Maria Doliashvili

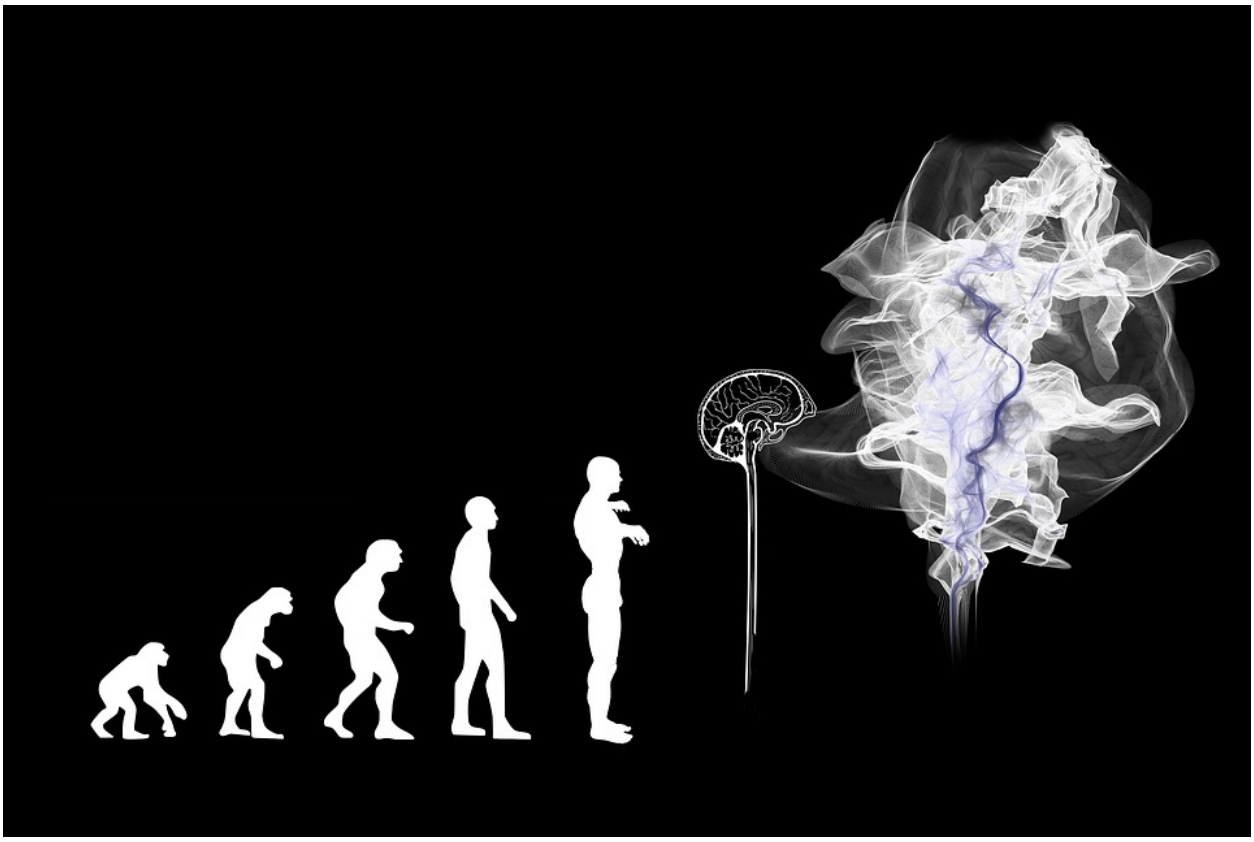
Learning to Learn - Artificial Evolution

Meta-Learning

Abstract

Learning to learn by solving many tasks is called meta-learning. The idea behind meta-learning for neural networks is to train a system on many tasks with the goal that it can solve new tasks and adapt to new environments quickly; by observing how different machine learning approaches perform on a wide range of learning tasks. Current artificial intelligence systems excel at constructing a single approach for a singular task, however fail to generalize and perform adequately when the task or the approach is being altered. There are several promising meta-learning approaches being developed and several success stories that will be surveyed in this literature review.

Keywords: Meta-learning, AutoML, Meta-knowledge, Hyperparameters, Few-Shot Learning



Introduction

"... a system that improves or discovers a learning algorithm"

(Hochreiter, Younger, & Conwell, 2001)

Human beings acquire knowledge about how to perform each task and re-use some of the knowledge in the future for performing new tasks or adapting to new environments. As the set of previously learned tasks increases, humans learn new skills more and more effectively. For example, learning to distinguish each animal from each other (like penguins from red pandas) only takes up to a few examples for training, while interacting with strangers can be an overwhelming experience for an infant, it becomes easily manageable for an adult. As for

artificial intelligence systems nowadays, they almost always learn how to perform a task starting from scratch each time for every new task and environment. None of the previous approaches that worked well are being reused. Trainings for building a good machine learning model usually take a large amount of data. For example machine translation models cannot hold a dialogue or any kind of conversation and DeepMind's AlphaGo while beating a master player Li Sidol in board game Go, cannot help Boston Dynamics robots in parkour (Feng, Whitman, Xinjilefu, & Atkeson, 2014; Silver et al., 2016; Wu et al., 2016).

When we (humans) build machine learning models, we usually use our previous experiences of building them and remember and reuse the approaches/models that worked well on similar tasks. We use settings that previously worked while we hand tune the parameters, etc. Basically we are training ourselves on how to build a machine learning model. The main task of meta-learning can also be phrased as: Is it possible to create a system that will be able to automate the process of creating optimal models? And can these systems be more efficient than human beings? How can we let AI models to be as adaptive to new environments and re-use the previous learning knowledge the same way humans do?

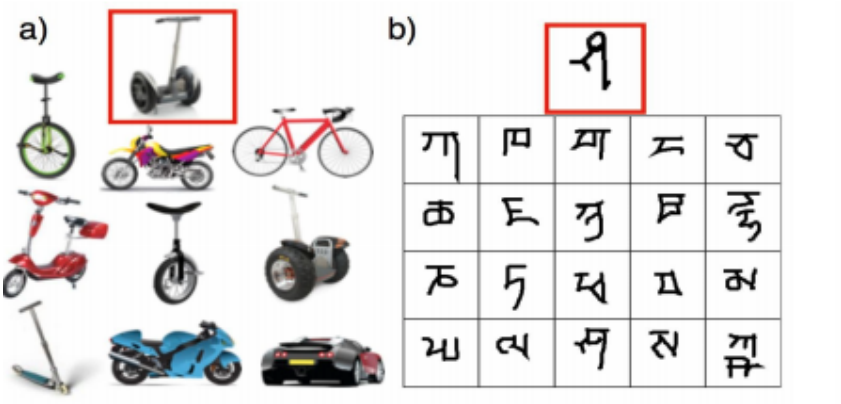
There are several techniques attempting to address the above questions. For creation of intelligent agents that are able to perform several tasks and acquiring several skills, it would be inefficient if we allocate the time and resources needed for learning each skill/task from scratch. Instead aiming to learning to learn the agents should not consider each task separately from each other, the tasks should be the training cases simultaneously for the model. The adaptation process should be done reusing the previous experience and it

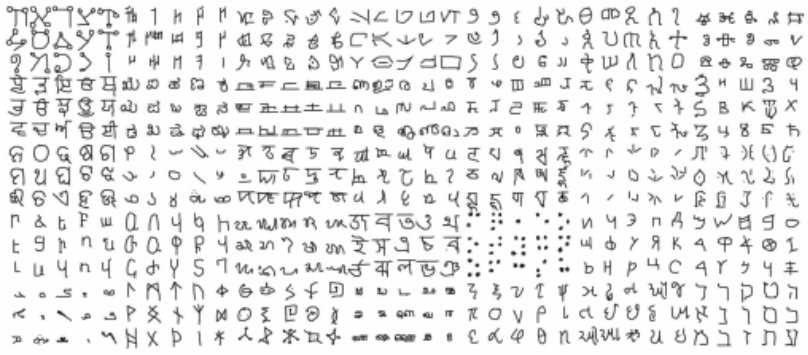
should make it possible to spend only a little time on training and with limited information about the new task or the environment that it did not encounter during the training time. The adapted model should be able to complete the new tasks. Basically, we can also look at the current approaches to meta-learning as a type of deep learning where instead of training cases we have training tasks and instead of test cases we have test tasks. Therefore meta-learning should enable us to create models that will be able to solve more complicated and interdisciplinary tasks in the future by providing more adaptive models and by doing so allowing us to be one step closer to artificial general intelligence.

The first meta-learning approaches are from the 1980s, though there are several remarkable papers from the recent years that made the topic more popular. Recent work includes techniques using hyperparameter and neural architecture search, few shot image recognition and reinforcement meta-learning, etc. (Bengio, Bengio, & Cloutier, 1990; Schmidhuber, 1987). The methods reviewed here are known to perform well on a variety of tasks and claim to achieve state of the art, or outperform humans in the corresponding tasks.

Few-Shot Learning

Lake et al. published a paper in 2015, challenging machine learning models to learn a task given only few instances of the task (Lake, Salakhutdinov, & Tenenbaum, 2015).





Lake et al, 2013, 2015

Lake suggested that humans can identify novel two-wheel vehicles from a single picture and draw a character in a new alphabet, after seeing a few examples, while machine learning models cannot generalize any concepts from few examples. Lake et al. also introduced Omniglot - a handwritten version of MNIST dataset, with 1623 (20 examples) character classes (Lake, Salakhutdinov, & Tenenbaum, 2013). Deep learning models showed its possibility to learn to learn from the few examples on Omniglot dataset in 2016 (Lake, Ullman, Tenenbaum, & Gershman, 2017). Few shot object recognition became a popular topic after the initial success with meta-learning. That is essentially selecting hyper-

parameters and parameters that can adapt to a new task without overfitting given the few shots, and resulted in one of the top benchmarks for few shot learning.

Some worth-mentioning meta-learning methods for few-shot learning are Lee et a.l using embeddings and SVM; edge-labeling graph neural network for few-shot learning by Kim et al. which represents each image as a node in a graph and there are features per edge based on how similar the nodes are to each other; task agnostic meta-learning for few-Shot learning by Jamal et al.; meta-transfer learning for few-shot learning by Sun et al., etc (Jamal & Qi, 2019; Kim, Kim, Kim, & Yoo, 2019; Lee, Maji, Ravichandran, & Soatto, 2019; Sun, Liu, Chua, & Schiele, 2019).

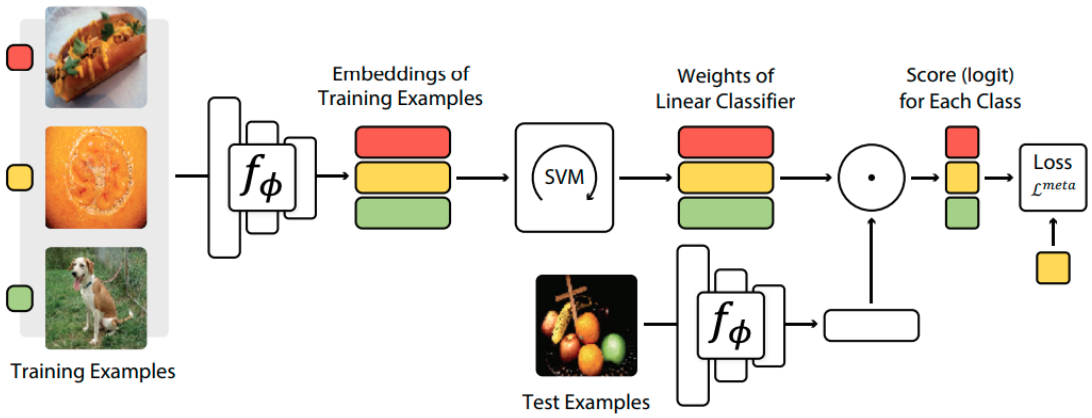


Figure 1. **Overview of our approach.** Schematic illustration of our method MetaOptNet on an 1-shot 3-way classification task. The meta-training objective is to learn the parameters ϕ of a feature embedding model f_ϕ that generalizes well across tasks when used with regularized linear classifiers (*e.g.*, SVMs). A task is a tuple of a few-shot training set and a test set (see Section 3 for details).

Neural Architecture Search

Neural Architecture Search (NAS) is related to both AutoML and hyperparameter optimization (Elsken, Metzen, & Hutter, 2018; Zoph & Le, 2016). NAS is a way of optimizing designing a neural network architecture. NAS shows its capability to design networks that

are as good as or even outperform hand-designed architectures for image classification (Zoph, Vasudevan, Shlens, & Le, 2018), semantic segmentation (Chen et al., 2018) etc. We can formulate the task as follows: Find a new network architecture with only a few bits to specify: Learn it on a small dataset, test it on a large one.

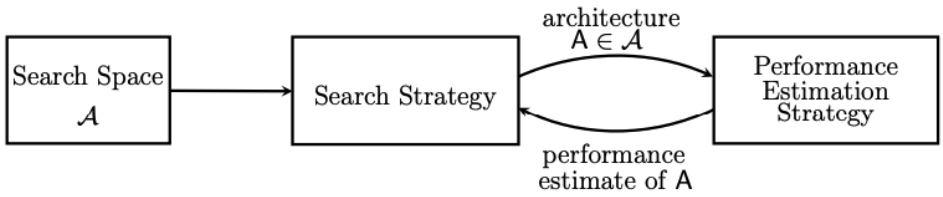


Figure 1: Abstract illustration of Neural Architecture Search methods. A search strategy selects an architecture A from a predefined search space $\mathcal{A}$. The architecture is passed to a performance estimation strategy, which returns the estimated performance of A to the search strategy.

The figure above, from NAS survey, illustrates three dimensions of NAS methods. Search space defines which architectures can be represented, considering the previous well-performed knowledge. Search strategy focuses on how to explore the search space. There is a threat of premature convergence when increasing search speed for a good architecture . Performance estimation strategy is focusing on the process of estimating performance, the recent methods are trying to reduce the cost of performance estimations, since it takes a lot of resources to do an end-to-end training and do the evaluation afterwards.

Training: Learner and Meta-Learner

The first challenge in meta-learning is to collect meta-data that describes the previous learning experience and the training tasks. This information includes: hyperparameter

settings, learning rate, model evaluations, network architecture and any other model or task parameters. These information is called mete-data. Examples of meta-learning tasks are:

- A game bot, that is able to quickly learn a new game. For example a bot that is trained on board games but not chess and can learn how to play chess by just learning it for a few iterations.
- A robot that is trained to navigate in a limited area and can adapt to finding routes in larger areas. For example an autonomous vehicle that is trained on a route inside of a company campus, but can adapt to driving outside of campus very quickly.
- Learn to recognize a new object on an image, just by looking at a few pictures. Etc.

Unlike meta-learning, standard machine learning approaches involve training on a big size of data for a single task and testing on smaller size of examples of the same task. In the meta-learning training there are two optimizers: the learner, which learns the tasks and the meta-learner, that is optimizing the learning of the learner. In the final step of the training we need to optimize for both learner and meta-learner.

There are three common approaches to meta-learning:

1) Metric based: learning efficient distance metric.
2) (Recurrent) Model based: using neural networks.
3) Optimization based: Optimizing model parameters for faster learning.

This summary of the main approaches of meta-learning is taken from the OpenAI blog:

	Model-based	Metric-based	Optimization-based
Key idea	RNN; memory	Metric learning	Gradient descent
How $P_\theta(y\|\mathbf{x})$ is modeled?	$f_\theta(\mathbf{x}, S)$	$\sum_{(\mathbf{x}_i, y_i)\in S} k_\theta(\mathbf{x}, \mathbf{x}_i) y_i$ (*)	$P_{g_\phi(\theta, S^L)}(y\|\mathbf{x})$

(*) k_θ is a kernel function measuring the similarity between $\mathbf{x}_i$ and $\mathbf{x}$.

Next we are gonna review classic models in each approach.

Metric Based Learning

Metric based approach is learning a metric space in which learning is more efficient. Metric based learning tries to represent a relationship between inputs and the task space (e.g. by using vector embeddings). It has been used for few-shot classification. The initial approaches for few-shot learning were purely based on metric learning, where the objective is to learn a mapping from images to a metric space in which images of the same category are grouped together and from different categories are far away from each other. Supposedly it will be correct for the unseen categories as well. It is somewhat similar to K- nearest neighbor (KNN) algorithm and kernel density estimation (Altman, 1992; Rosenblatt, 1956). kernel density estimation (KDE) is a non-parametric way to estimate the probability density function. The predicted probability over the known data samples is a weighted sum, where the weight is generated by a kernel function. To select an appropriate kernel is quite important for the metric based learning to work well. The same kernel function might not be useful for different tasks (personal experience). Therefore metric-learning aiming to learn the distance function over the objects is also depended on the task, since what is a good metric for one task might not be good for another.

In general, when we talk about metric meta-learning it is usually performed using gradient descent (or some other optimizer), the learner corresponds to a comparison scheme, e.g. nearest neighbors in the meta learned metric space.

If we are studying from a few pictures, then we could compare the images we are trying to classify with the images that the model already has. Comparing images in pixel space does not work. Therefore training a siamese network or performing comparisons in a learned metric space can be a better solution.

Convolutional Siamese Neural Network

Siamese time delay Neural Network was introduced in 1994 by Bromley at al (Bromley, Guyon, LeCun, Säckinger, & Shah, 1994). The construct is made of two identical twin networks, that are trained simultaneously and they both refer to the same embedding network that learns. Therefore output of the twin networks is jointly trained. The idea is to learn an embedding that conveys the relationship between pairs of the data points. Siamese Neural Networks for One-shot Image Recognition was introduced in 2015 by Koch at al (Koch, Zemel, & Salakhutdinov, 2015). The initial training is done for a verification task, it should allow to predict if two input samples are in the same class.

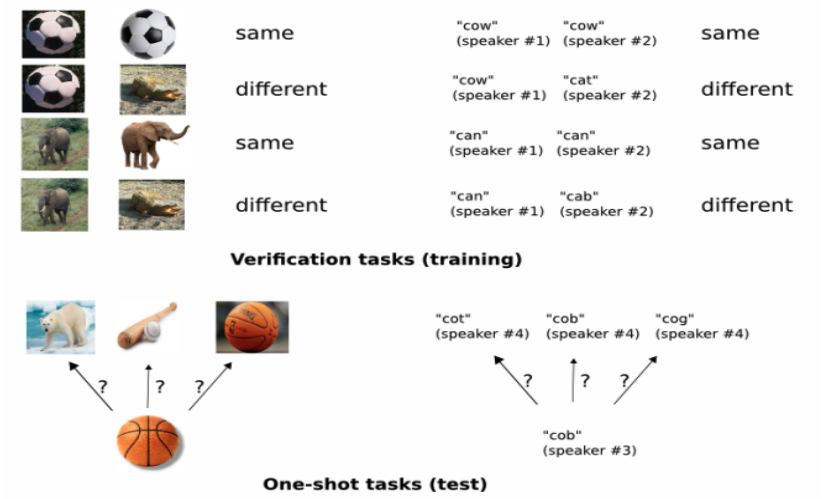


Figure 2. Our general strategy. 1) Train a model to discriminate between a collection of same/different pairs. 2) Generalize to evaluate new categories based on learned feature mappings for verification.

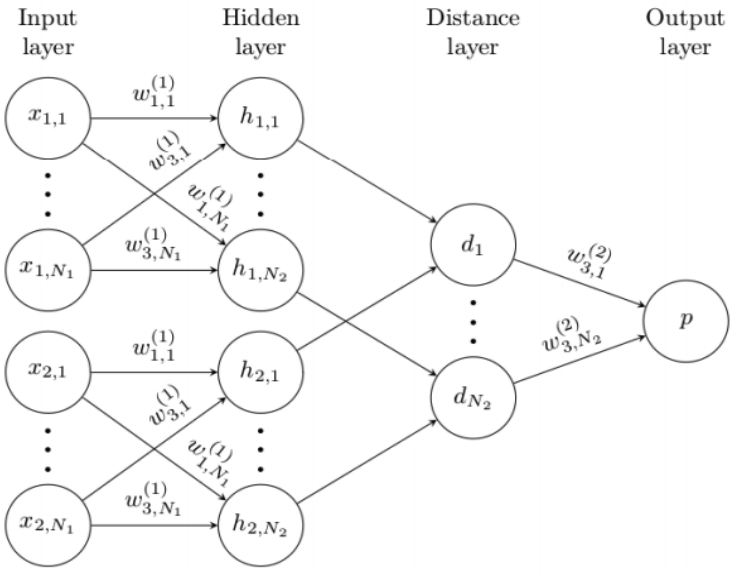


Figure 3. A simple 2 hidden layer siamese network for binary classification with logistic prediction p. The structure of the network is replicated across the top and bottom sections to form twin networks, with shared weight matrices at each layer.

The figures are from the original paper. The learned embedding should be generalized to measure the distance between the samples for unknown categories. However, as it is discussed in transfer learning section of this review, it is less likely to make the use of this approach if the new task and samples differ from the pre-training examples significantly.

Additionally, metric-based approaches include the matching networks by Vinals et al. that defines a probability distribution over output given the test sample, uses an attention kernel for measuring similarity between different samples, the attention kernel depends on two embedding functions for a test sample and for a support sample and the attention weight is a cosine similarity between their embeddings (Vinyals, Blundell, Lillicrap, & Wierstra, 2016); and relation network by Sung et al. where relationship is captured by a CNN classifier (Sung et al., 2018). During meta-learning, it learns a deep distance metric to compare a small number of images within episodes, similar to the few-shot setting. The below figure is from the original paper:

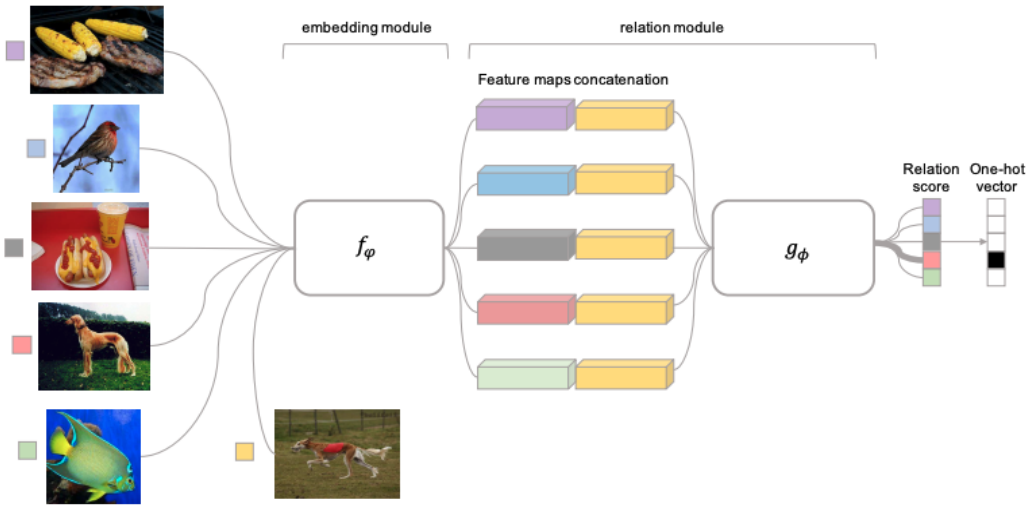


Figure 1: Relation Network architecture for a 5-way 1-shot problem with one query example.

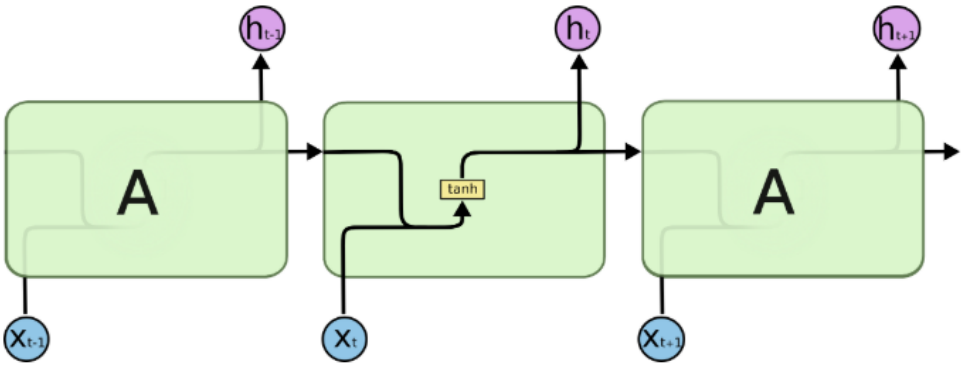
Finally, it is also important to mention Prototypical Networks. A prototype feature vector is defined for each class and the prototypical networks learn a metric space. Classification is done by computing distances to prototype representations of each class (Snell, Swersky, & Zemel, 2017).

Model Based Learning

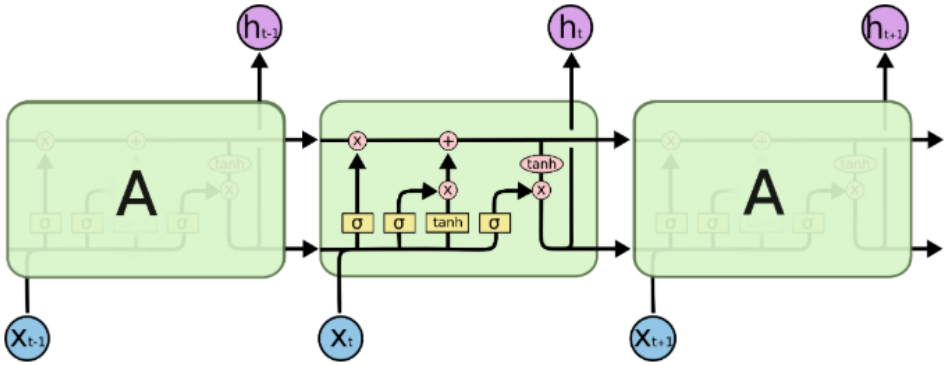
This approach is one of the most general approaches. It has been used in few-shot learning, regression and reinforcement meta-learning. Model based meta-learning depends on a model designed for fast learning. The model is supposed to updates its parameters quickly with only a few training steps. This rapid parameter update can sometimes be controlled by another meta-learner model.

Model training is usually done by using a recurrent neural network (RNN), e.g. long short term memory (LSTM) neural network that takes the dataset sequentially and then process the inputs from the task. Long short term memory networks are a special kind of RNN, capable of learning long-term dependencies, which were introduced by Hochreiter & Schmidhuber, and were refined and popularized by many people over the years (Hochreiter & Schmidhuber, 1997). LSTMs excel on a large variety of problems, and are widely used nowadays.

These figures illustrate the difference between standard RNN and LSTM:



The repeating module in a standard RNN contains a single layer.



The repeating module in an LSTM contains four interacting layers.

While the learner is studying using the LSTM, meta-learner can use gradient descent.

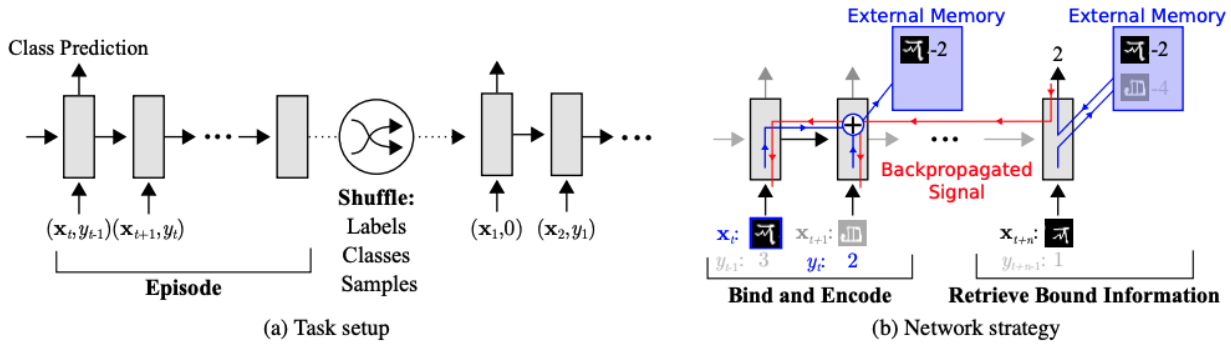
This approach is very flexible, though less efficient than other methods because the learner neural network starts learning from scratch

Memory-Augmented Neural Networks

Memory-Augmented Neural Networks (MANN) use external memory to save information about the learning process (Santoro, Bartunov, Botvinick, Wierstra, & Lillicrap, 2016). This class of learning approaches include Neural Turing Machines where memory is represented

as the Turing machine tape and the neural network has a control over the operation heads to read and write (Graves, Wayne, & Danihelka, 2014). Sometimes NTM is considered as a predecessor for MANN. Though in contrast with NTM, MANNs do not use location-based addressing to memory, instead it is using a new addressing schema called least recently used address. MANNs are extensively used in one-shot learning tasks, for Natural Language Processing tasks such as machine translation, language comprehension etc (Gulcehre & Chandar, 2017; Munkhdalai & Yu, 2016).

Using external memory solves the issue of forgetting the learned information over time. Therefore making the learning process more efficient. Using MANN for meta-learning means that the external memory should capture the information about the learned tasks so that it is easily accessible. (The following figure is from Santoro et al paper, 2016).



MANN uses two different weight vectors for read and write operations. The idea of using the least recently used memory location comes from the fact that, read and write to the location that was least recently used is based on the content, since the recently used memory location

is determined by the read operation, while the read operation is performed by content-based addressing. The content-based similarity is measured by cosine similarity.

Meta Networks

The main concept that allows Meta Networks to learn meta-data is the rapid generalization (Munkhdalai & Yu, 2017). Rapid generalization is defined as “allowing neural networks to learn and to generalize a new task or concept from a single example on the fly”. Rapid generalization across the tasks is done by using so called fast weights. One idea of learning weights faster than the standard gradient descent method is to use another Neural Network (NN) that predicts weights for the NN that is being trained on a task. In which case the loss gradients become the meta-data about the learner network, in fact it is used as one of the objectives. In the end, both slow and fast weights are used for the prediction. The architecture is shown in the following diagrams. A meta network has a base learner, a meta learner, and is equipped with an external memory:

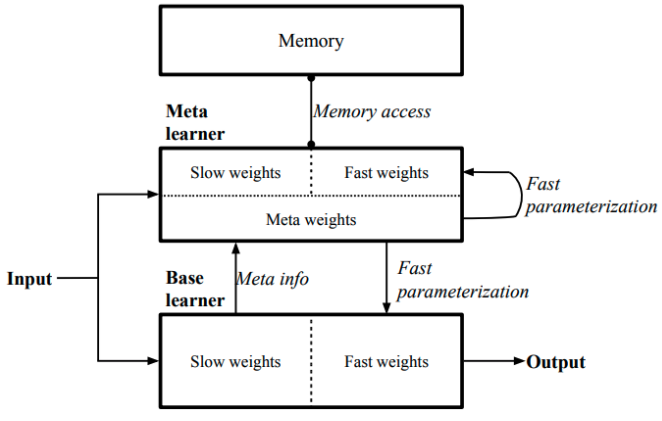


Figure 1. Overall architecture of Meta Networks.

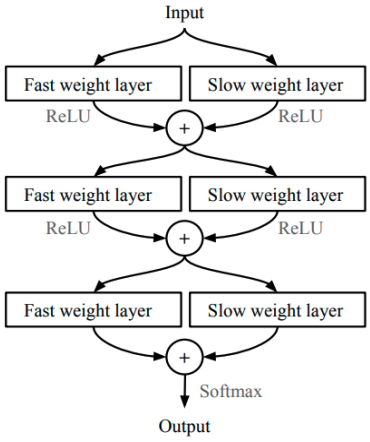


Figure 2. A layer augmented MLP

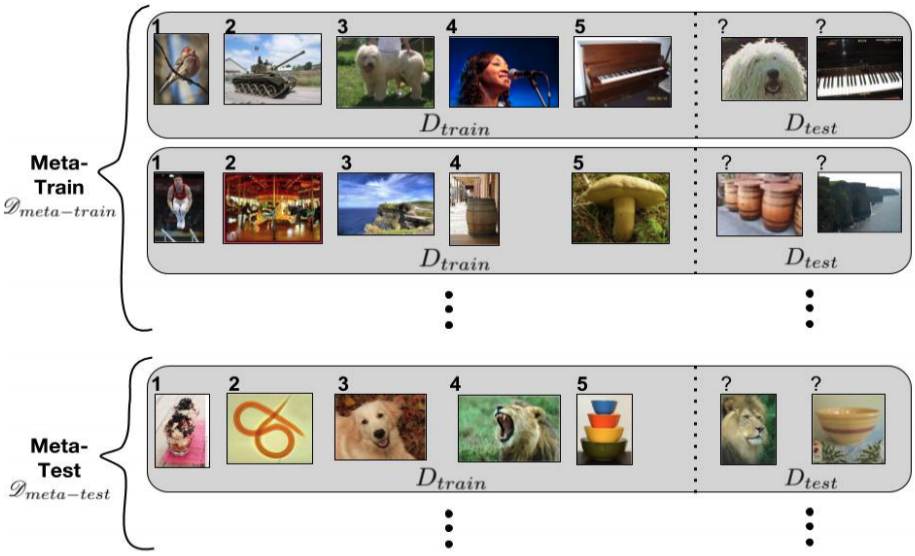
Optimization Based Learning

There are also approaches on learning an optimizer: Meta-learner, usually a neural network, learns to update/optimize the learner network so that the learner performs better on the task. The meta-learner is usually a recurrent neural network so that it can remember the updates of previous learner models. The meta-learner can be trained with reinforcement or supervised learning. Effectivity of this approach has been demonstrated for few-shot image classification by Ravi and Larochelle (Ravi & Larochelle, 2016). In other words, it is an attempt to optimize a deep neural network for few shot learning tasks, the main difficulty being that gradient-descent optimization requires many iteration steps.

LSTM Meta-learner

LSTM based meta-learner is using optimization based learning where the meta learner network is an LSTM. Meta-learner tried to learn the optimization in the few-shot regime while also generalizing on learning the parameters of the learner. LSTM neural network is picked as a meta-learner since "The meta-learner captures both short-term knowledge within a task and long-term knowledge common among all the tasks." (Ravi & Larochelle, 2016).

Meta-learning set-up for few-shot image classification:



The cell state of the LSTM is set to be the parameters of the learner. Updates for cell state in LSTM has similar form as updates in the learner's parameters. Also LSTM's ability of long term memory is very beneficial since knowing previous gradient values can lead to a more informed gradient update and faster convergence.

Model-Agnostic Meta-Learning (MAML)

Model-Agnostic Meta-Learning is a general meta-learning algorithm that is compatible with any model trained with gradient descent (Finn, Abbeel, & Levine, 2017). MAML is also quite general in terms of learned tasks, including classification, regression and reinforcement learning. MAML is directly training for a representation that can be quickly adapted to a new task with just a few steps. Therefore the search is for highly adaptable parameters.

MAML approach is shown in the below diagram, Θ is the set of parameters, when a gradient step is taken for a particular task the parameters Θ are close to the optimal parameters Θ*.

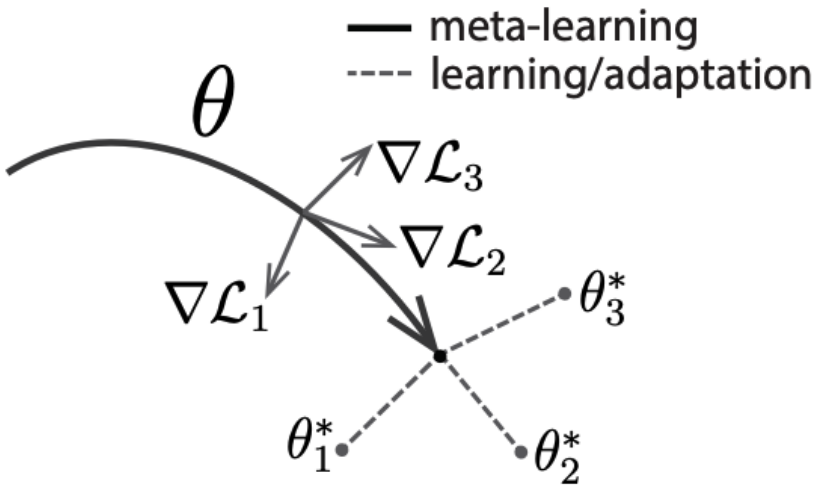


Figure 1. Diagram of our model-agnostic meta-learning algorithm (MAML), which optimizes for a representation θ that can quickly adapt to new tasks.

The approach is quite simple, the meta optimization step relies on the second derivatives. There is also a modified version of MAML that omits the second derivatives to make the computations more efficient it is called First-Order MAML or FOMAML.

Reptile

Another remarkable model-agnostic approach is Reptile by Nichol et al. from OpenAI (Nichol, Achiam, & Schulman, 2018) Reptile very similar to MAML. It also seeks an initialization for the parameters of a neural network. The network should be fine-tuned using a small amount of data from a new task. In contrast with MAML it is more computationally and memory efficient. Since MAML needs calculating second derivatives of the computation graph of the gradient descent algorithm and Reptile performs stochastic gradient descent.

Transfer Learning

Most of the discussed models are able to achieve a good result using meta-learning since the tasks they are trying to generalize for are somewhat similar to each other.

Humans are able to transfer knowledge across tasks. When can use the knowledge we acquired to learn one task while solving another task. The more related the two tasks are to each other it will be easier for us to learn a new task. E.g. we can re-use some knowledge from learning French language while learning Italian, but maybe not much of the knowledge is re-usable while learning Mandarin. Transfer Learning is a machine learning approach that makes use of the knowledge gained (a trained model) from one task to train for another task. It is a very popular approach in deep learning, since using a pre-trained model as a starting point can be faster and save some of the computational cost as well (Pan & Yang, 2009).

Lorien Pratt published a paper on transfer learning in 1993 and also published a paper applying transfer learning was also applied to cognitive science in 1996 (L. Pratt, 1996; L. Y. Pratt, 1993).The fundamental motivation for Transfer learning as far as meta-learning in the field of machine learning was discussed in a NIPS-95 workshop on “Learning to Learn” (Baxter et al., 1995).

Success stories about transfer learning probably start from initializing vision network weights using ImageNet pre-training. It is a fairly common practice for any new image classification task to re-use a model that has been pre-trained on ImageNet. This allows for image

recognition tasks to work well even if there is not much data available. However it is still necessary to have enough data for the pre-trained model to adapt to a new task.

Therefore transfer learning access the need since in deep learning enormous resources are required to train the models or the large and challenging datasets, or for some tasks there is not enough information available to train a deep neural network from scratch.

Discussing transfer learning alongside with meta-learning is important since they have to answer similar questions while designing approaches:

1) Is there any information in the first task transferable to target?
2) How should this information be transferred?

In both cases to answer these questions we need to take into account the similarities between the task parameters, feature spaces, machine learning approaches, etc.

There is approach of Meta-Transfer learning proposed for few-shot learning by Sun et al (Sun et al., 2019). The approach is learning to adapt a deep neural network for few shot learning tasks. In this case, meta-learning is referring to training multiple tasks. Transfer learning is used by learning scaling and shifting functions of weights for each task.

Analogies to other sciences

Analogy of meta-learning in biology is genetic evolution that learns the learning procedure encoded in genes and being executed in 'constructing' each individual and their brain. Humans are "constructed" based on DNA whilst Machine Learning models are constructed based on hyperparameters.

Human beings are being more adaptive to survive longer and collecting knowledge that makes it possible to perform complex previously undecided tasks.

For traditional machine learning models each training is the only way of acquiring the necessary knowledge to perform the tasks that can be analogical of training a child for a single purpose and not allowing to interact with any outside factors beside the given task.

By training models various times we (as human beings who created the models) acquire knowledge of what parameter settings work for which problems and try to construct the models based on our previous knowledge. The trial of creating models that can learn from the previous training experiences and re-use it for new tasks is called meta-learning since it in itself is the concept of learning of learning and as discussed in this review machine learning methods are able to perform meta-learning tasks.

Standard machine learning approaches to this problem the same way we approach to solving each task individually, by collecting the data, or in this case the meta-data of prior learning tasks and appropriate models and constructing models for eventually achieving optimal results in new tasks.

Based on previous work on the topic, it is definitely easier to to collect data and construct models if the new tasks are similar to the previously learned tasks.

Conclusion

Meta-learning, in other terms learning to learn is a science of designing machine learning models that can learn and re-use learned examples for the future trainings in new environments or tasks. Automatic methods are applied to meta-data i.e. hyperparameters etc. and outperform humans at designing new approaches or setting the training more efficiently. The goal is to decrease time and data requirements for learning a new concept by understanding and re-using the previously learned tasks and their parameters.

There are a variety of meta-learning approaches being developed, starting from early ideas in 1987 to current day papers. Meta-learning has impressive success stories e.g. with one-shot learning and neural architecture search.

Out of the three main areas of the meta-learning approaches: Metric based learning is one of the first and quite efficient though highly dependent on the similarity of the tasks and finding the correct metric space. Model based learning is the most general, it can work with a variety of different tasks but it comes with a cost of not always being able to perform well. Optimization based learning when one networks tried to optimize and update another network. The methods reviewed above achieve state-of-the-art performance on complicated

tasks and outperform human beings. It is discussed what is unique about their approach and what enables them to perform meta-learning.

Analogy of meta-learning in biology is genetic evolution that learns the learning procedure encoded in genes and being executed in 'constructing' each individual and their brain. Using genetic programming, evolutionary models are being improved based on meta evolution. The meta-learning can be improved on meta learning as well and so on (Bengio et al., 1990; Schmidhuber, 1987)

All the figures used in the literature review are from the original papers.

"We're mostly as good as our computers are" - a random reddit user.

References

Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, *46*(3), 175–185.

Baxter, J., Caruana, R., Mitchell, T., Pratt, L. Y., Silver, D. L., & Thrun, S. (1995). *NIPS 1995 workshop on learning to learn: Knowledge consolidation and transfer in inductive systems*.

Bengio, Y., Bengio, S., & Cloutier, J. (1990). *Learning a synaptic learning rule*. Université de Montréal, Département d'informatique et de recherche ….

Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., & Shah, R. (1994). Signature verification using a" siamese" time delay neural network. *Advances in Neural Information Processing Systems*, 737–744.

Chen, L.-C., Collins, M., Zhu, Y., Papandreou, G., Zoph, B., Schroff, F., … Shlens, J. (2018). Searching for efficient multi-scale architectures for dense image prediction. *Advances in Neural Information Processing Systems*, 8699–8710.

Elsken, T., Metzen, J. H., & Hutter, F. (2018). Neural architecture search: A survey. *ArXiv Preprint ArXiv:1808.05377*.

Feng, S., Whitman, E., Xinjilefu, X., & Atkeson, C. G. (2014). Optimization based full body control for the atlas robot. *2014 IEEE-RAS International Conference on Humanoid Robots*, 120–127. IEEE.

Finn, C., Abbeel, P., & Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 1126–1135. JMLR. org.

Graves, A., Wayne, G., & Danihelka, I. (2014). Neural turing machines. *ArXiv Preprint ArXiv:1410.5401*.

Gulcehre, C., & Chandar, S. (2017). Memory Augmented Neural Networks for Natural Language Processing. *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Hochreiter, S., Younger, A. S., & Conwell, P. R. (2001). Learning to learn using gradient descent. *International Conference on Artificial Neural Networks*, 87–94. Springer.

Jamal, M. A., & Qi, G.-J. (2019). Task Agnostic Meta-Learning for Few-Shot Learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 11719–11727.

Kim, J., Kim, T., Kim, S., & Yoo, C. D. (2019). Edge-Labeling Graph Neural Network for Few-shot Learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 11–20.

Koch, G., Zemel, R., & Salakhutdinov, R. (2015). Siamese neural networks for one-shot image recognition. *ICML Deep Learning Workshop*, *2*.

Lake, B. M., Salakhutdinov, R. R., & Tenenbaum, J. (2013). One-shot learning by inverting a compositional causal process. *Advances in Neural Information Processing Systems*, 2526–2534.

Lake, B. M., Salakhutdinov, R., & Tenenbaum, J. B. (2015). Human-level concept learning through probabilistic program induction. *Science*, *350*(6266), 1332–1338.

Lake, B. M., Ullman, T. D., Tenenbaum, J. B., & Gershman, S. J. (2017). Building machines that learn and think like people. *Behavioral and Brain Sciences*, *40*.

Lee, K., Maji, S., Ravichandran, A., & Soatto, S. (2019). Meta-learning with differentiable convex optimization. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 10657–10665.

Munkhdalai, T., & Yu, H. (2016). Reasoning with memory augmented neural networks for language comprehension. *ArXiv Preprint ArXiv:1610.06454*.

Munkhdalai, T., & Yu, H. (2017). Meta networks. *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 2554–2563. JMLR. org.

Nichol, A., Achiam, J., & Schulman, J. (2018). On first-order meta-learning algorithms. *ArXiv Preprint ArXiv:1803.02999*.

Pan, S. J., & Yang, Q. (2009). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, *22*(10), 1345–1359.

Pratt, L. (1996). *Reuse of neural networks through transfer*. Carfax Publishing Company.

Pratt, L. Y. (1993). Discriminability-based transfer between neural networks. *Advances in Neural Information Processing Systems*, 204–211.

Ravi, S., & Larochelle, H. (2016). *Optimization as a model for few-shot learning*.

Rosenblatt, M. (1956). Remarks on some nonparametric estimates of a density function. *The Annals of Mathematical Statistics*, 832–837.

Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., & Lillicrap, T. (2016). Meta-learning with memory-augmented neural networks. *International Conference on Machine Learning*, 1842–1850.

Schmidhuber, J. (1987). *Evolutionary principles in self-referential learning, or on learning how to learn: The meta-meta-... hook* (PhD Thesis). Technische Universität München.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Driessche, G. van den, … Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, *529*, 484–503.

Snell, J., Swersky, K., & Zemel, R. (2017). Prototypical networks for few-shot learning. *Advances in Neural Information Processing Systems*, 4077–4087.

Sun, Q., Liu, Y., Chua, T.-S., & Schiele, B. (2019). Meta-transfer learning for few-shot learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 403–412.

Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P. H., & Hospedales, T. M. (2018). Learning to compare: Relation network for few-shot learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1199–1208.

Vinyals, O., Blundell, C., Lillicrap, T., & Wierstra, D. (2016). Matching networks for one shot learning. *Advances in Neural Information Processing Systems*, 3630–3638.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., … Dean, J. (2016). Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *CoRR*, *abs/1609.08144*. Retrieved from http://arxiv.org/abs/1609.08144

Zoph, B., & Le, Q. V. (2016). Neural architecture search with reinforcement learning. *ArXiv Preprint ArXiv:1611.01578*.

Zoph, B., Vasudevan, V., Shlens, J., & Le, Q. V. (2018). Learning transferable architectures for

scalable image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 8697–8710.