Time Series forecasting with Temporal Fusion Transformers

Damir Khabibulin
Higher School of Economics

1. Introduction

In this project I research interpretable model for multi-horizon time series forecasting based on temporal fusion transformers that was proposed in the article [1]. The model will be reproduced on PyTorch framework, then I will extend training data for model and will experiment with hyperparameters and model.

2. Motivation

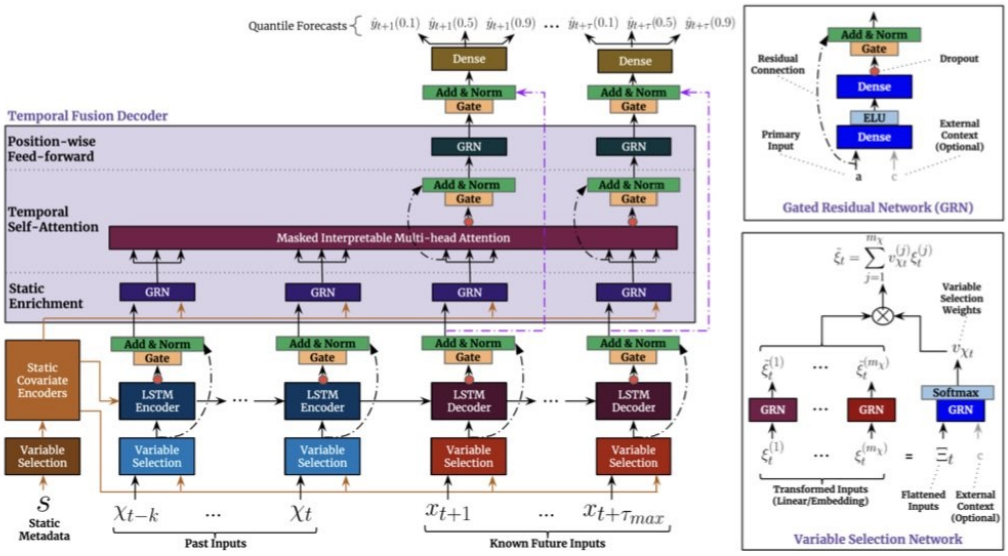
Multi-horizon forecasting is the prediction for estimates across several steps in future. It has many real-world applications. Typically, models provide forecasts from "black-box" models, while Temporal Fusion Transformer combines high-performance multi-horizon forecasting with interpretable insights into temporal dynamics.

3. Temporal Fusion Transformers

If we take I as unique entitles in a given time series dataset, in this project it is a different stock assets. Each entity i is associated with a set of static covariates $\boldsymbol{s}_i \in \mathbb{R}^{m_s}$, as well as inputs $\boldsymbol{X}_{i,t} \in \mathbb{R}^{m_x}$ and scalar targets $\boldsymbol{y}_{i,t} \in \mathbb{R}$ as each time-step $t \in [0, T_i]$. Time-dependent input features are subdivided into two categories $\boldsymbol{X}_{i,t} \in [\boldsymbol{z}_{i,t}^T, \boldsymbol{x}_{i,t}^T]^T$ - observed inputs $\boldsymbol{z}_{i,t} \in \mathbb{R}^{(m_z)}$ which can only be measured at each step and are unknown beforehand, and known inputs $\boldsymbol{x}_{i,t} \in \mathbb{R}^{m_x}$ which can be predetermined (e.g. the day-of-week at time t). In many scenarios, the provision for prediction intervals can be useful for optimizing decisions and risk management by yielding an indication of likely best and worst-case values that the target can take. As such, we adopt quantile regression to our multi-horizon forecasting setting (e.g. outputting the 10th, 50th and 90th percentiles at each time step). Each quantile forecast takes the form:

$$\hat{y_i}(q, t, \tau) = f_q(\tau, y_{i,t-k:t}, \boldsymbol{z}_{i,t-k:t+\tau}, \boldsymbol{s_i}), \tag{1}$$

where $\hat{y}_{i,t+\tau}(q, t, \tau)$ is the predicted q^{th} sample quantile of the τ-step-ahead forecast at time t, and $f_q(.)$ is a prediction model. In line with other direct methods, we simultaneously output forecasts for τ_max time steps - $\tau \in \{1, ..., \tau_max\}$. We incorporate all past information within a finite look-back window k, using target and known inputs only up till and including the forecast start time t (i.e. $y_{i,t-k:t} = \{y_{i,t-k}, ..., y_{i,t}\}$) and known inputs across the entire range (i.e. $\boldsymbol{x}_{i,t-k:t} = \{\boldsymbol{x}_{i,t-k}, ..., \boldsymbol{x}_{i,t}, ..., \boldsymbol{x}_{i,t+\tau}\}$)



TFT architecture. TFT inputs static metadata, time-varying past inputs and time-varying a priori known future inputs. Variable Selection is used for judicious selection of the most salient features based on the input. Gated Residual Network blocks enable efficient information flow with skip connections and gating layers. Time-dependent processing is based on LSTMs for local processing, and multi-head attention for integrating information from any time step.

Figure 1: Model Architecture

4. Model Architecture

1. Gating mechanisms to skip over any unused components of the architecture, providing adaptive depth and network complexity to accommodate a wide range of datasets and scenarios.

2. Variable selection networks to select relevant input variables at each time step.

3. Static covariate encoders to integrate static features into the network, through encoding of context vectors to condition temporal dynamics.

4. Temporal processing to learn both long- and short-term temporal relationships from both observed and known time-varying inputs. A sequenceto-sequence layer is employed for local processing, whereas long-term dependencies are captured using a novel interpretable multi-head attention block.

5. Prediction intervals via quantile forecasts to determine the range of likely target values at each prediction horizon

5. Experiments

5.1. Volatility prediction

In this experiment I used proposed dataset [2] that contains daily non-parametric measures of how volatility financial assets or indexes were in the past. Each day's volatility measure depends solely on financial data from that day. I have trained model on this dataset and checked perfomance.

5.2. Change Softmax to SM-Taylor softmax

Softmax function is a popular choice in deep learning classification tasks. Recently, this function has found application in other operations as well, such as the attention mechanisms. In this project I replaced softmax with SM-Taylor softmax [3]. Then trained model and checked perfomance.

5.3. Test on simple dataset

I have generated three simple datasets and checked perfomance.

6. Results

On a simple datasets that contain only known inputs and complex datasets which encompass the full range of possible inputs – we show that TFT achieves state-of-the-art forecasting performance.
Changing model in both cases came up to overfitting.
The code used for this project presents in this repository:
https://github.com/damu4/tft-forecast
Examples of predictions:

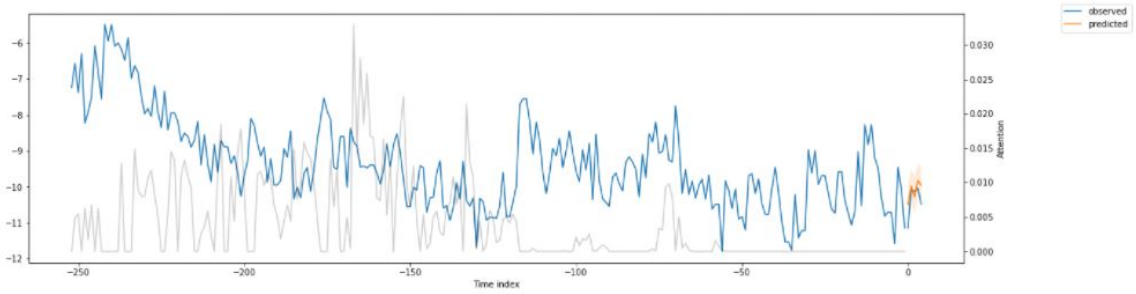


Figure 2: Volatility predictions for asset "SPX"

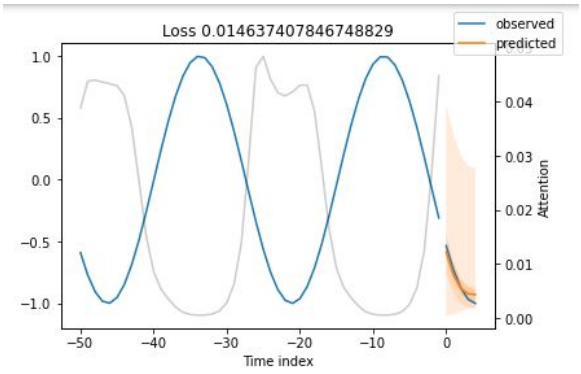


Figure 3: sin(x/2)

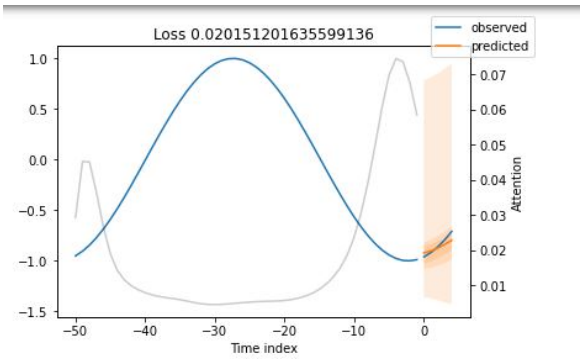


Figure 4: sin(x/8)

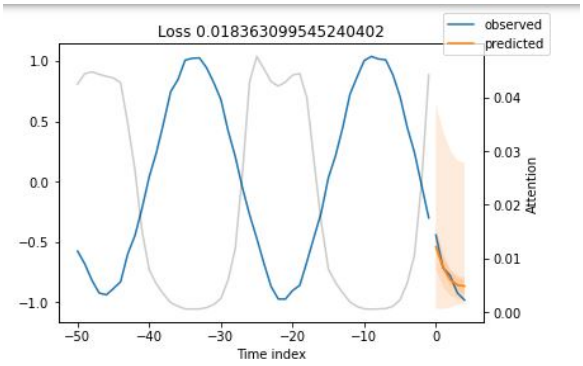


Figure 5: sin(x/4) with noise

7. References

[1] Bryan Lim, Sercan O. Arik, Nicolas Loeff, Tomas Pfister (2019). Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting.
[2] G. Heber, A. Lunde, N. Shephard, K. K. Sheppard, Oxford-man institute's realized library (2009). https://realized.oxford-man.ox.ac.uk
[3] Kunal Banerjee, Vishak Prasad C, Rishi Raj Gupta, Karthik Vyas, Anushree H, Biswajit Mishra. Exploring Alternatives to Softmax Function