Script - Presentation (Code Demo)

Slide 1: In this presentation I will tell you about the most important pieces of code explaining the full codes. Key important algorithmic importance codes I am going to present. First we will start with maml (model agnostic meta learning). Then I will cover iMAML (the contribution of this paper) So lets begin

Slide 2: The MAML code is available on the given github repository for reference that provides a high level API for implementing all meta learning based algorithms using pytorch. From this presentation I will try to explain the code in parallel to the algorithm so that it is much easier to digest the algorithm. so first of all what we need to sample some batches so what we need to sample some batches so thank amount of batches from our table ability over over tasks and we want to sample a batch of tasks in the specific case so here you will need just probably a couple of lines of code to do to do this you can easily do it using this potage method api's and then what we do is to iterate over all tasks that we just sampled from P of T this is done in this for loop.

Slide 3: then next we are going to in the algorithm basically a sample sampling K data points but we don't know to do to do this operation since we are going to just take the support set we also take the query set at this stage this can be done later action but you can easily do it both of them here so for the moment we are using the support set we are passing these trained inputs to our model so this is our forward pass in the model and will give us some prediction to in logits that this represent our Y ok our predictions then we have the first inner loss you can see that we're still making the in turn loss on the on this data points D and this

represent estimating the inner loss on our support set okay we're still meeting their loss with a cross and so it is just a standard cross entropy not in special on our 20 large it's obtained here in three targets

Slide 4: now what we have to do is to compute the adapted parameters the gradient descent this is done in the algorithm with this line here line seven and we can do this in this way in PI torch first of all we are going to estimate the gradients on the inner loss on our model meta parameters and we want to keep the gradients the graph sorry we want to keep the graph so that later on on the other loop our graph is kept in that list we are going we are not going to reset it ok then we want to define a dictionary of parens and we are going to iterate here over our modern meta parameters we are going to take the gradients and what we do we are is just it's identical to this we are going to do a gradient step with a time step size okay

Slide 5: then here in the algorithm line 8 we are going to sample some data points but actually this in our specific set just represent a represented by the query points that we already have completed in line 8 of our algorithm so we don't need to do this and to be careful now here because for all tasks we are doing these other multiple tasks okay so in our case when we are going to do this in our Python implementation we have to call we have to submit the loss inside the for loop and we are going to not doing a simple assignment where but we are going to do a sum of all the losses over all the tasks okay so since we are doing a sum here we are accumulating in the outer loss all the possible task specific losses while we are accumulating because once you go out from this for loop your outer loss will contain all the possible losses over all the tasks that you that you add in your for loop and later on when we

are going to apply your backward pass we are going to apply this backward pass over all the lost a specific losses so this is a simple way to do an average or all the possible losses and once we have done the backward pass then we are going to apply our optimizer that in our case is just another optimizer with a specific step size and this is our other loss you see is accumulated here but then the gradient in the fourth step is applied out of this for for loop

Slide 6: Move slowly towards the target parameter value Default value for lam assumes learning rate determined by optimizer Useful for implementing Reptile """ # we can implement this with the regularization loss, but regularize around the target point # and with specific choice of lam=2.0 to preserve the learning rate of inner_opt

Slide 7:""" Given the gradient, step with the outer optimizer using the gradient. Assumed that the gradient is a tuple/list of size compatible with model.parameters() If flat_grad, then the gradient is a flattened vector """ # initialize the grad fields properly. # this would initialize required variables