HIS PROJECT - TSA

Task 08

Author
Mehjabeen Jahangeer Khan
Hardikkumar Khunt

Contents

1 Chapter 15: Strategies for Global Deep Learning Forecasting Models

1.1 Chapter 10: Global Deep Learning Forecasting Models

Chapter 10 focuses on the significance of global deep learning models for forecasting. It underscores the following points:

- **Benefits of Global Models:** These models excel due to their ability to leverage larger datasets, enable cross-learning among different time series, facilitate multi-task learning, introduce regularization, and minimize engineering complexity.
- **Data Requirement:** Deep learning thrives on substantial data, making global models an excellent option for comprehensive analysis.
- **Shifting from Local to Global Models:** Adapting data loaders to sample from multiple time series is key to transitioning to global models.
- **Simplifying Forecasting:** Tools like the PyTorch Forecasting library streamline the process with features such as a high-level API and the `TimeSeriesDataset` class for efficient data preparation.

The chapter also discusses the implementation of a simplified forecasting model. In this setup:

- A single decoder operates at each timestep and target sequence.
- The model is trained only on historical data, focusing solely on predicting future values based on past trends.
- It avoids complexities like multi-sequence interactions or external variables, making forecasting straightforward and efficient.

In essence, Chapter 10 emphasizes the advantages of global models for deep learning forecasting and introduces tools that ease their implementation.

1.2 Using time-varying information

In this context,GroupNormalizer in TimeSeriesDataset is used to scale time series data, aiming for zero mean and unit variance. This stops the model from changing its parameters according to the consumption scales of different households. However, because distinct patterns from both bigger and smaller consumption families are mixed, information is lost. The mean and standard deviation values for every household, gathered during the original scaling phase, can be reintroduced as static-real features to the model in order to reintroduce this scale information without incorporating unscaled targets. As the model learns time series patterns, this gives it access to scale information.

1.3 Using static/meta information

The model may identify distinct patterns linked to various households by including household-specific factors like dynamic pricing and the Acorn group. In machine learning models, categorical features—which are not numerical—can provide difficulties. Nevertheless, there is a unique method for managing these categorical characteristics in deep learning models known as embedding vectors. In order to make categorical variables appropriate for neural networks, this approach transforms them into continuous vector representations. This improves the model's predicting ability by enabling it to discover intricate linkages and patterns in categorical data.

- **One-hot encoding:** creates a high-dimensional binary representation from categorical data, making each category a dimension. By adding dimensions equal to the number of categories, this technique produces a sparse representation. However, it ignores any potential similarities and sees all groups as equally remote. The fact that Saturday and Sunday are closer as weekend days, for example, is not captured.

- **Embedding vectors:** provide a dense representation of categorical characteristics by mapping each category to a numerical vector using an embedding layer. The dimensions of these vectors are smaller than the cardinality of the categorical feature. The model chooses the best vector for each category during training. In natural language processing, this method is frequently employed to embed hundreds of words into dimensions as tiny as 200 or 300..

1.4 Using the scale of the time series

TimeSeriesDataset's GroupNormalizer is used to scale time series data, aiming for zero mean and unit variance. As a result, the model is unable to modify its parameters in response to variations in household consumption levels. However, because distinct patterns from both larger and smaller consumption households are mixed, information is lost. The mean and standard deviation values for every household, gathered during the original scaling phase, can be added as static-real features to the model in order to reintroduce this scale information without incorporating unscaled targets. As the model learns time series patterns, this gives it access to scale information.

1.5 Balancing the sampling procedure

When choosing batches from time series data, the idea of balancing the sample process is explained using the bowl and paper chits analogy. Consider a bowl full of balls, each of which stands for a home time series, and the paper chits for various windows of samples that can be taken from each time series.

All of the chits are thrown into the bowl in the standard batch sampling process, and a batch is chosen at random by selecting a predetermined number of

chits. A skewed batch composition may arise from this, particularly if the time series's characteristics—such as varying lengths or amounts of consumption—are unbalanced.

A threshold-based sampling strategy is presented to counteract this bias. Rather than choosing chits in a uniform manner, each chit is assigned a random integer between 0 and 1 (p), and the chit is only chosen if p is less than a predetermined threshold (for example, p ¡ 0.5). The acceptance rate of chits can be controlled by varying this threshold, which can make it more or less difficult for a chit to be included in the batch.

The objective is to determine the appropriate thresholds for every chit in order to produce a batch that more evenly represents the various attributes (represented by colors in the analogy). In order to do this, the thresholds are established by taking the inverse of the number of chits in each bin or category. Chits from bins with more samples are less likely to be included, whereas bins with fewer samples are given greater weights, increasing the likelihood that their chits will be chosen. This method aids in producing batches with a more uniform distribution of various attributes.

1.6 Summary

In recent chapters, we've laid the groundwork for deep learning models and explored the concept of global models in deep learning. We've utilized PyTorch Forecasting and the versatile TimeSeriesDataset for model development.

Starting with a basic LSTM in the global context, we expanded our models by incorporating time-varying, static, and individual time series scale information, enhancing their performance. We also delved into an alternating sampling approach for mini-batches to ensure balanced representation in each batch.

While this chapter provides valuable techniques for improving forecasting models, it serves as a foundation for further model development. In the upcoming chapter, we'll explore specialized deep learning architectures tailored for time series forecasting.