半监督学习笔记

Wu, 2020/2/28

Entropy Regularization

In: Semi-Supervised Learning, pages 151–168, MIT Press, 2006.

文章从贝叶斯概率的角度来解释半监督学习。文章首先说明无标签的数据对于一个判别模型的最大似然估计是无信息的，因为最大似然估计是不能缺少标签的。但是无标签的数据包含信息是被广泛接受的，并且符合我们的尝试。作者认为，通过无标签的数据，我们只能学习先验分布的知识。作者使用的先验分布知识即无标签的样本是会按照其本来的类别聚集在一起，不同的样本族之间的概率密度就较低。文章中使用香农条件熵（ Shannon's conditional entropy）来衡量无标签样本交叉度，然后规范的目标就是要减少这个交叉度。（我的简单理解就是一个样本只有一个标签，所以我们可以设计一种标准（香农条件熵）来规范待学习的分布 $P(y|x)$, 使得其对于无标签样本的类别概率输出只有一个值较大而其它的值较小。）

Measure of Class Overlap

文中使用香农条件熵来衡量样本的类别交叉度，公式如下：

$$\begin{aligned} H(y|x,h=1) &= -\mathbf{E}_{xy}\left[\ln P(y|x,h=1)\right] \qquad (9.3) \\ &= -\int \sum_{m=1}^{M} \ln P(y=m|x,h=1) p(x,y=m|h=1)\ dx\ . \end{aligned}$$

其中，随机变量x 表示样本；随机变量h 表示该样本有没有标签，$h=1$ 表示该样本没有标签；随机变量 y 表示该样本的标签。一般来说样本的标签概率分布只与样本 x 有关，而与该样本是否有标签无关，所以根据该假设可以得出 $P(y|x,h=1)=p(y|x)$。(该假设在大部分情况下是成立的，文章也指出有少数情况是特殊的，例如数据采集者采集一些敏感数据时，被采集者没有提供标签可能说明该标签是让其羞耻的，在这种情况下，那些羞耻类的标签的概率分布就会大点。）

实际数据中，样本 x 是有限的，经验条件熵可以写成如下形式:

$$H_{\mathrm{emp}}(y|x,h=1;\mathcal{L}_n) = -\frac{1}{u}\sum_{i=l+1}^{n}\sum_{m=1}^{M} P(m|x_i)\ln P(m|x_i)\ .$$

Entropy Regularization

作者将该条件熵和有标签样本的似然概率结合在一起就得到最终的最大后验的目标如下：

$$\begin{aligned} C(\theta,\lambda;\mathcal{L}_n) &= L(\theta;\mathcal{L}_n) - \lambda H_{\mathrm{emp}}(y|x,h=1;\mathcal{L}_n) \\ &= \sum_{i=1}^{l}\ln P(y_i|x_i;\theta) + \lambda\sum_{i=l+1}^{n}\sum_{m=1}^{M} P(m|x_i;\theta)\ln P(m|x_i;\theta)\ , \end{aligned}$$

公式左边部分就是有标签样本的最大似然，右边就是最小化香农条件熵，熵值在其均匀分布的时候取最大值，极端分布的时候取最小值。所以右半部分简单来说，就是鼓励概率模型 P 为每个样本都输出极端的类别概率分布，即概率分布中某一个值较大而其它的值都较小。

Optimization Algorithms

文中使用 EM 算法来优化上文中目标函数。在 E 阶段，作者根据当前的参数 θ 求出，每个无标签样本的标签的后验分布，如下：

$$g_m(x_i;\theta) = \frac{P(m|x_i;\theta)^{\frac{1}{1-\lambda}}}{\sum_{\ell=1}^{M} P(\ell|x_i;\theta)^{\frac{1}{1-\lambda}}} \quad ,$$

在 M 阶段，就固定 m，优化 θ, 如下：

$$\theta^{s+1} = \arg\max_{\theta} \sum_{i=1}^{n} \sum_{m=1}^{M} g_m(x_i;\theta^s) \ln P(m|x_i;\theta) \quad ,$$

这个部分只包含了香农条件熵的部分，在这一步优化还要结合有标签样本的最大似然目标。

实验

数据：人工数据，从分布中采样得到。

1. 无标签数据和有标签数据从相同分布中采样

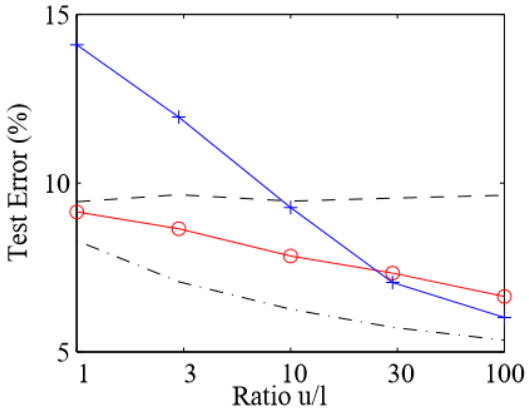


Figure 9.2 Test error *vs.* u/l ratio for 5 % Bayes error ($a = 0.23$). Test errors of minimum entropy logistic regression (○) and mixture models (+). The errors of logistic regression (dashed), and logistic regression with all labels known (dash-dotted) are shown for reference.

其中， 横轴是无标签数据和有标签数据的比例；minimum entropy logistic regression 是本文提出的模型， mixture models 是混合高斯模型，这是一种生成式模型。logistic regression 直接忽略了无标签数据。a 是样本采样的均值，class 1：(a, a, ..., a), class 2: -(a, ,a, .., a)。下同。

2. 无标签的数据为分布边缘采样或随机生成。

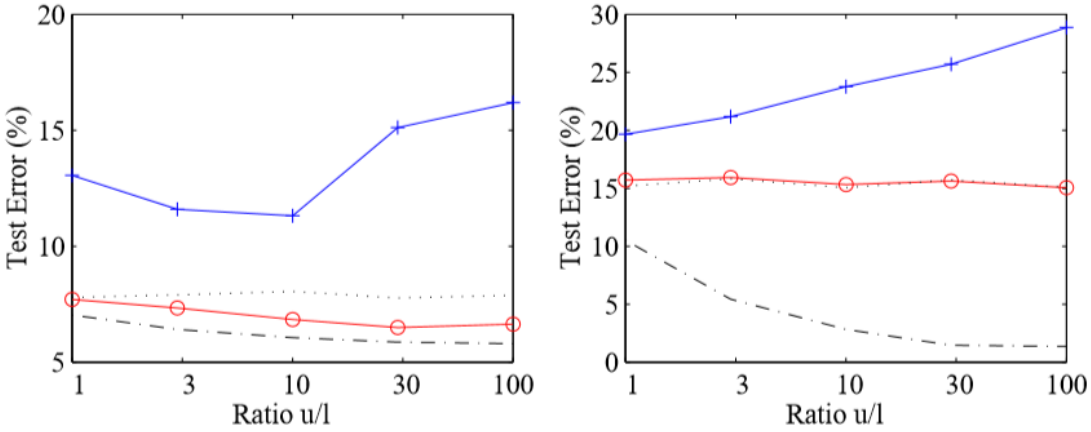


Figure 9.3 Test error *vs.* u/l ratio for $a = 0.23$. Average test errors for minimum entropy logistic regression ($\circ$) and mixture models ($+$). The test error rates of logistic regression (dotted), and logistic regression with all labels known (dash-dotted) are shown for reference. Left: experiment with outliers; right: experiment with uninformative unlabeled data.

Pseudo-Label : The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks

In ICML'13

本文提出一种伪标签的学习方法迭代式地为大量的未标注样本标注一个伪标签，然后将标注的样本加入到训练集中训练模型。通过这种方法，模型可以利用到无标签的数据，从而提高模型性的分类能力。

训练过程

模型的损失函数是一个随着时间 t 变化的损失函数，如下：

$$L = \frac{1}{n}\sum_{m=1}^{n}\sum_{i=1}^{C} L(y_i^m, f_i^m) + \alpha(t)\frac{1}{n'}\sum_{m=1}^{n'}\sum_{i=1}^{C} L(y_i'^m, f_i'^m),$$

其中，C 是样本类别的个数，y_i 是一个 1 of K 的标签，损失函数 L 为交叉熵；n 为有标签样本的个数，n' 为有伪标签样本的个数，其标签是在训练的过程中用模型 f 动态标注的，每次训练其标签都有可能不同。$\alpha(t)$ 是当前时刻 t 时，伪标签样本所占的权重。

伪标签定义

文章确定伪标签的方法非常简单，即使用当前时刻 t 之前一次的模型的预测结果。文章使用预测概率最大的那一个类别作为待预测样本的伪标签。公式如下：

$$y_i' = \begin{cases} 1 & \text{if } i = \text{argmax}_{i'}\, f_{i'}(x) \\ 0 & \text{otherwise} \end{cases}$$

$\alpha(t)$ 的变化趋势

文章指出，训练过程中，权重$\alpha(t)$的调整非常重要。首先 α 的值一开始一段时间一定是 0，这段时间，模型只使用有真实标签的数据训练模型。之后，如果 α 值过大，则模型会受到无标签数据的错误干扰；反之，模型又不能有效利用无标签数据的信息。文中，作者给出了一个调整的策略，如下：

$$\alpha(t) = \begin{cases} 0 & t < T_1 \\ \frac{t-T_1}{T_2-T_1}\alpha_f & T_1 \leq t < T_2 \\ \alpha_f & T_2 \leq t \end{cases} \tag{16}$$

with $\alpha_f = 3, T_1 = 100, T_2 = 600$ without pre-training, $T_1 = 200, T_2 = 800$ with DAE.

其它细节

模型的预训练

文中，作者还使用了常用的 DAE (Denoised AutoEncoder) 模型来预训练模型 f。DAE 基本过程如下：

$$h_i = s\left(\sum_{j=1}^{d_v} W_{ij}\widetilde{x}_j + b_i\right) \tag{6}$$

$$\widehat{x}_j = s\left(\sum_{i=1}^{d_h} W_{ij}h_i + a_j\right) \tag{7}$$

其中，输入 $\widetilde{x}_j$ 表示加了随机小噪声的样本，$\hat{x}_j$ 为模型重构的样本，公式 (6) 为 Encoder，公式 (7) 为 Decoder，两本部分公式均为简单表示，可以采用更复杂的结构。我们的分类模型 f 就作为 Encoder。DAE 的损失函数事宜真实样本为标签的交叉熵，公式如下：

$$L(x, \widehat{x}) = \sum_{j=1}^{d_v} -x_j \log \widehat{x}_j - (1 - x_j)\log(1 - \widehat{x}_j) \tag{8}$$

防止过拟合

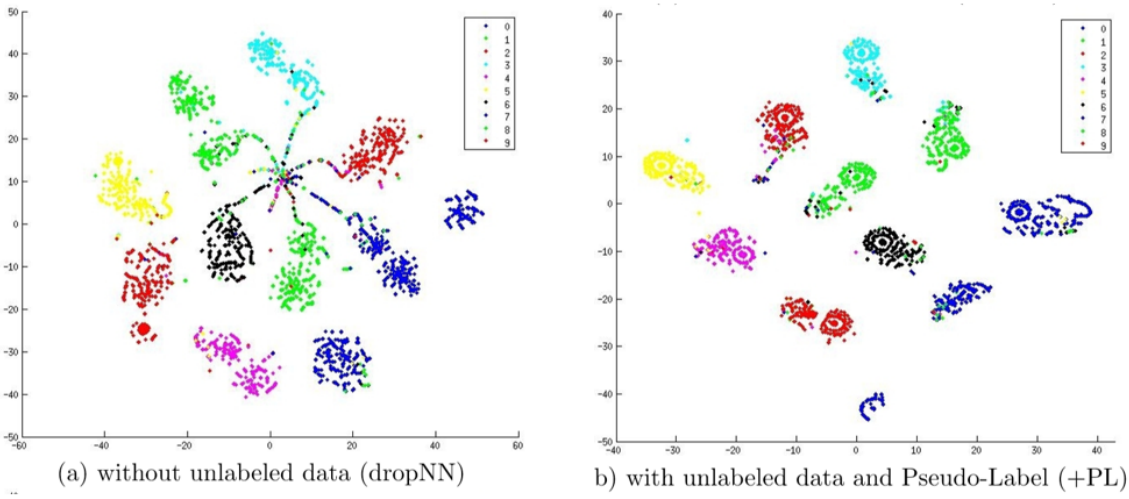
文中，作者使用 Dropout 方法来防止模型过拟合。

分析：

这种优化方式和 Entropy Regularization基本上一样，不同的是，在 Entropy Regularization在求标签的后验分布的时候，保留了连续性。而本文中预测伪标签的时候直接将其离散化了，保留概率最大的为 1，其余的都设为 0。然后本文在α的调整上更细，也使用了新的预训练和防止过拟合技术。

实验结果

数据集：MNIST，训练集中有标签数据量为 600， 无标签数据量为 60000.



(a) without unlabeled data (dropNN)

b) with unlabeled data and Pseudo-Label (+PL)

文章中，作者为展示伪标签方法效果，准备了额外的有真实标签的样本作为测试集。图中为模型提取的测试集特征使用 t-SNE 方法降维后的展示结果。

上图中左图为只使用训练集中有标签数据训练模型后的结果，右图为使用了伪标签方法后的结果。可以看出来，通过伪标签方法利用到无标签数据的信息后，模型能够提取更具有区别性的特征。

下图为测试集上错误率的结果, 从左到右分别为测试集大小为 100， 600， 1000， 3000 的结果。

DROPNN	21.89	8.57	6.59	3.72
+PL	16.15	5.03	4.30	2.80
+PL+DAE	**10.49**	**4.01**	**3.46**	2.69

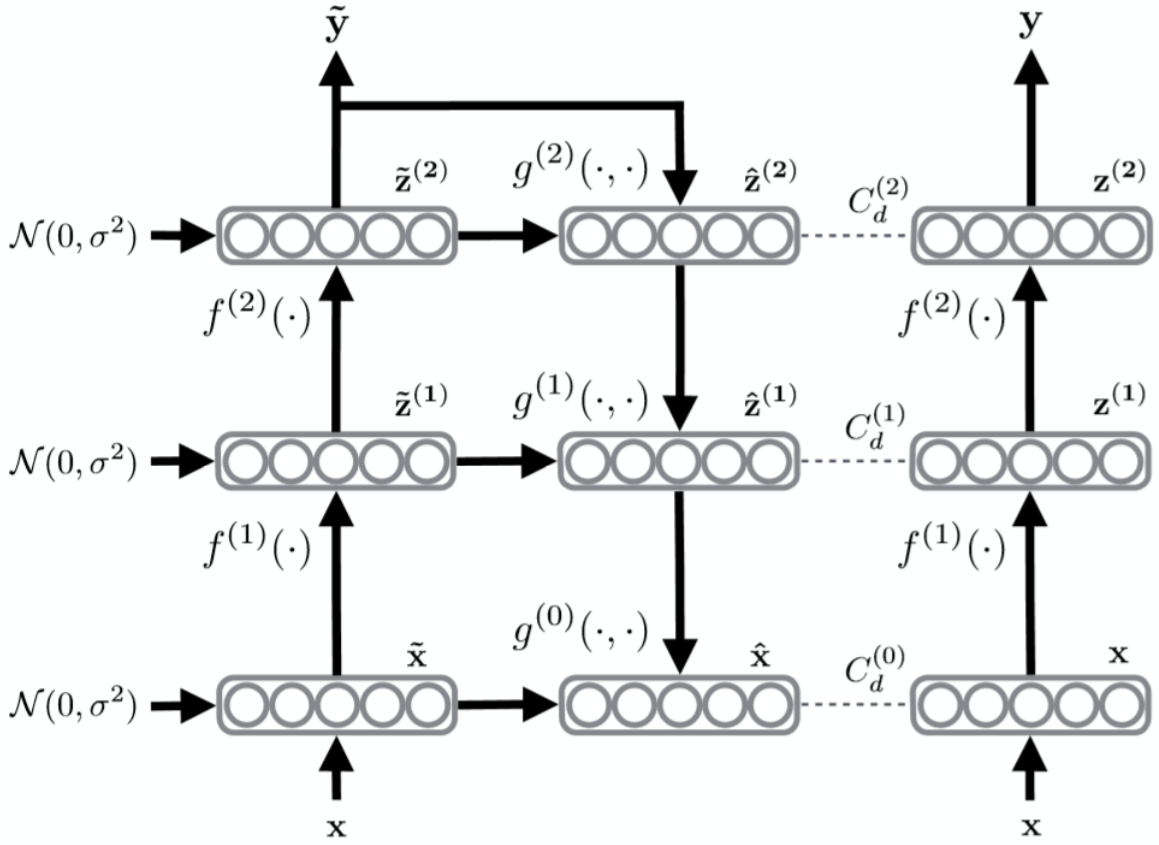
Semi-Supervised Learning with Ladder Networks

In NIPS'15

Ladder Networks 是一种无监督学习方法，在 DAE 的基础上增加了 skip connection 和 layer reconstruct loss。之前的方法都使用 Ladder Networks 之类的无监督学习方法来预训练网络。例如前面，伪标签学习的方法就是用 DAE 方法预训练网络。本文就是将 Ladder networks 和有监督学习结合起来一起训练。关于结合方法，作者提出三个重要的点：

1. Compatibility with supervised methods. 能够尽可能多的监督学习方法兼容。
2. Scalability resulting from local learning. 通过每一层的无监督loss，使得模型的深度具有可扩展性，模型能够有很多层。
3. Computational efficiency. 和普通的监督学习网络具有同等级的效率，保证只是常数倍的复杂度增加。

跟 Entropy Regularization 和 Pseudo label 的方法比起来，该方法在有监督学习loss是相同的，然后正则化项换成了 AutoEncoder 结构重构 loss。具体方法如下：



如上图所示，模型运算过程如下：

1. 给定输入 x，模型分为两条路径向前传播。一条是左边的 corrupted encoder，这一条路径会对每一层的特征加上一个高斯噪声，噪声的均值和方差是当前层特征的均值和方差；另一条是右边的 clean encoder，这就是一个正常的前向传播网络。两个 encoder 的参数是共享的。
2. Decoder loss：中间的部分是一个 decoder network，除了原来的decoder路径，还增加按层对应的 skip connection。重构 loss 也是每层一个，对应的标签是 clearn encoder 的每一层。如上图中的 $C_d^{(0)}, C_d^{(1)}, C_d^{(2)}$ 三个loss。如前文所说这样可以使得模型深度具有可扩展性。
3. Supervised loss: 值得注意的是，文章使用 corrutted encoder 的输出 $\hat{y}$ 作为样本 x 的类别预测输出，而不是用 clean encoder 的输出或者两者的结合。
4. 将 decoder loss 和 supervised loss 结合起来，其中无标签的样本只有 decoder loss。然后就可以使用任意的梯度下降方法更新模型。

具体的算法如下图所示，可以看出作者在encoder 和 decoder 的每一层都使用了 Batch Normalization。

Algorithm 1 Calculation of the output and cost function of the Ladder network

Require: $\mathbf{x}(n)$
  # Corrupted encoder and classifier
  $\tilde{\mathbf{h}}^{(0)} \leftarrow \tilde{\mathbf{z}}^{(0)} \leftarrow \mathbf{x}(n) + \texttt{noise}$
  for l = 1 **to** L **do**
    $\tilde{\mathbf{z}}_{\text{pre}}^{(l)} \leftarrow \mathbf{W}^{(l)}\tilde{\mathbf{h}}^{(l-1)}$
    $\tilde{\boldsymbol{\mu}}^{(l)} \leftarrow \texttt{batchmean}(\tilde{\mathbf{z}}_{\text{pre}}^{(l)})$
    $\tilde{\boldsymbol{\sigma}}^{(l)} \leftarrow \texttt{batchstd}(\tilde{\mathbf{z}}_{\text{pre}}^{(l)})$
    $\tilde{\mathbf{z}}^{(l)} \leftarrow \texttt{batchnorm}(\tilde{\mathbf{z}}_{\text{pre}}^{(l)}) + \texttt{noise}$
    $\tilde{\mathbf{h}}^{(l)} \leftarrow \texttt{activation}(\boldsymbol{\gamma}^{(l)} \odot (\tilde{\mathbf{z}}^{(l)} + \boldsymbol{\beta}^{(l)}))$
  end for
  $P(\tilde{\mathbf{y}} \mid \mathbf{x}) \leftarrow \tilde{\mathbf{h}}^{(L)}$
  # Clean encoder (for denoising targets)
  $\mathbf{h}^{(0)} \leftarrow \mathbf{z}^{(0)} \leftarrow \mathbf{x}(n)$
  for l = 1 **to** L **do**
    $\mathbf{z}^{(l)} \leftarrow \texttt{batchnorm}(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)})$
    $\mathbf{h}^{(l)} \leftarrow \texttt{activation}(\boldsymbol{\gamma}^{(l)} \odot (\mathbf{z}^{(l)} + \boldsymbol{\beta}^{(l)}))$
  end for
  # Final classification:
  $P(\mathbf{y} \mid \mathbf{x}) \leftarrow \mathbf{h}^{(L)}$
  # Decoder and denoising
  for l = L **to** 0 **do**
    if l = L **then**
      $\mathbf{u}^{(L)} \leftarrow \texttt{batchnorm}(\tilde{\mathbf{h}}^{(L)})$
    else
      $\mathbf{u}^{(l)} \leftarrow \texttt{batchnorm}(\mathbf{V}^{(l)}\hat{\mathbf{z}}^{(l+1)})$
    end if
    $\forall i : \hat{z}_i^{(l)} \leftarrow g(\tilde{z}_i^{(l)}, u_i^{(l)})$ # Eq. (1)
    $\forall i : \hat{z}_{i,\text{BN}}^{(l)} \leftarrow \frac{\hat{z}_i^{(l)} - \tilde{\mu}_i^{(l)}}{\tilde{\sigma}_i^{(l)}}$
  end for
  # Cost function C for training:
  $\text{C} \leftarrow 0$
  if $t(n)$ **then**
    $\text{C} \leftarrow -\log P(\tilde{\mathbf{y}} = t(n) \mid \mathbf{x})$
  end if
  $\text{C} \leftarrow \text{C} + \sum_{l=0}^{L} \lambda_l \left\| \mathbf{z}^{(l)} - \hat{\mathbf{z}}_{\text{BN}}^{(l)} \right\|^2$

实验

数据集：MNIST

Test error % with # of used labels	100	1000	All
Semi-sup. Embedding (Weston *et al.*, 2012)	16.86	5.73	1.5
Transductive SVM (from Weston *et al.*, 2012)	16.81	5.38	1.40*
MTC (Rifai *et al.*, 2011)	12.03	3.64	0.81
Pseudo-label (Lee, 2013)	10.49	3.46	
AtlasRBF (Pitelis *et al.*, 2014)	8.10 ($\pm$ 0.95)	3.68 ($\pm$ 0.12)	1.31
DGN (Kingma *et al.*, 2014)	3.33 ($\pm$ 0.14)	2.40 ($\pm$ 0.02)	0.96
DBM, Dropout (Srivastava *et al.*, 2014)			0.79
Adversarial (Goodfellow *et al.*, 2015)			0.78
Virtual Adversarial (Miyato *et al.*, 2015)	2.12	1.32	0.64 ($\pm$ 0.03)
Baseline: MLP, BN, Gaussian noise	21.74 ($\pm$ 1.77)	5.70 ($\pm$ 0.20)	0.80 ($\pm$ 0.03)
Γ-model (Ladder with only top-level cost)	3.06 ($\pm$ 1.44)	1.53 ($\pm$ 0.10)	0.78 ($\pm$ 0.03)
Ladder, only bottom-level cost	1.09 ($\pm$0.32)	0.90 ($\pm$ 0.05)	0.59 ($\pm$ 0.03)
Ladder, full	**1.06** ($\pm$ 0.37)	**0.84** ($\pm$ 0.08)	**0.57** ($\pm$ 0.02)

上图中， top-level cost 就是结构图中最上层重构 loss，也就是$C_d^{(2)}$, bottom-level cost 也就是最下层的重构 loss，也就是 $C_d^{(0)}$ 。从实验可以看出，该方法效果很不错。

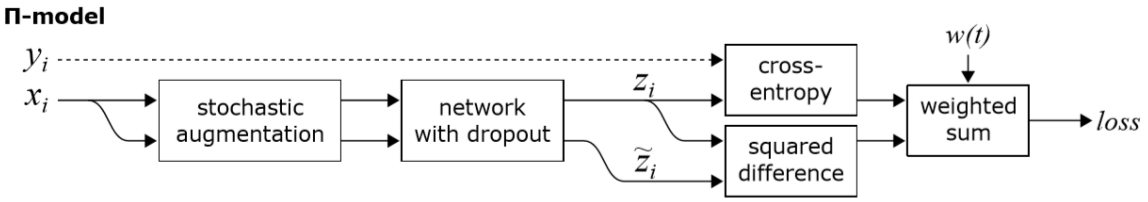
Temporal Ensembling for Semi-supervised Learning

In ICLR'17

Ensemble 大法好。作者认为： *It has long been known that an ensemble of multiple neural networks generally yields better predictions than a single network in the ensemble.* 所以作者就将网络前面多次的预测结果累积起来作为当前训练的监督信息。

本文的监督方法是类似于 Ladder Network 的结构，相比于 Ladder Network 简化了许多。

作者首先提出了没有 Ensemble 的基本结构，称之为 $\Pi - Model$. 结构如下：



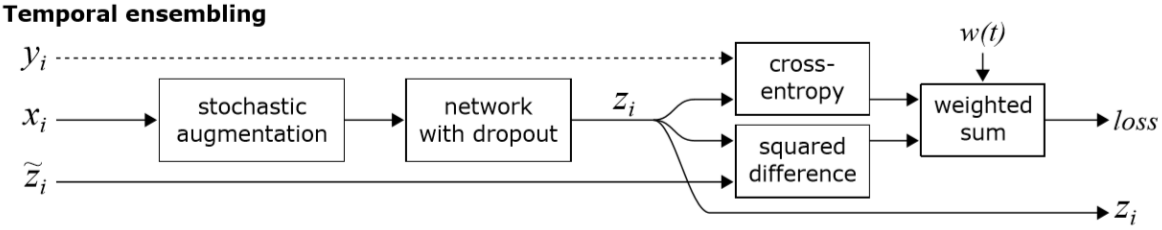
给定一个输入 x_i，如 Ladder Network 一样，模型使用相同的结构给出两个输出 $z_i, \hat{z}_i$, 两个输出的网络结构相同，但是dropout 参数和噪声不同。之后直接算一个 squared loss 作为无监督的 loss，相当于 Ladder Network 的 top level loss。然后再和有监督loss结合起来，结合权重是动态的 $w(t)$, 这个和伪标签方法一样，但是实验中具体的变化趋势不同，这个对于不同的问题，都要具体调整。具体的算法过程如下图所示：

Algorithm 1 Π-model pseudocode.

Require: x_i = training stimuli
Require: L = set of training input indices with known labels
Require: y_i = labels for labeled inputs $i \in L$
Require: $w(t)$ = unsupervised weight ramp-up function
Require: $f_\theta(x)$ = stochastic neural network with trainable parameters θ
Require: $g(x)$ = stochastic input augmentation function
for t in $[1, num\_epochs]$ **do**
  for each minibatch B **do**
    $z_{i \in B} \leftarrow f_\theta(g(x_{i \in B}))$ ▷ evaluate network outputs for augmented inputs
    $\tilde{z}_{i \in B} \leftarrow f_\theta(g(x_{i \in B}))$ ▷ again, with different dropout and augmentation
    $loss \leftarrow -\frac{1}{|B|} \sum_{i \in (B \cap L)} \log z_i[y_i]$ ▷ supervised loss component
      $+ w(t) \frac{1}{C|B|} \sum_{i \in B} ||z_i - \tilde{z}_i||^2$ ▷ unsupervised loss component
    update θ using, e.g., ADAM ▷ update network parameters
  end for
end for
return θ

Temporal ensembling model

在 $\Pi - model$ 的基础上，作者累积之前的预测结果作为监督信息，而不是在同一次迭代预测两次。



模型的结构如上图所示，监督信息来自该样本之间的预测结果。这样的好处是每次迭代，模型只要预测一次，提高了效率；使用之前累积的预测结果，达到了 ensemble 的效果。具体的ensemble 方法如下图所示

Algorithm 2 Temporal ensembling pseudocode. Note that the updates of Z and $\tilde{z}$ could equally well be done inside the minibatch loop; in this pseudocode they occur between epochs for clarity.

Require: x_i = training stimuli
Require: L = set of training input indices with known labels
Require: y_i = labels for labeled inputs $i \in L$
Require: α = ensembling momentum, $0 \leq \alpha < 1$
Require: $w(t)$ = unsupervised weight ramp-up function
Require: $f_\theta(x)$ = stochastic neural network with trainable parameters θ
Require: $g(x)$ = stochastic input augmentation function

$Z \leftarrow \mathbf{0}_{[N \times C]}$ ▷ initialize ensemble predictions
$\tilde{z} \leftarrow \mathbf{0}_{[N \times C]}$ ▷ initialize target vectors
for t in $[1, num\_epochs]$ **do**
  for each minibatch B **do**
    $z_{i \in B} \leftarrow f_\theta(g(x_{i \in B}, t))$ ▷ evaluate network outputs for augmented inputs
    $loss \leftarrow -\frac{1}{|B|} \sum_{i \in (B \cap L)} \log z_i[y_i]$ ▷ supervised loss component
      $+ w(t) \frac{1}{C|B|} \sum_{i \in B} ||z_i - \tilde{z}_i||^2$ ▷ unsupervised loss component
    update θ using, e.g., ADAM ▷ update network parameters
  end for
  $Z \leftarrow \alpha Z + (1 - \alpha) z$ ▷ accumulate ensemble predictions
  $\tilde{z} \leftarrow Z / (1 - \alpha^t)$ ▷ construct target vectors by bias correction
end for
return θ

从图中，可以看出作者使用 Z 来累积预测结果，每当有新的预测结果 z 时，更新如下：

$$Z \leftarrow \alpha Z + (1 - \alpha) z$$

提供监督信息时，作者按照迭代次数做了纠正，称之为 *correct up for the startup bias*，方法如下：

$$\hat{z} \leftarrow Z / (1 - \alpha^t)$$

实验

Table 1: CIFAR-10 results with 4000 labels, averages of 10 runs (4 runs for all labels).

	Error rate (%) with # labels	
	4000	All (50000)
Supervised-only	35.56 ± 1.59	7.33 ± 0.04
with augmentation	34.85 ± 1.65	6.05 ± 0.15
Conv-Large, Γ-model (Rasmus et al., 2015)	20.40 ± 0.47	
CatGAN (Springenberg, 2016)	19.58 ± 0.58	
GAN of Salimans et al. (2016)	18.63 ± 2.32	
Π-model	16.55 ± 0.29	6.90 ± 0.07
Π-model with augmentation	12.36 ± 0.31	$\mathbf{5.56 \pm 0.10}$
Temporal ensembling with augmentation	$\mathbf{12.16 \pm 0.24}$	5.60 ± 0.10

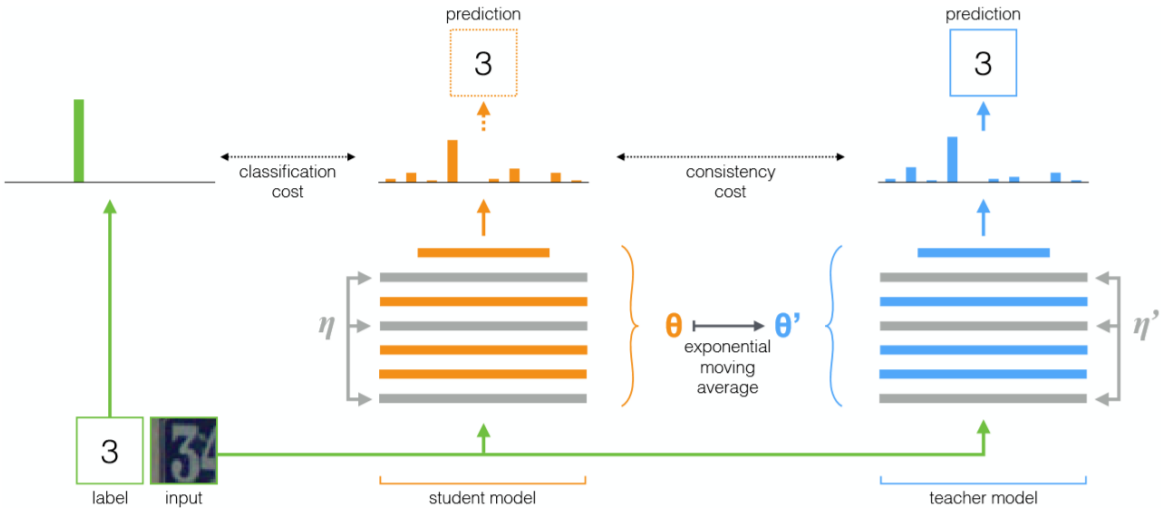
基于CIFAR-10 数据集构造的实验，从实验可以看出，该方法效果很不错。其中，$\tau - Model$就是前文介绍的基于 Ladder Network 的模型，但是实验标准依然没有统一，因为深度学习非常依赖参数调整，所以参考性也不强。ensemble 的想法是非常可取的。

Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results

In arXiv'18

理一理思路，Eentropy Regularization 和 Pseudo Label 就是把上次预测的标签作为本次的监督信息，分别用 soft label 和 hard label。Temporal Ensembling 时把之间的预测结果都累积起来，因为不一定最近的一次就是最准确的监督信息，使用的时 soft label。

本文借鉴了 Temporal Ensembling 按照迭代次数 ensemble 的想法，但是使用伪标签（soft label or hard label）的缺点是，每经过一个 epoch，label 才会更新。这样，随着数据集数据集的增大，信息反馈就会越来越大。所以作者提出了一个大胆的想法，不 ensemble 样本的预测标签，而 ensemble 模型的参数作为监督信息。流程如下图所示：



左边是有监督的分类 loss，右边是无监督 loss，不同的的是 teacher model 是之前的student models 的平均结果。这样没过一个 mini batch 就可以更新teacher model，而不需要等待一个 epoch。

直接看这个想法是很大胆的，自己学自己，有点优化方法中动量的感觉，但是优化中的动量只是保持趋势，还是容易理解的，这里就很费解。作者的解释是，为了泛化性，模型的输入都是加噪声的，在模型训练到一定情况时，不同的噪声引起不同的变化，这时候取均值往往是更准确的结果。总之，ensemble 总是好的Orz

实验

Table 2: Error rate percentage on CIFAR-10 over 10 runs (4 runs when using all labels).

	1000 labels 50000 images	2000 labels 50000 images	4000 labels 50000 images	50000 labels 50000 images
GAN [25]			18.63 ± 2.32	
Π model [13]			12.36 ± 0.31	5.56 ± 0.10
Temporal Ensembling [13]			12.16 ± 0.31	**5.60 ± 0.10**
VAT+EntMin [16]			**10.55**	
Supervised-only	46.43 ± 1.21	33.94 ± 0.73	20.66 ± 0.57	5.82 ± 0.15
Π model	27.36 ± 1.20	18.02 ± 0.60	13.20 ± 0.27	6.06 ± 0.11
Mean Teacher	**21.55 ± 1.48**	**15.73 ± 0.31**	12.31 ± 0.28	5.94 ± 0.15

这是一个benchmark实验，缺失的数据是原文没有做这个实验。可以看出来效果还是不错的。VAT 模型在下面介绍。

Virtual Adversarial Training

In arXiv'18

前面的半监督学习方法都会用到在输入样本上加随机噪声的方法来增加模型的泛化性。但是其所增加的噪声都是随机的，并不一定是好的噪声。为了解决这个问题，本文就利用对抗学习的成果设计方法来学习这个噪声。

这个噪声的学习目标最能干扰模型的判别，评估标准即为加了噪声后，模型预测的类别概率分布发生最大的变化，公式如下：

$$r_{\mathrm{qadv}} := \arg\max_{r;\|r\|\leq\epsilon} D\left[q(y|x_*), p(y|x_*+r,\theta)\right],$$

其中 $q(y|x_*)$ 是真实的标签概率分布，$p(y|x_*+r,\theta)$ 为干扰后的模型输出的标签概率分布。这个公式的近似求解，在对抗样本中已经使用很频繁，就是对样本求一阶导数，如下：

$$r_{\mathrm{adv}} \approx \epsilon\frac{g}{\|g\|_2}, \text{ where } g = \nabla_{x_l} D\left[h(y;y_l), p(y|x_l,\theta)\right]$$

之后，我们再在该对抗样本的基础上训练模型，以增强模型的泛化性，目标如下：

$$D\left[q(y|x_*), p(y|x_*+r_{\mathrm{qadv}},\theta)\right]$$
$$\text{where } r_{\mathrm{qadv}} := \arg\max_{r;\|r\|\leq\epsilon} D\left[q(y|x_*), p(y|x_*+r,\theta)\right],$$

这就是一个标准的对抗样本学习的直接应用，但是在半监督学习中，样本是无标签的，即我们不知道样本的标签概率分布 $q(y|x_*)$. 为了解决这个问题，作者使用更前面的文章一样的方法，即使用之前的模型的预测结果 $p(y|x,\hat{\theta})$作为伪标签，这里称之为 virtual label，所以叫 Virtual Adversarial Training。修改后 loss 如下：

$$\mathrm{LDS}(x_*,\theta) := D\left[p(y|x_*,\hat{\theta}), p(y|x_*+r_{\mathrm{vadv}},\theta)\right]$$
$$r_{\mathrm{vadv}} := \arg\max_{r;\|r\|_2\leq\epsilon} D\left[p(y|x_*,\hat{\theta}), p(y|x_*+r)\right],$$

最后，再和有标签样本的 loss 结合，就得到最终的训练目标：

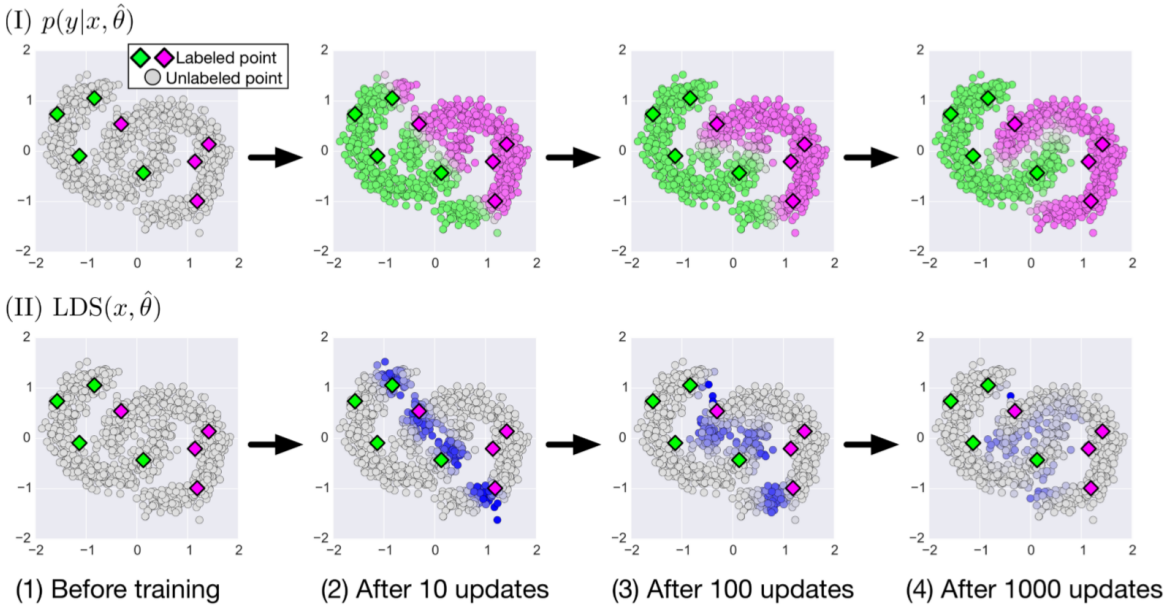
$$\ell(\mathcal{D}_l,\theta) + \alpha\mathcal{R}_{\mathrm{vadv}}(\mathcal{D}_l,\mathcal{D}_{ul},\theta), \tag{8}$$

分析

这篇文章可以说是另辟蹊径，着眼点与之前的文章都不同。文章中，作者使用最近一次的预测结果 $p(y|x,\hat{\theta})$ 作为伪标签，使用标签是概率分布，属于 soft label，也是前面的文章中最常用的。重要的是，该文章还可以和 ensemble 的方法结合达到更好的结果。

实验

数字的实验前面已经汇报过了。本文还做了一个简单的可视化实验，有助于加深我们对半监督学习的理解。实验结果如下图所示。



1. 第一部分很好理解，就是随着训练的进行，无标签的样本都能够正确分类。
2. 第二部分展示的是 $LDS(x,\hat{\theta})$的热力图，即该数值越大，颜色就越深。一开始的时候，输出的都是随机的，加了噪声之后也没有影响，所以 LDS 的值都很小。等训练好之后，LDS数值较大的都分布在分类边界上，因为在分类边界上的点，再加一个小噪声，其标签概率分布就可能改变。
3. 这个效果比较好是因为选取的有标签样本很具有代表性，基本覆盖分类边界。如果有标签样本都分布在中间就很难有这样的效果。这些数据是人为设计的。在现实的数据收集中，我们就要求特征全面>特征完整。例如做恶意软件识别，我们希望利用人力在8类恶意行为中，每类都有部分收集，然后再利用半监督学习学习其它的，而不是用人力收集一类恶意行为，然后学习其他类别的恶意行为。

Realistic Evaluation of Deep Semi-Supervised Learning Algorithms

In NIPS'18

这是一篇泼冷水的文章，但是冷水使人冷静，对我们也非常有用, 我们在复现时，这篇文章就提供了很好的参考。这篇文章对前面方法中的一些方法进行了系统规范，统一标准的评估。

评估方法改进（摘录）

SSL算法一般遵循以下流程：首先，选择一个用于监督学习的通用数据集，删去其中大多数数据的标签；其次，把保留标签的数据制作成小型数据集D，把未标记数据整理成数据集DUL；最后，用半监督学习训练一些模型，在未经修改的测试集上检验它们的性能。

下面是现有方法的缺陷及其改进：

1. 一个共享的实现

现有SSL算法比较没有考虑底层模型的一致性，这是不科学的。在某些情况下，同样是简单的13层CNN，不同实现会导致一些细节，比如参数初始化、数据预处理、数据增强、正则化等，发生改变。不同模型的训练过程（优化、几个epoch、学习率）也是不一样的。因此，如果不用同一个底层实现，算法对比不够严谨。

2. 高质量监督学习基线

SSL的目标是基于标记数据集 D 和未标记数据集 DUL，使模型的性能比单独用 D 训练出来的完全相同的基础模型更好。虽然道理很简单，但不同论文对于这个基线的介绍却存在出入，比如去年 Laine & Aila 和 Tarvainen & Valpola 在论文中用了一样的基线，虽然模型是一样的，但它们的准确率差竟然高达15%。（两篇文章分别为前面的 Temporal ensembling 和 Mean teacher）

为了避免这种情况，我们参考为SSL调参，重新调整了基线模型，确保它的高质量。

3. 和迁移学习的对比

在实践中，如果数据量有限，通常我们会用迁移学习，把在相似大型数据集上训练好的模型拿过来，再根据手头的小数据集进行“微调”。虽然这种做法的前提是存在那么一个相似的、够大的数据集，但如果能实现，迁移学习确实能提供性能强大的、通用性好的基线，而且这类基线很少有论文提及。

4. 考虑类分布不匹配

需要注意的是，当我们选择数据集并删去其中大多数数据的标签时，这些数据默认 DUL 的类分布和D的完全一致。但这不合理，想象一下，假设我们要训练一个能区分十张人脸的分类器，但每个人的图像样本非常少，这时，你可能会选择使用一个包含随机人脸图像的大型未标记数据集来进行填充，那么这个 DUL 中的图像就并不完全是这十个人的。

现有的SSL算法评估都忽略了这种情况，而我们明确研究了类分布相同/类分布不同数据之间的影响。

5. 改变标记和未标记数据的数量

改变两种数据的数量这种做法并不罕见，研究人员通常喜欢通关删去不同数量的底层标记数据来改变D的大小，但到目前为止，以系统的方式改变 DUL 确不太常见。这可以模拟两种现实场景：一是未标记数据集非常巨大（比如用网络数十亿未标记图像提高模型分类性能），二是未标记数据集相对较小（比如医学影像数据，它们的成本很高）。

6. 切合实际的小型验证集

人为创建的SSL数据集往往有个特征，就是验证集会比训练集大很多。比如SVHN的验证集大约有7000个标记数据，许多论文在用这个数据集做研究时，往往只从原训练集里抽取1000个标记数据，但会保留完整验证集。这就意味着验证集是训练集的7倍，而在现实任务中，数据更多的集一般是会被作为训练集的。

实验

Method	CIFAR-10 4k Labels	SVHN 1k Labels
Π-M (Sajjadi et al., 2016b)	11.29%	–
Π-M (Laine & Aila, 2017)	12.36%	4.82%
MT (Tarvainen & Valpola, 2017)	12.31%	3.95%
VAT (Miyato et al., 2017)	11.36%	5.42%
VAT + EM (Miyato et al., 2017)	10.55%	3.86%
Results above this line cannot be directly compared to those below		
Supervised	20.26 ± 0.38%	12.83 ± 0.47%
Π-Model	16.37 ± 0.63%	7.19 ± 0.27%
Mean Teacher	15.87 ± 0.28%	5.65 ± 0.47%
VAT	13.86 ± 0.27%	5.63 ± 0.20%
VAT + EM	13.13 ± 0.39%	5.35 ± 0.19%
Pseudo-Label	17.78 ± 0.57%	7.62 ± 0.29%

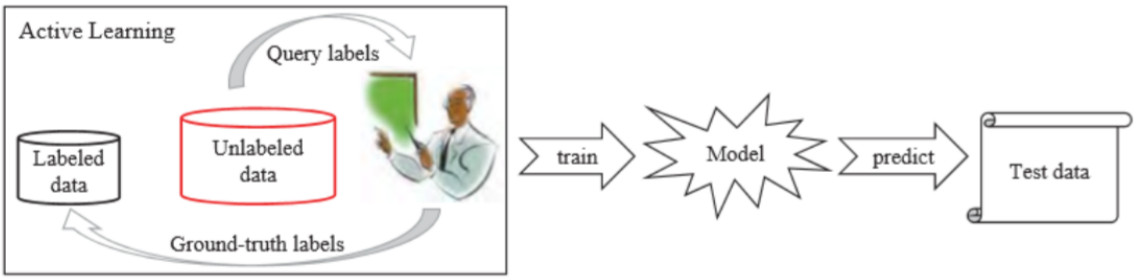
上半部分为原论文汇报的数据，下半部分为本文的实验。因为模型配置不同，本文使用的是统一模型，参数比原文都少，所以不能对比。EM 是 Entropy Minimization，就是第一部分的 Entropy Regularization 的 loss。

文章还做了迁移学习的实验，首先在一个大的数据集上（例如 ImageNet）预训练模型，然后只用有标签的少量样本做 fine-tuning，CIFAR-10的结果为 12% 的错误率，效果是很好的。使用迁移学习也是很好的，不过我们的场景中没有数据。

Large-scale interactive object segmentation with human annotators

In CVPR'19

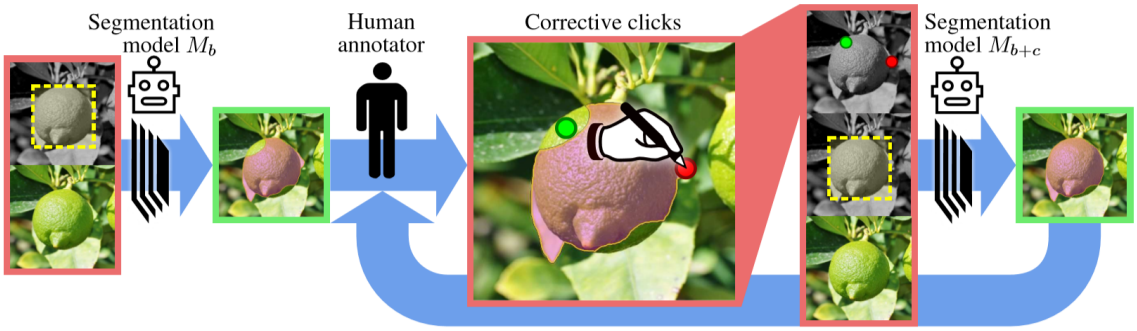
这篇文章使用了半监督学习中的，主动学习(Active Learning)的模式，如下：



主动学习相比于传统的半监督学习，加入了人工指导，可以使结果更加准确。我们也是考虑要使用主动学习的模式的，但是这篇文章的交互方式对我们的方法还没有什么启发，我们现在的方案是人工抽选评估伪标签。要写文章的话，这篇文章可以提供一个思路，其突出的贡献就是做了大量的数据标注工作。

内容介绍

这篇文章是做图像分割掩码的，纯人工的分割掩码非常耗时。本文提出一种交互式标注方法。主要流程如下：



1. 输入是原始图片和box掩码
2. 模型 M_b 使用卷积结构，输入是四个channel，第四个channel为 box mask 信息，输出就就是每个像素的二分类结果，即这个像素是不是mask。训练 M_b 需要有正确的 mask 作为监督信息。
3. 然后人工对预测结果进行标注。标注为这块应当是mask（绿点）或者不应当是mask（红点）。
4. 模型 M_{b+c} 和 M_b 使用同样的卷积结构，输入是五个或更多的channel，第五个以后的channel 都是迭代过程中人工标注的信息。输出与 M_b 相同。训练时同样需要正确的 mask 作为监督信息。

M_b 和 M_{b+c} 的迭代都是迭代式的，一开始使用现有的人工数据训练 M_b, 然后使用这套框架训练 M_{b+c}, 同时可以得到一些标记好的数据；然后再训练 M_b。 文章的一个主要贡献是花了9个月时间用这个框架为 250 万图片做了分割掩码。文章还在 COCO dataset 做了实验，验证这套方法的标注效率大概是人工标注的3倍。