Generalizing from a Few Examples: Few-Shot Learning Taxonomy

Can machines think? – Alan Turing raised this question in 1950.
He explained that "The idea behind digital computers may be explained by machines intended to carry out any operations which could be done by a human computer".

Machine Learning Algorithm began with the idea of "learn from experience". The machines are feed with more examples (data), then the system learns to perform the task quickly and accurately. The performance metrics helps the system to fine-tune and improve.

Few-Shot learning (FSL) is a type of machine learning problem where the experiences (or data) limited with supervised information for the target task completion. In notation, N-Way K-shot classification refers to N classes each with K examples, D_{train} = K*N examples. If there is only one sample in supervised information, it is called *one-shot learning*. If there is no example in supervision, then the FSL is called *zero-shot learning*.

AI has speedup in pace and has beat humans in many fields. AlphaGo defeats human champions in the old game Go; Residual networks (ResNet) obtains better image classification in challenges like ImageNet. Computer programs parse and generate new handwritten characters from fewer samples.

In FSL, the number of available training samples is less. Hence there is a high chance for the model to overfit, and empirical risk minimizer is no longer reliable. Let see how to tackle this problem using three different approaches –

- Data – Using augmentation to increase the data volume
- Model – Using prior knowledge to constrain the complex problem
- Algorithm – Using prior knowledge which parameterizes the best hypothesis

The below diagram shows the taxonomy of FSL

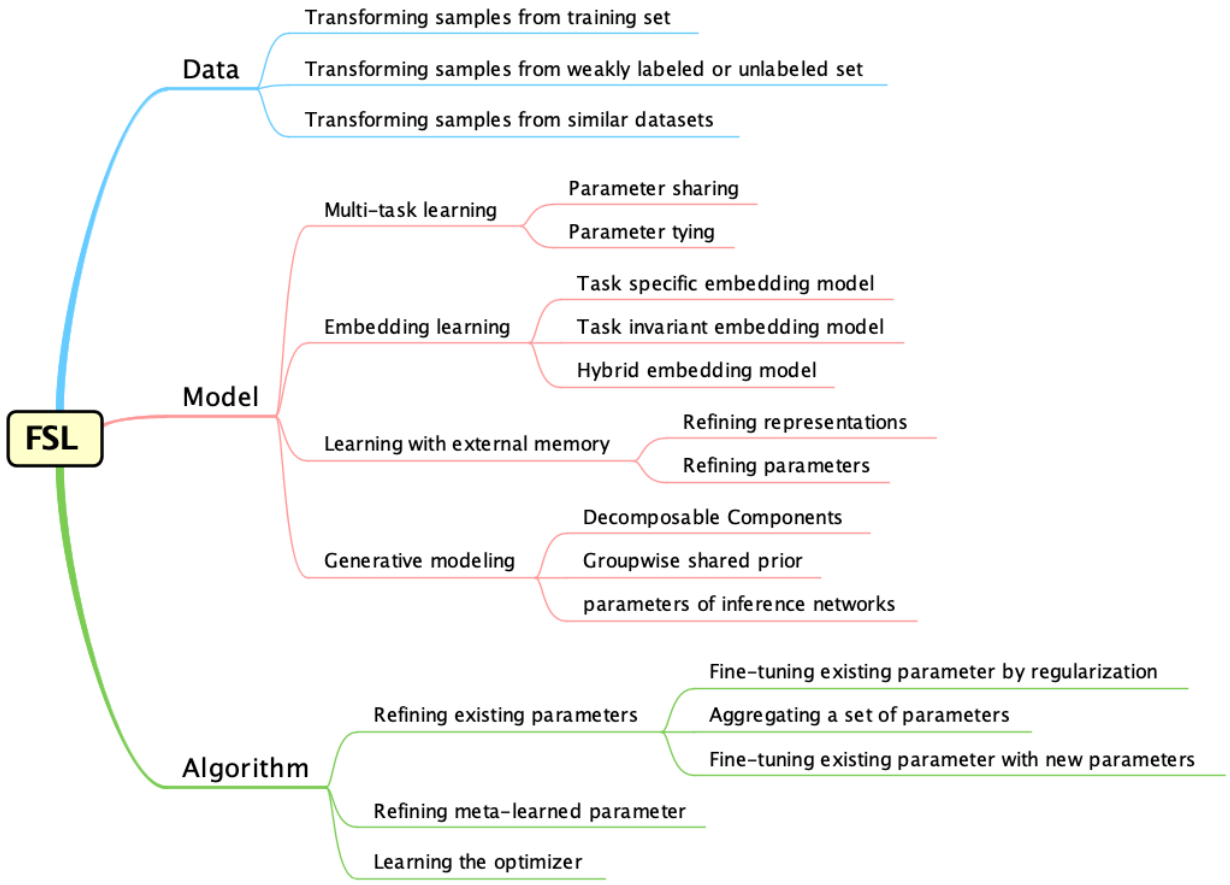

Figure 1: Taxonomy of FSL
Reference: A survey on Few shot learning https://arxiv.org/abs/1904.05046

Data

They are using prior methods of data augmentation to enrich the dataset such as: translation, flipping, shearing, scaling, reflection, cropping and rotation. However, these data augmentations are dataset specific, may make it hard to apply to all datasets. Moreover, it is unlikely that human can enumerate all possible invariance.

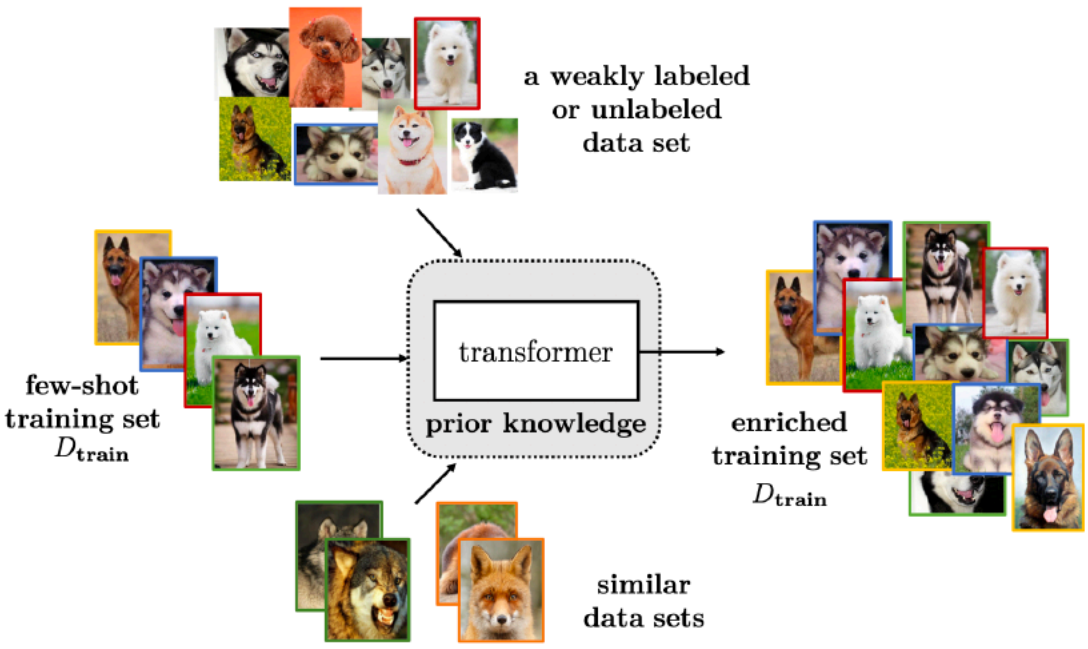

Figure 2: Solving FSL by data augmentation

Table 1: Characteristics for FSL methods focusing on the data perspective

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 $(\bar{x}, -)$	a predictor trained from D_{train}	$(\bar{x}, t(\bar{x}))$
transforming samples from similar data sets	samples $\{(\hat{x}_j, \hat{y}_j)\}$ from similar data sets	an aggregator to combine $\{(\hat{x}_j, \hat{y}_j)\}$	$(t(\{\hat{x}_j\}), t(\{\hat{y}_j\}))$

Transforming Samples from Training set

As per one research, the learnt geometric representations from one class is applied iteratively to similar classes. A set of auto-encoders learns interclass variability from similar classes. A single transformation function is known to transfer variation between sample pairs learned from the other classes. In further research, continuous attribute subspace is used to add attribute variations in space.

Transforming Samples from a Weakly labeled or Unlabeled Data set:

This strategy augments target label from a large dataset, which is weakly labeled or unlabeled. Example: Gestures of the speaker in a presentation. Not all, gestures are annotated explicitly. Such a data set contains large variations of samples and collecting such data set is more accessible as no human effort needed. In earlier research, exemplar SVM is used to identify the target label. In recent times, a progressive strategy used to select informative unlabeled samples. The selected samples are then assigned pseudo-labels and used to update the CNN.

Transforming Samples from Similar Data Sets:

It is aggregating or Adapting input-output pairs from a similar but larger data set. The aggregation process based on a similarity measure between the samples. A generative adversarial network (GAN) designed to generate indistinguishable synthetic aggregated from a data set of many samples. It has two generators, one maps samples of the few-shot class to the large-scale class, and the other maps samples of the large-scale class to the few-shot class.

Overall, a large number of weakly supervised or unlabeled samples exist for the target task (or class), but few-shot learning preferred because of the high cost of gathering annotated data and/or computational cost

Model

FSL methods in this section manage to learn by constraining hypothesis space to a smaller value based on prior knowledge. The empirical risk minimizer is then more reliable, and the risk of overfitting is reduced.

In terms of what prior knowledge is used, methods belonging to this category can be classified as four types.

Table 2: Characteristics for FSL methods focusing on the model perspective

strategy	prior knowledge	how to constrain $\mathcal{H}$
multitask learning	other T's with their data sets D's	share/tie parameter
embedding learning	embedding learned from/together with other T's	project samples to a smaller embedding space in which similar and dissimilar samples can be easily discriminated
learning with external memory	embedding learned from other T's to interact with memory	refine samples using key-value pairs stored in memory
generative modeling	prior model learned from other T's	restrict the form of distribution

Multitask learning

With multiple related tasks, Multi-task learning learns these tasks simultaneously by exploiting both task-generic or task-specific information. According to how the task parameters are constrained, we divide methods in this strategy as (i) parameter sharing; and (ii) parameter tying. Parameter sharing strategy directly shares some parameters among tasks. In the two task networks share the first few layers for the generic information and learn different final layers to deal with different outputs. Parameter tying encourages parameters of different tasks to be similar. If there is a CNN for the source task and another one for the target task. Layers of these two CNNs are aligned using some specially designed regularization terms.

Embedding learning

Embedding learning has the following key components: (i) a function $f(x_{test})$ which embeds test sample (ii) a function $g(x_i)$ which embeds training sample and (iii) a similarity function which measures the similarity between $f(x_{test})$ and $g(x_i)$

According to whether the parameters of embedding functions f and g vary across tasks, we classify these FSL methods as using an (i) task-specific embedding model; (ii) task-invariant embedding

model; and (iii) hybrid embedding model, which encodes both task-specific and task-invariant information.

Learning with External Memory

Learning with external memory extracts knowledge from training samples and stores it in external memory. Each new sample x_{test}, then represented by a weighted average of contents extracted from the memory. This limits x_{test} to represent by contents in the memory, and thus essentially reduces the size of hypothesis space. As each x_{test} is defined as a weighted average of values extracted from the memory, the quality of key-value pairs in the memory is important. According to the functionality of the memory, FSL methods in this category can be subdivided into two types: (i) Refining Representations (ii) Refining Parameters

Generative Modeling

Generative modeling methods estimate the probability distribution p(x) from the observed sample with the help of prior knowledge. Methods in this class can deal with many tasks, such as generation, recognition, reconstruction, and image flipping. According to the way latent space is represented, we group FSL generative modeling methods as three types: (i) Decomposable components (ii) Groupwise shared prior (iii) Parameters of Inference Networks.

When there exist similar tasks or auxiliary tasks, multitask learning can be used to constrain the hypothesis space of the few-shot task. However, note that joint training of all the tasks together is required. Thus, when a new few-shot task arrives, the whole multitask model has to be trained again, which can be costly and slow. Hence, generative modeling methods have high inference cost and are more difficult to derive than deterministic models.

Algorithm

When supervised information is rich, there are enough training samples to update θ and to find an appropriate step size α by cross-validation. However, in FSL, the provided few-shot D_{train} is not large enough, and the obtained empirical risk minimizer is unreliable.
Methods in this section use prior knowledge to influence how θ is obtained, either by (i) providing a good initialized parameter θ_0, or (ii) learning an optimizer to output search steps. In terms of how the search strategy is affected by prior knowledge, we classify methods in this section into three groups

Table 3: 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

Refining Existing Parameters:

Fine-tune parameters to improve regularization: Early stopping, Selective updating θ_0, updating related parts of θ_0 together, using a model regression network. As per the fine-tune strategy, we can choose either of these categories: Aggregating a Set of Parameters Fine-Tuning Existing Parameter with New Parameters

Refining Meta-Learned Parameter

Methods in this section use meta-learning to refine meta-learned parameter. It uses gradient descent to refine meta learner. A representative method is the Model Agnostic Meta-Learning (MAML), further research reveals the following improvements: Incorporating task-specific information, Modeling the uncertainty of using a meta-learned θ_0 , Improving the refining procedure.

Learning the Optimizer

Instead of using gradient descent, the steps in this method uses optimizer directly. Hence tuning step size is not needed, as algorithm does that automatically.

Overall, these methods need a lower computation cost to obtain a reasonable hypothesis. Important issues such as how to meta-learn across different granularities or different data sources (such as images versus texts) are still open. From this perspective, meta-learning and multi-tasks are similar, and so there is also a concern on how to avoid the negative transfer.

FUTURE WORK

Four critical directions for the further development of FSL, namely, (i) problem setups, (ii) techniques, (iii) applications and (iv) theories. Fine-tune methods may lead to overfitting. In various scenarios, the new tasks are arrived continuous; there is a chance for catastrophic forgetting. Another direction is to extend the AutoML methods to automated feature engineering, model selection and neural architecture search to FSL. There are more interesting fields in FSL applications: Computer vision, NLP, Robotics, Acoustic signal processing, and so on. Theories prove that FSL lack supervised information. Convergence of FSL is not fully clear. A more general analysis of the convergence of meta-learning methods will be highly useful.