Homework6

Jillian Myler

2024-04-06

What is the difference between gradient descent and *stochastic* gradient descent as discussed in class? (*You need not give full details of each algorithm. Instead you can describe what each does and provide the update step for each. Make sure that in providing the update step for each algorithm you emphasize what is different and why.*)

In regular gradient descent, the update step in minimizing theta involves the previous theta minus an alpha (stepsize) times the current parameters and running through all of the training data to best update the new parameters. In stochastic gradient descent, the update step runs through a tuple of randomly selected training points in optimizing the parameter selection instead of the full training set.

Consider the `FedAve` algorithm. In its most compact form we said the update step is $\omega_{t+1} = \omega_t - \eta \sum_{k=1}^{K} \frac{n_k}{n} \nabla F_k(\omega_t)$. However, we also emphasized a more intuitive, yet equivalent, formulation given by $\omega_{t+1}^k = \omega_t - \eta \nabla F_k(\omega_t); w_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k$.

Prove that these two formulations are equivalent.
(*Hint: show that if you place ω_{t+1}^k from the first equation (of the second formulation) into the second equation (of the second formulation), this second formulation will reduce to exactly the first formulation.*)

$\sum_{k=1}^{K} \frac{n_k}{n} (\omega_t - \eta \nabla F_k(\omega_t))$

Substituting ω_{t+1}^k

$\sum_{k=1}^{K} \left(\frac{n_k}{n} \omega_t - \frac{n_k}{n} \eta \nabla F_k(\omega_t) \right)$

Distributing η $\sum_{k=1}^{K} \frac{n_k}{n} \omega_t - \sum_{k=1}^{K} \frac{n_k}{n} \eta \nabla F_k(\omega_t)$

Distributing over addition

$\omega_t \sum_{k=1}^{K} \frac{n_k}{n} - \eta \sum_{k=1}^{K} \frac{n_k}{n} \nabla F_k(\omega_t)$

Pulling out ω_t and η

$\omega_t - \eta \sum_{k=1}^{K} \frac{n_k}{n} \nabla F_k(\omega_t)$

Since $\sum_{k=1}^{K} \frac{n_k}{n} = 1$

Now give a brief explanation as to why the second formulation is more intuitive. That is, you should be able to explain broadly what this update is doing.

The second formulation is more intuitive because it directly outlines the process. First, each client updates its local parameters, ω_t using its own gradient ∇F_k then sends its updated parameters ω_{t+1}^k to a centralized dataset.Then, the centralized dataset aggregates the updated parameters to create a global update ω_{t+1} by taking a weighted average $\frac{n_k}{n}$. This process allows the local knowledge to contribute to the overall/global model without being easily individually discoverable and allows for the maintenence of privacy at the local level while still contributing to the global model. Further, this second formulation is also far more intuitive because it takes the form of gradient descent much more clearly, such that you can clearly see the last set of parameters from the previous step minus an average of the gradient function and ω_t. It is very much more in the form of gradient descent but also accounting for the averageing step inherent in FedAve.

Explain how the harm principle places a constraint on personal autonomy. Then, discuss whether the harm principle is *currently* applicable to machine learning models. (*Hint: recall our discussions in the moral philosophy primer as to what grounds agency. You should in effect be arguing whether ML models have achieved agency enough to limit the autonomy of the users of said algorithms.*)

The harm principle places a constraint on personal autonomy because it limits what an individual should do to the actions that do not cause harm to others. In a more formal definition, Harm principle states:"Personal autonomy extends only insofar as exercising that autonomy does not cause objective harm to a moral agent." Currently, the harm principle is applied when considering the uses of an algorithm with respect to the developer. For example, there are already discussions surrounding limiting the use of an algorithm if it has the ability to cause harm through arbitrary discrimination or poses serious risk with regards to privacy breaches. In general, through the application of the harm principle, the developer should not be able to have full autonomy in using an algorithm for gain when it comes at the cost of harm to another individual. However, in response to when a model should be granted autonomy, it seems that it would be required that the model itself has some sort of gauge of if it is causing harm. It seems logical that if a human developer is subject to the harm principle, it should follow that the human is able to understand whether apriori (and should not use it in the first place) or a posteriori (and should stop using or modify the algorithm) what harm is being done. I would argue that for the algorithms themselves to be subject to scrutiny limiting their "autonomy", these models would be required to have a deeper understanding of not only how people interact and feel, but also some historic understanding in whether there are trends that are systematically hurtful and resultant from society that should not be perpetuated. Since it would be hard to say that algorithms are able to understand people's feelings, interactions, and history (to understand systematic wrongdoings) I do not think any models have the capacity to be considered autonomous beings. GPT does a strong job of mimickery, but it is still far from truly personable; thus, I would argue that at this current point in time, ML models are not able to be treated as ends in themselves (as they lack the aforementioned capacities) but are instead still a means to achieve an end (even if that end is to approach human interaction).