Machine Learning Club, Summer Induction Assignment, 2023 - Q2

1 Predicting Bacterial Property

Considering this is a Tabular Dataset it is best to use a tree based models such as XG Boost, LightGBM, CatBoost etc. The selection of the model will depend on the dataset. It is better to use these models instead of the model due to the following reasons:

1. Tree-based models remain state-of-the-art on medium-sized data (10K samples) even without accounting for their superior speed.
2. Tree-based models are more robust to uninformative features than Neural Networks (NNs).
3. Tree-based models are better at learning irregular patterns in the data (non-smooth target functions) than deep learning models.

References

[1] L. Grinsztajn, E. Oyallon, and G. Varoquaux, ”Why do tree-based models still outperform deep learning on tabular data?,” *arXiv preprint arXiv:2207.08815*, 2022. [Online]. Available: `https://arxiv.org/abs/2207.08815`

Even if you do have access to an GPU there is a very high chance tree based models will perform better. However, where the datasets can be represented in some other format such as a graph (the bacterias are the nodes, and if they make contact there is an edge between them) possibly we can try out graph neural networks, but this totally depends on the dataset.

2 Predicting number of people on the beach

Considering this is also a tabular dataset, there is a very high chance that tree-based models will be hard to beat.

However, if we consider the problem as a fit for temporal GNNs, they might outperform tree-based models. In this approach, the nodes would represent the different places at the beach, and the other features such as temperature, precipitation, humidity, wind speed, etc., can be considered as node features. We can create a fully connected graph with the edge weights representing the distances between the places. We would create such a graph for each timestamp and apply temporal GNNs for prediction.

I suggested this model because a similar problem of traffic forecasting has a lot of literature on GNNs, and they have shown to outperform tree-based models.

References

[1] W. Jiang and J. Luo, ”Graph Neural Network for Traffic Forecasting: A Survey,” *arXiv preprint arXiv:2101.11174*, 2021. [Online]. Available: `https://arxiv.org/abs/2101.11174`

The above paper summarises the development of Temporal GNNs for traffic forecasting. Use only if you have a **GPU**

3 The Generative Model

For the task of creating an AI-based search engine using the MS-COCO 2014 dataset, which involves text-image search, there are multiple approaches that can be considered. One possible model that has shown promising results in this domain is the use of multimodal deep learning models, specifically combining convolutional neural networks (CNNs) for image processing and recurrent neural networks (RNNs) for text processing.

One paper that could be cited for this task is "Learning Deep Representations of Fine-Grained Visual Descriptions" by Reed et al. The paper proposes a model that uses a CNN to encode images and an RNN with LSTM cells to encode text descriptions. By training the model in a multimodal manner, they learn joint embeddings that can be used for text-image retrieval.

The model architecture for the text-image search engine could involve the following steps:

1. Image Processing: Use a pretrained CNN (such as ResNet or VGGNet) to extract image features. This involves passing the input image through the CNN layers and obtaining a high-level feature representation.

2. Text Processing: Utilize an RNN (e.g., LSTM) to process the input text. The text can be tokenized into words, and each word is passed through the RNN to obtain a sequence of text embeddings.

3. Multimodal Fusion: Combine the image features and text embeddings to obtain a joint representation of the image and text. This can be done through concatenation or other fusion techniques such as bilinear pooling or attention mechanisms.

4. Similarity Calculation: Compute the similarity between the query (image or text) and the database of images and text descriptions using the joint representation obtained in the previous step. This can be done using distance metrics like cosine similarity or Euclidean distance.

In terms of training speed, deep learning models can be computationally intensive, especially when working with large-scale datasets. Training times can vary depending on the model's complexity, hardware resources, and optimization techniques used. Techniques such as transfer learning, pretraining, and batch normalization can help speed up training.

4 Matrix Multiplication

Just do a simple matrix multiplication. The most efficient and with correct solutions with 0 RMSE. No need to use any models.