Generalizing from a few examples-A survey on few-shot learning

2020.06.07 许茜

一、读论文的目的：了解 few-shot learning 的现有的工作，对自己后面研究的领域有一个整体的认知，了解其他相关领域是做什么的，清楚自己要做的研究处于整个框架下的哪个分支哪个点。

笔记中提出了四个疑问，目前都已解决，未来得及将手写草稿整理成文字。

二、论文笔记

文章架构：

·PART I: 给出 few-shot learning 的定义

·PART II: 指出 few-shot learning 面临的核心问题--经验风险最小化不可靠

·PART III: 对现有的工作进行分类，分类的标准是从 data、model、algorithm 层面解决核心问题

·PART IV: 提出未来的研究方向、可应用的场景

Part I：给出 few-shot learning 的定义

1. few-shot leaning 定义

机器学习定义：面对任务 T，计算机程序可以从 experience E 中学习提升性能。

few-shot learning（FSL）：属于机器学习问题的一种，E 仅包含任务 T 相关的、有限数量的有监督信息

现存的 FSL 问题主要是有监督学习问题和 few-shot 强化学习。FSL algorithm 是一种寻找最优参数的优化策略。FSL 方法是将可获得的有监督信息（E）和一些先验知识结合在一起。

2. few-shot learning 三种典型场景：

使机器学习更能模仿人的学习，仅通过小样本即可获得知识 如 字符生成

有监督的信息很难获取（涉及隐私、安全、伦理等原因） 如 新药物的发现

减少收集大量有监督信息的 gathering effort 和 conputational cost 如 图片分类

3. 一些和 FSL 相关又略有不同的机器学习问题

- Weakly supervised learning：从仅包含弱监督信息的 E 中学习。

  根据是否有 oracle 和人为干预可分类为半监督学习（无干预）和主动学习（有干预）

  与 FSL 的异同：

  （1）弱监督学习仅包含分类和回归，FSL 除此外还包含增强学习问题

  （2）弱监督学习主要使用无标签数据作为额外的信息，FSL 利用多样的先验知识，如预训练模型，其他领域的有监督数据等，不局限于无标签数据。

  总结：当先验知识是无标签数据，任务是分类或回归时，FSL==weakly supervised learning

- Imabalanced learning

  疑问 1：谈及与 FSL 区别时，训练和测试时，不均衡学习选择所有可能的 y，FSL 把其他的 y 作为先验知识，如何理解？

- Transfer learning：先验知识从源任务迁移到 few-shot 任务。常用于 FSL。

- Meta-learning：通过提供的数据集和 meta-learner 从多个任务中获得的 meta-knowledge 来提升在新任务上的性能。

PART II: 指出 few-shot learning 面临的核心问题--经验风险最小化不可靠

1. FSL 的核心问题：经验风险最小化不可靠

整体的误差进行分解：

$$\mathbb{E}[R(h_I) - R(\hat{h})] = \underbrace{\mathbb{E}[R(h^*) - R(\hat{h})]}_{\mathcal{E}_{\text{app}}(\mathcal{H})} + \underbrace{\mathbb{E}[R(h_I) - R(h^*)]}_{\mathcal{E}_{\text{est}}(\mathcal{H}, I)},$$

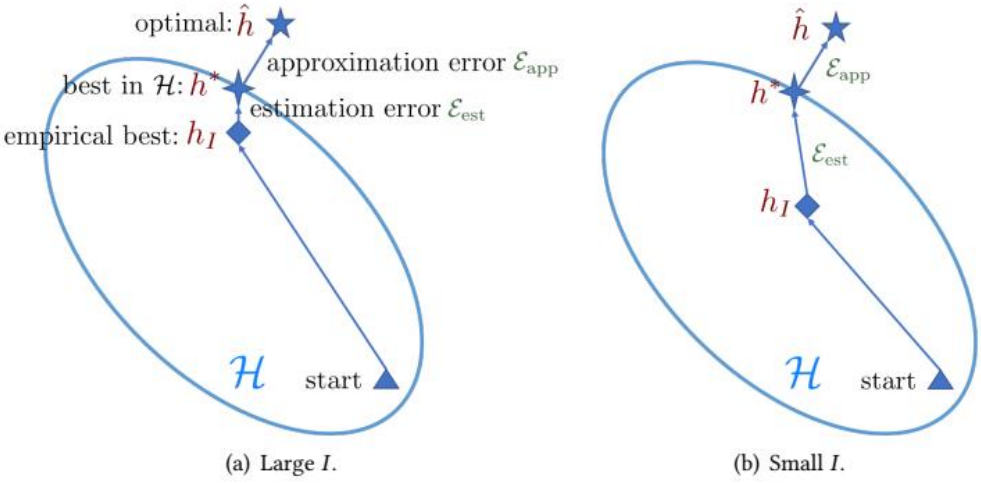


Fig. 1. Comparison of learning with sufficient and few training samples.

可以用大量样本来减小 E_{est}（H,I），但 few-shot 没有大量的样本，所以 few-shot 任务的经验风险最小化不再可靠。这是核心问题。

2. 为什么本文从三方面（数据、模型、算法）对当前 few-shot learning 工作进行分类?

E_{app} 是全局最优解和假设空间最优解的误差，E_{est} 是假设空间中，经验风险和期望风险的误差。可以看出整体误差受 H（假设空间）和 I（D_{train} 中的 examples 数量）的影响。因此根据用先验知识加强哪一方面进行分类，可以从三方面改

进减少误差提升性能-扩充 D_{train}、减小 H、使用先验知识优化在 H 中寻找最优参数θ的算法。

PART III: 从 data、model、algorithm 层面解决核心问题

DATA: 这一部分介绍的现有办法主要用于图像

1. 分类准则：根据用什么样本来转换并加入 D_{train} 分为—从 D_{train} 中转换样本、从弱标签或无标签数据集中转换样本、从相似的数据集中转换样本。
2. Data 层面的三种 FSL methods 特点：

category	input (x, y)	transformer t	output $(\tilde{x}, \tilde{y})$
transforming samples from D_{train}	original (x_i, y_i)	learned transformation function on x_i	$(t(x_i), y_i)$
transforming samples from a weakly labeled or unlabeled data set	weakly labeled or unlabeled $(\tilde{x}, -)$	a predictor trained from D_{train}	$(\tilde{x}, t(\tilde{x}))$
transforming samples from similar data sets	samples $\{(\hat{x}_j, \hat{y}_j)\}$ from similar data sets	an aggregator to combine $\{(\hat{x}_j, \hat{y}_j)\}$	$(t(\{\hat{x}_j\}), t(\{\hat{y}_j\}))$

3. 从弱标签或无标签数据集转换样本：从弱标签或无标签的大数据集中选择有目标 target 的样本，怎么选？exemplar SVM、label propagation、progressive strategy…
4. 总结（什么情形用哪种办法？）：

   收集、计算成本高→unlabeled/weakly labeled

   Unlabeled/weakly labeled 难收集，但 few-shot 类有相似的类别→similar datasets

   无可用样本，只有一些学习的转换器→从原始 D_{train} 里转换

5. 缺点：数据扩充准则都是为每个数据集量身定制的，迁移性差，而且现有办法主要用于图像。

MODEL: 利用 E 中的先验知识限制假设空间 H 的大小，增强经验风险最小化可靠性，减小过拟合风险

1. 分类准则：根据 prior knowledge 不同分为—multitask learning、embedding learning、learning with external memory、generative modeling
2. Multitask learning

   多个相关的 tasks，联合学习多个相关任务的 task-genertic 信息和 task-specific 信息。联合学习所以参数会受其他任务的限制，根据任务参数如何受限制分为 parameter sharing 和 parameter tying。

   ·Parameter sharing：多个任务共享一些参数。

   ·Parameter tying： 使不同的任务的参数比较相似。
3. Embedding learning

   将样本嵌入更低维的空间中，令相似的样本更近，不相似的样本更易区分，在底维空间中可以构建更小的假设空间 H。

   嵌入函数是从先验知识和 D_{train} 的 task-specific 信息中学习得到。

   实现：需学习测试样本 x_{test} 的嵌入函数 f，训练样本 x_i 的嵌入函数 g，衡量 $f(x_{test})$和 $g(x_i)$相似度的函数 s。$f(x_{test})$和哪个 $g(x_i)$更相似，就分给 x_{test} 对应的 y_i。

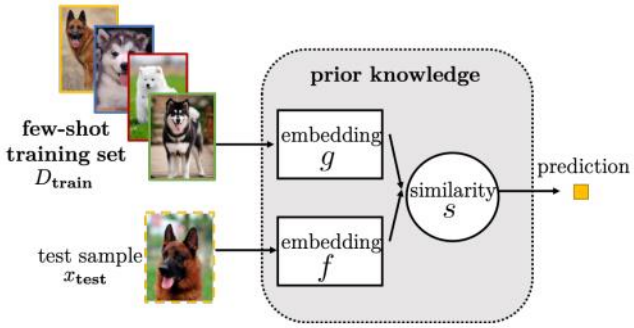
   NOTE：f 和 g 可以一样，但设置不同函数性能更好。

   根据嵌入函数的参数是否根据不同的任务变化分为：

   - Task-specific embedding model

     通过不同任务独特的信息学习一个量身定制的嵌入函数。
   - Task-invariant embedding model

     从有不同 output 的大数据集中学习一个通用的嵌入函数。虽然是大数据集，但是在训练的时候模拟了 few-shot 的情形。



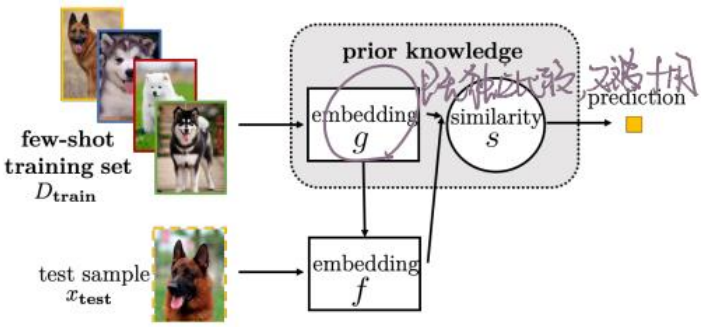
最近有一些通过 meta-learning 方法的 task-invariant 模型：

· Matching Nets: meta-learn 不同的嵌入函数 f 和 g.

· ProtoNet：不比较 $f(x_{test})$和 $g(x_i)$,而是比较 $f(x_{test})$和每个类别中 $g(x_i)$的均值。

- Hybrid embedding model（结合了前两种）

  从 Dtrain 中的 task-specific 信息来调整先验知识中学习到的 task-invariant 嵌入模型。



与 task-invariant 的对比：图中 g 既是 f 的参数，又要和 f 比较相似度。

4. Learning with external memory

从训练集中抽取知识存入 external memory，对每个 x_{test} 用 external memory 中对应的值加权平均表示。$f(x_{test})$不直接用于表示 x_{test}，而是用于查询最相似的 memory slots，取加权平均。根据 memory 的功能，FSL methods 分为：

- Refining Representations： MANN meta-learns 嵌入函数 f，将相同类别的样本映射入相同的值。那么相同类的样本共同限制类的表示，类的表示可以看做 ProtoNet 中的求均值。The surprised-based memory module 只有在不能很好地表示 x_i 的时候才会更新 memory。The abstract memory 用两个 memory，一个从大量标注数据中抽的键值对，一个从 few-shot 中抽最有用的信息来修正抽到的值。

  因为 few-shot 的样本少，所以很难保留在 M 中，有个解决办法是当 memory 满了擦掉最久未被更新的值，但是这个缺点是因为 few-shot 样本少，还是会不怎么更新，还是会被优先删掉。
- Refining Parameters：Meta-Networks 从多数据集中 meta-learn 到的慢权重和 task-specific 的快权重。

疑问 2：为什么限制了 x_{test} 的表示会使得假设空间减小？x_{test} 和训练过程又没有什么关系?

5. Generative modeling：在先验知识的帮助下，从观测到的 x_i 估计概率分布 p(x). 通常分布中有一个隐变量 z。z 是从其他数据集中学习的先验知识。通过 z 的分布与 Dtrain 结合约束 $p(\theta|x)$，从而约束了 H。

   根据 z 的含义不同分为：

   - Decomposable Components：z 是可分解部件，与其他任务的 sample 有共享部件，比如说眼睛鼻子，可以很简单的学习到部件的模型。将部件组合后确定属于哪个分类。

- Groupwise shared prior：相似的任务有相似的先验。孟加拉虎图像少，用橘猫的来训练学习。
- Parameters of inference networks：z 不具有语义，使用一些大规模数据集训练推理网络近似估计后验概率中的某个难求的部分。

6. 总结（什么时候用哪种方法？）：

存在相似任务→multitask learning

存在每个类样本都充足的大数据集→embedding 方法

如果可用 memory network→external memory 缺点是空间和计算的花费以及 memory 有大小限制

如果除了 FSL 需要实现生成或者重构的任务→generative model

ALGORITHM:（1）提供一个较好的初始化参数（2）学习一个优化器输出搜索步骤

1. 分类准则： 根据先验知识如何影响搜索策略分为 Refining existing parameters、refining meta-learned parameters、learning the optimizer

Table 7. Characteristics for FSL methods focusing on the algorithm perspective.

strategy	prior knowledge	how to search θ of the h^* in $\mathcal{H}$
refining existing parameters	learned θ_0	refine θ_0 by D_{train}
refining meta-learned parameters	meta-learner	refine θ_0 by D_{train}
learning the optimizer	meta-learner	use search steps provided by the meta-learner

2. Refining existing parameters：从其他相关的任务中学习一个θ_0
   - Fine-tuning existing parameter by regularization

     简单的通过正梯度下降调整θ_0会过拟合，解决这一问题的方法如下：

     （1） Early stopping：当验证集上效果无提升了就停止学习

     （2） Selectively updating θ_0 together：只更新θ_0的一部分

（3） Updating related parts of θ_0 together：将θ_0的元素分组，根据相同的更新信息联合更新分组

（4） Using a model regression network：将少样本训练的参数值映射到大量样本训练得到的参数值。

- Aggregating a set of parameters

  相关任务中学习得到的模型参数 aggregate 后，直接使用或通过 D_{train} 调整一下。

  （1） 使用 unlabeled dataset 训练得到的模型如何面对新任务调整：从无标签数据中预训练函数聚相似数据类和分离不相似数据，再用神经网络来调整。

  （2） 使用 similar datasets：先用 similar datasets 训练，再用新类的 features 替换掉相似类的 features，然后对于新类仅调整分类的阈值。

- Fine-tuning existing parameter with new parameters

  额外设置参数δ用于考虑上当前任务的 D_{train}。

3. Refining meta-learned parameters：从任务的集合中 meta-learn 一个θ_0，这些任务服从相同的任务分布。

   与 2 的区别：2 的θ_0是固定的，3 的θ_0是不断优化的。

   Model-Agnostic Meta-learning（MAML）方法的一些改进：

   - Incorporating task-specific information：不再一个初始化通用，而是从一个初始化参数的子集中选{θ_0}。
   - Modeling the uncertainty of using a meta-learned θ_0：examples 少，预测能力不好，对这个不确定性来建模。

- Improving the refining procedure：用正则化来校正梯度下降的方向。

疑问 3：上述 2/3 中标红的两个任务，含义相同吗?

疑问 4：任务的集合服从相同的任务分布，是必须从同一个大数据集中构造不同的训练任务吗?

4. Learning the optimizer：学习一个优化器直接输出更新$\Delta\theta$

根据第 t-1 轮的迭代中计算出的误差信号直接输出更新的$\Delta\psi_{t-1}$，$\psi_t=\psi_{t-1}+\Delta\psi_{t-1}$，得到的$\Delta\psi_t$用于充当误差信号在下一次迭代中学习

5. 缺点：三种方法，第一种牺牲精度提高速度，第二三种依赖于元学习，跨粒度的元学习问题存在，需要考虑如何避免负迁移。