ReMixMatch : semi-supervised learning with distribution alignment and augmentation anchoring

Abstract

- 通过引入distribution alignment和augmentation anchoring改进了近期的MixMatch工作
- Distribution alignment鼓励对无标签的预测的边缘分布更加接近于真实标签的边缘分布
- Augmentation anchoring将多个strongly augmented的input输入进模型, 并鼓励每个输出都相近于同样input的weakly-augmented的prediction
- 为了实现strong augmentation, 提出了一个新的AutoAugment方法
- 实验表明, 本文方法significantly more data-efficient, 仅需要5到16分之一的数据达到相同的准确率. 值得关注的是, CIFAR-10中, 每类仅用了4个标签数据就达到了84.92%的准确率

Introdution

- SSL在标签数据有限的时候, 提供了有意义使用无标签数据提高模型性能的方法
- 近期的工作主要可以分为两大类 : **Consistency Regularization**和**Entropy Minimization**, MixMatch在一个unified loss function中集成了这些方法, 并取得了更好的效果
- 本文中, 提出两个可以readily集成进MixMatch框架的改进提升

  Distribution Alignment 使得模型预测的边缘分布与真实类别的边缘分布能够相匹配. 通过最大化模型输入和输出的mutual information, 提出一个相关的loss项.

  Augmentation Anchoring 使用augmentation anchoring替换在MixMatch中的consistency regularization. 对于每个无标签的样本, augmentation anchoring首先生成简单的weakly augmented version, 然后再生成很多strongly augmented version. 模型对于weakly augmented样本的预测被当做所有strongly augmented版本guessed label的basis. 同时, 为了声场strong augmentation, 提出一种AutoAugmentation的改进, CTAugment. 不同于AutoAugment, CTAugment, 随着模型的训练单独学习增强策略, 使得在SSL问题的setting中比较适合

Backgorund

- Consistency Regularization : 大多数SSL方法都是基于consistency regularization的. 该方法使模型对于perturbed的输入保持prediction相同, 首先在2014提出. 主要工作有Regularization with stochastic transformations and perturbation和π-Model, 其他工作也有使用perturbs adversarially或者dropout. 最常见的perturbation是加入domain-specific的数据增强. loss function中测量perturbed和non-perturbed输入的consistency通常使用MSE或者cross-entropy
- Entropy Minimization : 认为无标签数据应该被用于确保每一类well-separated. 可以通过使得模型对于无标签数据输出low-entropy的预测分布来实现. 比如可以显式的添加一个loss项去最小化模型对于无标签数据输出的的entropy. 与这个想法相关的有self-training方法, 比如Pseudo-Label. Pseudo-Label使用对于相同无标签数据 (augmented) 的预测类别作为hard target, 这样就隐式的最小化预测的entropy

- Standard Regularization : 不光是在SSL中, 在过多参数中对模型进行约束通常是有用的. 这样的regularization通常可以用于有标签和无标签数据. 比如标准的weight decay, 这个在MixMatch中也有提到
- Other Approaches : 除了上面3个主流分类, 还有一些其他方法, 比如transductive. transductive的想法认为无标签数据应该是被分配到sufficiently similar有标签的数据的标签

ReMixMatch

介绍Distribution Alignment和Augmentation Anchoring, 并介绍如何将这些方法集成进MixMatch, 整个ReMixMatch流程如下

Algorithm 1 ReMixMatch algorithm for producing a collection of processed labeled examples and processed unlabeled examples with label guesses (cf. Berthelot et al. (2019) Algorithm 1.)

Input: Batch of labeled examples and their one-hot labels $\mathcal{X} = \big((x_b, p_b) : b \in (1, \dots, B)\big)$, batch of unlabeled examples $\mathcal{U} = \big(u_b : b \in (1, \dots, B)\big)$, sharpening temperature T, number of augmentations K, Beta distribution parameter α for MixUp.
for $b = 1$ **to** B **do**
  $\hat{x}_b = \text{StrongAugment}(x_b)$ // *Apply strong data augmentation to* x_b
  $\hat{u}_{b,k} = \text{StrongAugment}(u_b); k \in \{1, \dots, K\}$ // *Apply strong data augmentation* K *times to* u_b
  $\tilde{u}_b = \text{WeakAugment}(u_b)$ // *Apply weak data augmentation to* u_b
  $q_b = p_{\text{model}}(y \mid \tilde{u}_b; \theta)$ // *Compute prediction for weak augmentation of* u_b
  $q_b = \text{Normalize}(q_b \times p(y) / \tilde{p}(y))$ // *Apply distribution alignment*
  $q_b = \text{Normalize}\big(q_b^{1/T}\big)$ // *Apply temperature sharpening to label guess*
end for
$\hat{\mathcal{X}} = \big((\hat{x}_b, p_b); b \in (1, \dots, B)\big)$ // *Augmented labeled examples and their labels*
$\hat{\mathcal{U}}_1 = \big((\hat{u}_{b,1}, q_b); b \in (1, \dots, B)\big)$ // *First strongly augmented unlabeled example and guessed label*
$\hat{\mathcal{U}} = \big((\hat{u}_{b,k}, q_b); b \in (1, \dots, B), k \in (1, \dots, K)\big)$ // *All strongly augmented unlabeled examples*
$\hat{\mathcal{U}} = \hat{\mathcal{U}} \cup \big((\tilde{u}_b, q_b); b \in (1, \dots, B)\big)$ // *Add weakly augmented unlabeled examples*
$\mathcal{W} = \text{Shuffle}\big(\text{Concat}(\hat{\mathcal{X}}, \hat{\mathcal{U}})\big)$ // *Combine and shuffle labeled and unlabeled data*
$\mathcal{X}' = \big(\text{MixUp}(\hat{\mathcal{X}}_i, \mathcal{W}_i); i \in (1, \dots, |\hat{\mathcal{X}}|)\big)$ // *Apply* MixUp *to labeled data and entries from* $\mathcal{W}$
$\mathcal{U}' = \big(\text{MixUp}(\hat{\mathcal{U}}_i, \mathcal{W}_{i+|\hat{\mathcal{X}}|}); i \in (1, \dots, |\hat{\mathcal{U}}|)\big)$ // *Apply* MixUp *to unlabeled data and the rest of* $\mathcal{W}$
return $\mathcal{X}', \mathcal{U}', \hat{\mathcal{U}}_1$

- Distribution Alignment

  enforce无标签的数据匹配有标签数据的分布. 首次用于SSL, 示例图如下:

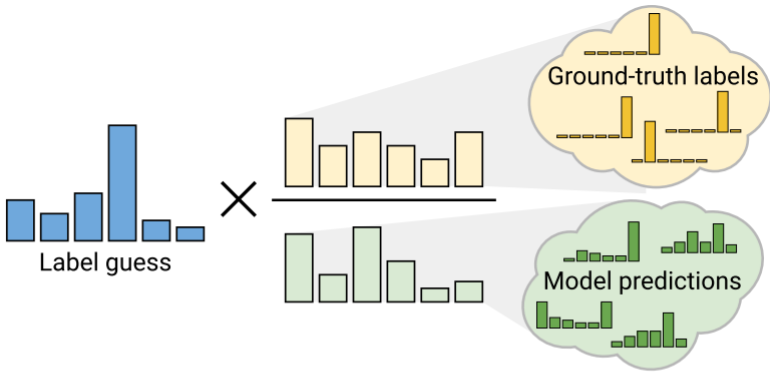

Figure 1: Distribution alignment. Guessed label distributions are adjusted according to the ratio of the empirical ground-truth class distribution divided by the average model predictions on unlabeled data.

希望在每一个batch的基础上进行distribution alignment, 同时也不引入额外的loss项或者任何敏感的超参数. 因此在训练中, 我们维持一个模型在无标签数据上预测的running average. 给定一个在无标签数据的模型预测 q, 本文通过 $p(y) / \tilde{p}(y)$ 的比值来对q进行缩放, 然后在renormalize缩放后的distribution到一个valid probability distribution. 用公式表达为$\tilde{q}$ = Normalize($q \times p(y) / \tilde{p}(y)$), 同时Normalize($x_i$) = x_i / $\sum_j x_j$, $\tilde{q}$用于作为多无监督数据的标签, 通常还会进行sharpening处理. 在实际中, 本文通过最后的128个batch计算$\tilde{p}(y)$作为模型在无标签眼样本上输出的moving average, 同时也在训练过程中估计有标签数据的$p(y)$, 更好的估计可能会提高性能, 但是本文中没有更多深入

- Improved Consistency Regularization

  Consistency Regularization在绝大多数SSL方法中都有使用. 在MixMatch中, 对于无标签数据使用K=2次的augmentation, 然后将他们预测的均值做为他们的guessed label.

  近期的一些工作表明, 使用更强形式的增强能够显著的提高consistency regularization的性能. 在MixMatch中仅使用简单的flip-and-crop增强策略, 因此作者也尝试使用AutoAugment替换其中的weak augmentation, 但结果发现训练不能收敛. 因此本文中提出**Augmentation Anchoring**方法, 其基本想法是, 使用对weak augmented无标签数据的预测作为更多strongly augmented version的guessed label

  Augmentation Anchoring 作者假设结合AutoAugment的MixMatch是unstable的, 因为MixMatch会把K个增强样本的预测一起取平均作为其guessed label. 然而stronger augmentation可能导致disparate prediction, 所以取所有增强样本的均值不是一个有意义的guessed label. 为了解决这个问题, 本文对无标签的数据首先产生anchor, 即使用weak augmentation对数据做增广. 然后在使用CTAugment做strong augmentation, 并且使用weak augmentation的guessed label作为K个strong augmentation的label, 如图所示

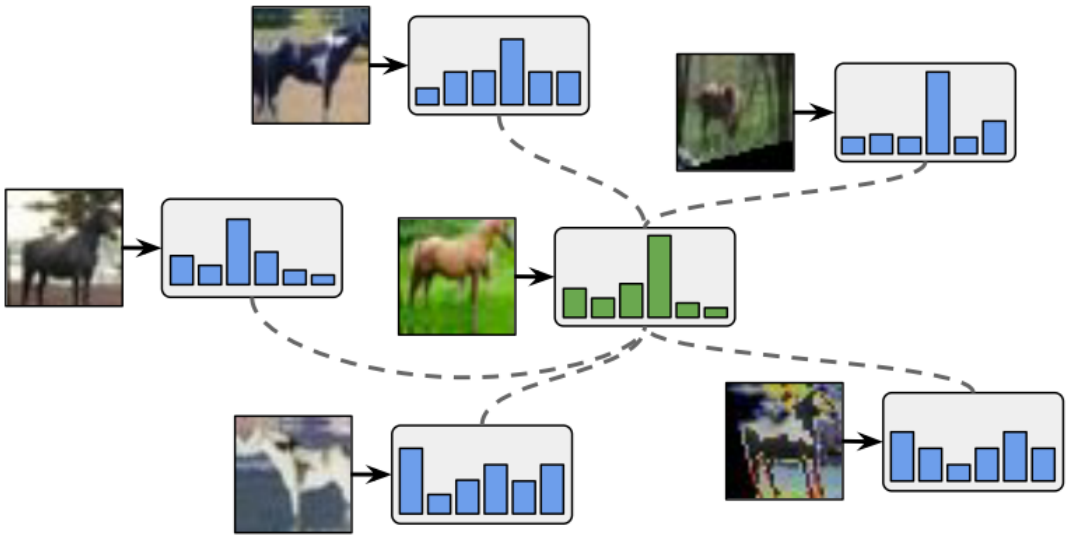

Figure 2: Augmentation anchoring. We use the prediction for a weakly augmented image (green, middle) as the target for predictions on strong augmentations of the same image (blue).

在使用Augmentation Anchoring实验中, 本文发现该方法使得我们能用cross-entropy够换MixMatch的无标签数据的MSE loss, 在维持稳定性的同事也能简化实现

Control Theory Augment

AutoAugment通过学习一个数据增强策略来产生high validation set accuracy, 一个augmentation policy由一系列的transformation-parameter组成. 重要的是, AutoAugment policy是通过supervision的方式学习到的, 因此实际应用AutoAugment会是methodologically problematic, 由于SSL本事就是仅有少量label的setting

为了解决AutoAugment的这个问题, RandAugment被提出, uniformly randomly采样transformation, 但是需要调节超参数, 因此同样也是methodologically problematic的

本文中提出一种高性能的方法, CTAugment. 与RandAugment相似, CTAugment会uniformly randomly采取transformation, 但在训练过程中, 对于每个transformation会dynamically infers magnitudes. CTAugment不需要在一个proxy task上进行优化, 也没有敏感的超参数, 因此可以直接整合进MixMatch中. CTAugment会学习到产生被正确分类图片的likelihood, 这个过程类似于Fast AutoAugment的density-matching, 使得augmented validation set能够和training set的density相匹配

初始, CTAugment把每个transformation的每个参数划分为distortion magnitude bins. 选取一个m作为每个bin中参数的权重, 初始化设置为1, 这些权重决定被用于决定哪些magnitude bin被采用. 在每个training step, 对于每个image, uniformly and randomly选择两种transformation. 为了增强图片数据, 对于每个transformation的每个参数产生一个modified set of bin weights $\widehat{m}$, 如果m_i > 0.8, 则$\widehat{m} = m_i$, 不然=0

Experiments

所有实验使用相同的codebase和model. 按照Realistic semi-supervised learning的evaluation setting, 并与VAT MeanTeacher和MixMatch进行比较

- CIFAR-10 & SVHN

	CIFAR-10			SVHN		
Method	250 labels	1000 labels	4000 labels	250 labels	1000 labels	4000 labels
VAT	36.03±2.82	18.64±0.40	11.05±0.31	8.41±1.01	5.98±0.21	4.20±0.15
Mean Teacher	47.32±4.71	17.32±4.00	10.36±0.25	6.45±2.43	3.75±0.10	3.39±0.11
MixMatch	11.08±0.87	7.75±0.32	6.24±0.06	3.78±0.26	3.27±0.31	2.89±0.06
ReMixMatch	6.27±0.34	5.73±0.16	5.14±0.04	3.10±0.50	2.83±0.30	2.42±0.09
UDA, reported*	8.76±0.90	5.87±0.13	5.29±0.25	2.76±0.17	2.55±0.09	2.47±0.15

Table 1: Results on CIFAR-10 and SVHN. * For UDA, due to adaptation difficulties, we report the results from Xie et al. (2019) which are not comparable to our results due to a different network implementation, training procedure, etc. For VAT, Mean Teacher, and MixMatch, we report results using our reimplementation, which makes them directly comparable to ReMixMatch's scores.

- STL-10

Method	Error Rate
SWWAE	25.70
CC-GAN	22.20
MixMatch	10.18 ± 1.46
ReMixMatch (K=1)	6.77 ± 1.66
ReMixMatch (K=4)	6.18 ± 1.24

Table 2: STL-10 error rate using 1000-label splits. SWWAE and CC-GAN results are from (Zhao et al., 2015) and (Denton et al., 2016).

- Ablation study

Ablation	Error Rate	Ablation	Error Rate
ReMixMatch	5.94	No rotation loss	6.08
With K=1	7.32	No pre-mixup loss	6.66
With K=2	6.74	No dist. alignment	7.28
With K=4	6.21	L2 unlabeled loss	17.28
With K=16	5.93	No strong aug.	12.51
MixMatch	11.08	No weak aug.	29.36

Table 3: Ablation study. Error rates are reported on a single 250-label split from CIFAR-10.