learn learn

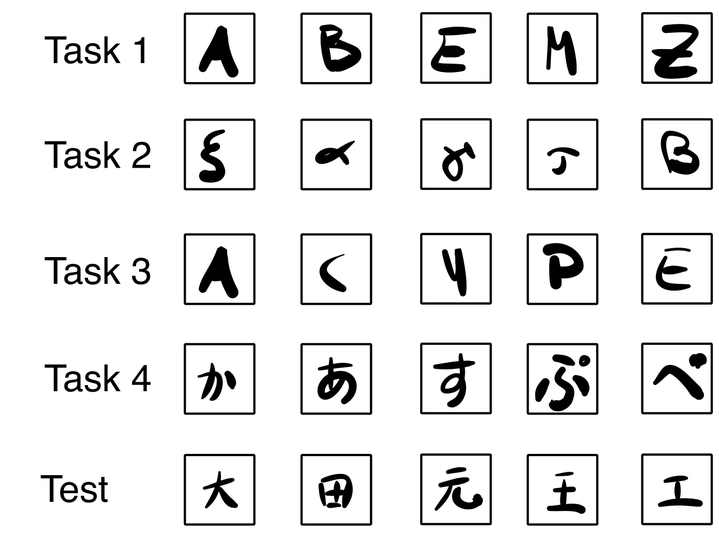
A Meta-Learning Library for Researchers

Summary

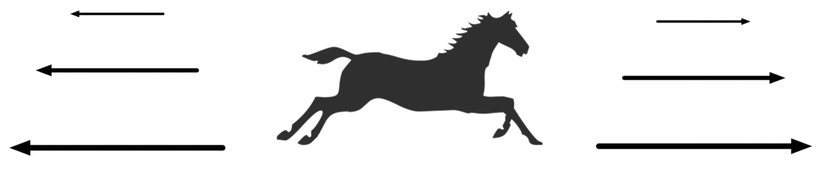
The goal of meta-learning is to enable agents to learn how to learn. That is, we would like our agents to become better learners as they solve more and more tasks.

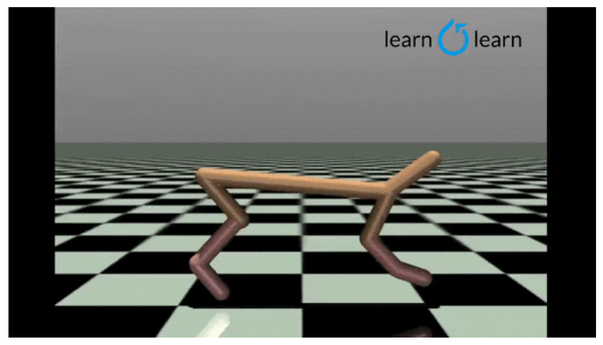
But meta-learning algorithms are difficult to implement, and standard benchmarks all use different reward functions or pre-processing steps.

learn2learn solves those issues by providing low- and high-level utilities to easily enable researchers (and practitioners!) to develop new meta-learning algorithms and compare them against well-tested implementations.



Meta Supervised Learning





Meta Reinforcement Learning

pip install learn2learn

```
import learn2learn as l2l

def fast_adapt(adaptation_data, evaluation_data, learner, adaptation_steps=5):
    for step in range(adaptation_steps):
        train_error = loss(learner, adaptation_data)
        learner.adapt(train_error)
    eval_error = loss(learner, evaluation_data)
    eval_accuracy = accuracy(learner, evaluation_data)
    return eval_error, eval_accuracy

train_dataset = l2l.vision.datasets.MiniImagenet(root='./data', mode='train')
train_dataset = l2l.data.MetaDataset(train_dataset)
train_generator = l2l.data.TaskGenerator(dataset=train_dataset, ways=5, tasks=20000)

# Create model
model = l2l.vision.models.MiniImagenetCNN(output_size=5)
maml = l2l.algorithms.MAML(model, lr=0.5, first_order=False)
opt = optim.Adam(maml.parameters(), 0.003)


for iteration in range(num_iterations):
    opt.zero_grad()
    # Compute meta-training loss
    learner = maml.clone()
    adaptation_data = train_generator.sample(shots=1)
    evaluation_data = train_generator.sample(shots=1,
                                             task=adaptation_data.sampled_task)
    evaluation_error, evaluation_accuracy = fast_adapt(adaptation_data,
                                                       evaluation_data,
                                                       learner)
    evaluation_error.backward()
    print('Meta-Train Loss', evaluation_error.item())
    print('Meta-Train Accuracy', evaluation_accuracy.item())
```

Features

- Support for meta-supervised and meta-reinforcement learning.
- High-level wrappers around popular meta-learning algorithms. (FO/MAML, MetaSGD.)
- Many low-level algorithms implemented.
- Universal task generators for supervised learning. (Works with any Dataset!)
- Standard benchmark definitions and downloaders.
- Unit and regression tested.
- Carefully nurtured documentation.

Use *any* module, *any* library, *any* dataset.

learnables/learn2learn

learn2learn.net