A ML, Silver Bullet ?

Report Issue (https://github.com/LeetCode-Feedback/LeetCode-Feedback/issues)

Once one starts to learn various kinds of Machine Learning (ML) algorithms, and how versatile they are to handle challenging tasks such as image recognition and language translation *etc*, one might be indulged to apply ML to every problem that they face, regardless of whether it fits or not. Because often the case, once one acquires a hammer, every problem might seem to be just another nail.

As a result, in this section, we would like to stress on some negative notes of ML. Like all other solutions, ML is **no silver bullet**.

> Like humans, ML models make mistakes.

For instance, one might notice that sometimes Facebook fails to tag a face from a photo. Unfortunately, people seem to accept the current state-of-the-art that ML algorithm is usually not 100% accurate. One can probably defend for ML algorithms, with the argument that the problems that ML deals with are indeed difficult, even for humans, *e.g.* image recognition. However, it is contrasting to the general conception that machines make no mistake or at least less than humans. For a moment (before 2012), people could easily claim the championship of ImageNet (http://www.image-net.org/) challenge with a model of 75% accuracy. One should bear in mind that the challenge was considered to the Olympic game in the domain of image recognition. So one can consider the results in ImageNet challenge as the *state-of-the-art* in the domain. Yet till now (2018), still no model can achieve 100% of accuracy. In general, a ML model that can reach ~80% accuracy, is considered to have a decent performance. ***Therefore, in the scenarios where the accuracy of the algorithm is critical, one should carefully examine their decision of adopting ML algorithms.***

> It is hard, if not impossible, to correct the mistakes made by ML, in the case-by-case manner.

One might wonder, if we consider each mistake made by a ML model as a bug in the software, can't we just correct them one by one so that we can boost the accuracy step by step? The answer is no. The reason is twofold: *1).* In general, one does not explicitly manipulate a ML model, but apply a ML algorithm with a given data to generate a model. To improve a model, we either improve the algorithm or the quality of data, without modifying the model directly. *2).* Even we can manipulate a generated ML model afterward, it is not intuitive how one can change the output of the ML model in certain 'erroneous' cases, without impacting the other correct cases. For instance, for a decision-tree model, the output of the model is the conjunction of branching conditions at each node, following the path from root to leaf. One can change certain branching conditions in the nodes to alter the decision of erroneous cases. However, this change would also impact the outputs for every case that passes through the modified nodes. In summary, one can not treat the mistakes made by a ML model simply as bugs in the software. ***It requires a holistic approach to improve the model, rather than patching the model case by case.***

> It is hard, if not impossible, to reason about certain ML models.

So far, one has learned that ML model makes mistakes and it is hard to correct the mistakes case by case. Perhaps things aren't so bad, since at least we could explain why it makes mistakes, such as the decision-tree model. Yet, in some cases, particularly for the ML models with neural networks, we cannot really reason about the models, *i.e.* it is hard to interpret the model, to identify the key parameters within a model. For instance, there is a state-of-the-art neural network model called ResNet (https://arxiv.org/abs/1512.03385) **[1]** which achieves 96.43% accuracy in the ImageNet (http://www.image-net.org/) challenge. The ResNet-50 model consists of 50 layers of neurons, including 25.6 million of parameters in total. Each of the parameters contributes to the final output of the model. Either the output is correct or not, it is the millions of parameters behind the model that accounts for. It is hard to attribute any logic to each of the parameters individually. ***Therefore, in the scenarios where one looks for interpretability for the model, one should think over the decision to apply any neural-network-based ML model.***

So to summarise, ML is no silver bullet, because it is often not 100% accurate, and we cannot correct the ML model case by case, and in certain cases we cannot even reason about the ML models.