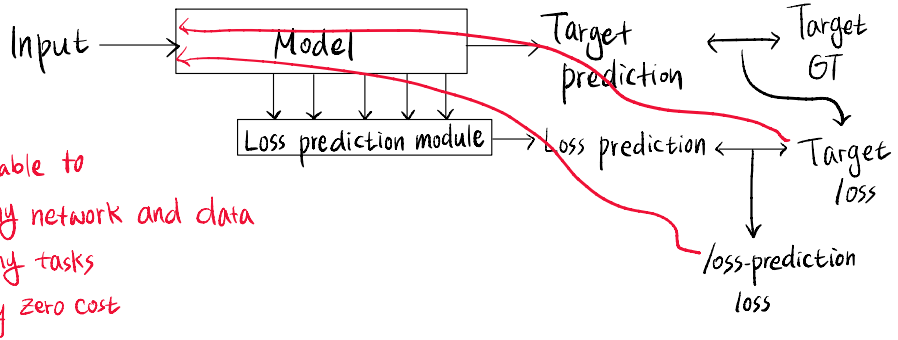
- Requirements :
  - Task-agnostic method
  - Not heuristic, learning based
  - Scalable to state-of-the-art networks and large data.

- Active learning by learning loss
  - Attach a "loss prediction module" to a target network
  - Learn the module to predict the loss



아무리 큰 모델이더라도, 목적 자체가 하나의 스칼라 값, loss를 최소화하는 것이기 때문에 loss가 모든 것을 담고 있을 것이다.

Active Learning by Learning Loss



(+) Applicable to
- any network and data
- any tasks

(+) Nearly zero cost

〈Notations〉

- x : Input
- $\hat{y}$: Target prediction
- y : Target ground truth
- $\hat{l}$: loss prediction
- l : Target loss
- $L_{loss}(\hat{l}, l)$: The loss for loss-prediction

Mean square error?

- The loss for loss-prediction $L_{loss}(\hat{l}, l)$

$$L_{loss}(\hat{l}, l) = \|\hat{l} - l\|^2$$

The scale changes...
학습할수록 크기가 작아짐
→ $\hat{l}$이 l의 scale을 따라가는 느낌

margin 이상으로 더 커야 penalty를 안 받음

To ignore scale changes of l, we use a ranking loss (pytorch에 내장)

$$L_{loss}(\hat{l}_i, \hat{l}_j, l_i, l_j) = \max(0, -1(l_i, l_j)\cdot(\hat{l}_i - \hat{l}_j) + \xi) \quad \text{margin} = 1$$

where $1(l_i, l_j) = \begin{cases} +1 & \text{if } l_i > l_j \\ -1 & \text{otherwise} \end{cases}$

i, j : 미니 배치 내에 있는 두 개의 샘플

a pair of predicted losses

a pair of real losses

"누가 더 크냐 / 작냐만 고려" 스케일 변화 고려 X

ex1) $l_i = 0.7$ $l_j = 0.5$ $\hat{l}_i = 0.5$ $\hat{l}_j = 0.7$

$$\mathcal{L} = \max(0, -1 \cdot (-0.2) + \xi^{1}) = 1.2$$

ex2) $l_i = 0.7$ $l_j = 0.5$ $\hat{l}_i = 0.8$ $\hat{l}_j = 0.3$

$$\mathcal{L} = \max(0, 1 \cdot (0.5) + \xi) = 1.5$$

· Given a mini-batch B, the total loss is defined as

$$\frac{1}{|B|}\sum_{(x,y)\in B} \mathcal{L}_{task}(\hat{y}, y) + \lambda \frac{1}{|B|}\sum_{(x_i,y_i,x_j,y_j)\in B} \mathcal{L}_{loss}(\hat{l}_i, \hat{l}_j, l_i, l_j)$$

Target Task

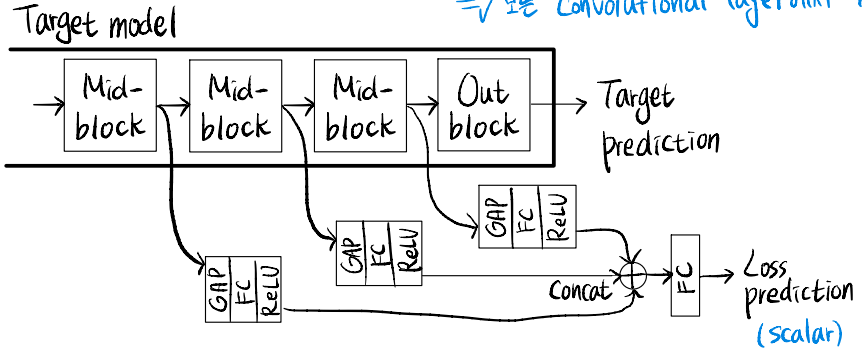
A pair (i, j) within a mini batch B

Loss prediction

where $l_i = \mathcal{L}_{task}(\hat{y}_i, y_i)$

· Loss prediction module

⇒ 모든 Convolutional layer에서 가져온 건 X

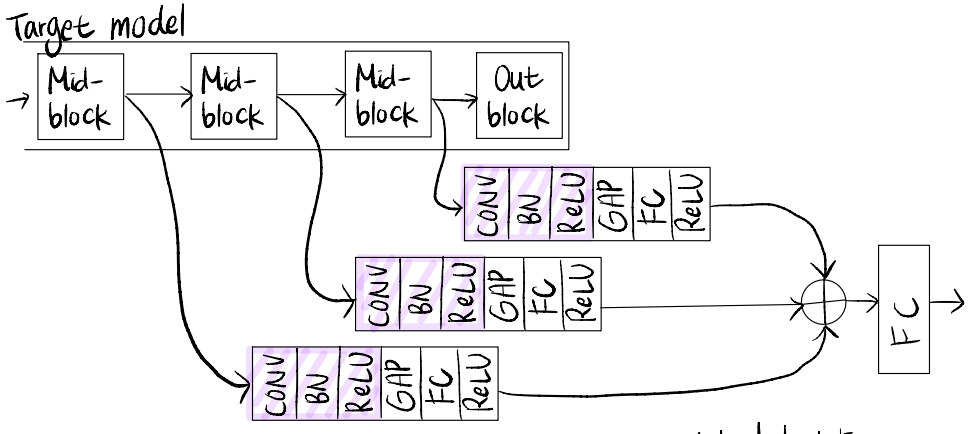


(+) Very efficient as GAP reduces the feature dimension

└ Global Average Pooling

(→ spatial dimension info ×)

· Experiment on indirect global average pooling



↓

×to1 ×

[Conclusion]

① Subsampling (diversity) → ② uncertainty

· Active Learning
  - ↳ works well with current deep networks
  - ↳ task-agnostic

· Verified with
  - ↳ three major visual recognition tasks
  - ↳ three popular network architectures

" Pick more important data,
and get better performance! "