Thanks for your careful and valuable comments. We will explain your concerns point by point.

We first address a common concern about the reproducibility. We will publish full source codes with the camera-ready manuscript, includes all of our specifications, training settings, and hyper-parameters.

[Reviewer1] • **Input and output of Transformer** The values and covariates of time series are normalized by z-score normalization as stated in the first paragraph of supplementary. Particularly, means and variances of values are computed with regard to values in conditioning range. Meanwhile, means and variances of covariates are computed with regard to the whole window. The input of Transformer is the concatenation of $(X_{1:(t_0+\tau)}, Y_{1:t_0})$ explained in Line. (107)-(111) and Algorithm 1. The output of Transformer is the value of the predicted time series in the prediction range directly. • L_G **and** L_D **in Figure 2** Sorry for the unclear expression of formulas, $L_G = L_\rho$ of Eq. (6)(7), $L_D = L_{adv}$ of Eq. (5). We will clarify it in the next version. • **Model prediction** We train different models for different quantile scores. • **Distribution of generator output** The output of Transformer is the predicted time series value which is used to compute quantile loss directly. • L_ρ **in Algorithm 1** The L_ρ in Algorithm 1 is a mistake, the correct representation is $L_\rho(\hat{Y}_{t_0+1:t_0+\tau}, Y_{t_0+1:t_0+\tau})$ and $\hat{Y}_{t_0+1:t_0+\tau}$ relying on Θ_G. There are some bits of mismatch between Algorithm 1 and Eq. (4)-(7), we will amend them in the next version. • **Discriminator loss** We will note the minibatch in the next version. • **Baseline N-BEATS** Considering that the hyper-parameters of N-BEATS may influence results in some terms, we compare our AST method with N-BEATS on Electricity and Traffic datasets with three splitting strategies (refers to Appendix C.4 of [5]). In short, AST achieves 0.050 (+34%), 0.065 (+3%), 0.069 (+157%) for Electricity dataset and 0.103 (+11%), 0.175 (+31%), 0.078 (+42%) on Traffic dataset with ND(Q_{50}) metrics. The notations in red are the performance gains relative to the best in N-BEATS. We will add these comparison results in the next version. • **Line. (36)** Our claim is invalid if the length of the predicted sequence equals to 1. However, such length is typically lager than 1 in most time series forecasting tasks (such as all tasks in [Oreshkin, ICLR 2020], AST and so on), which triggers the circumstance elaborated in Line. (36). • **Line. (74)-(75)** This work has been proved not scalable on large time series datasets since it has only been evaluated on small datasets[Seeger, Neurips2016]. • **Figure 1** Although the error in one time series is small, when it comes to hundreds of time series in some real-world conditions (such as data from real industrial manufacture), the accumulated error will seriously damage the system. Furthermore, more accurate peak forecasting improves not only decision making in capital expenditures, but also the reliability of the system. For example, peak load forecasting is a very important task for decision making processes in the electricity supply system of a country, as the consequences of over- or underestimation may increase the operating cost and hurt the stabilization of the whole system[Laouafi A, 2016]. • **Line. (120)** Refers to [Vaswani , Neurips 2017], attention-based Transformer can learn the attention due to the overall sequence, which is regarded as long-term dependencies. • **Line. (123)** The footnote '2' next to 'h' seems to easily be misunderstood, we will amend this in the next version. The 'n' in Line. (123) represents sequence length, which is a new 'n' compared 'n' in Line. (113), we will add this explanation and change the 'n' in Line. (113) in the next version. • **Line. (166)-(175)** The input time series to discriminator network has a fixed length of conditioning length + forecast horizon.

[Reviewer2] • **Difference with TimeGAN and LogSparse Transformer** TimeGAN[Yoon J, Neurips 2019] trains the time series forecasting model on synthetic time series and tests the model on real time series, whose forecasting performance will be poor when the quality of the generated data is not good enough. However, AST trains the forecasting model by real time series data and GAN in AST can help to regularize the model from the sequence level. Sparse Transformer[Li S, Neurips 2019], as you mentioned, can learn sparse attention by the fixed logL interval, ignoring the real attention relationship. While our sparse attention is learned by α-entmax, which is more flexible and appropriate for time series. • **Generator of AST** AST uses a standard transformer but changes the softmax to α-entmax. • **DSSM** DSSM is the abbreviation of DeepSate, sorry for the unclear expression. • Q_{50} **in Table 4** Thanks for indicating the mistake, the correct Q_{50} loss of AST on elect_1d is 0.042 in Table 4, and all the results in Table 2 are correct. 0.039 is one of the result of five experiments. we will correct it in the next version.

[Reviewer3] • **More discussion and baseline** To fulfill R3 requests, we compare AST with STDL [Vincent, Neurips 2019]. The results on Traffic dataset are obtained by running the original authors implementation code. In short, The Q_{50} and Q_{90} on Traffic_1d & Traffic_7d in STDL achieves 0.151 & 0.168 and 0.073 & 0.102 respectively, which are worse than AST. The dual-attention model in [Qin, IJCAI 2018] can only predict one step ahead, which is not suitable for mid-term and long-term time series forecasting. For LSTNet[Lai, SIGIR 2018], the model is based on RNN architecture and Auto-Regressive to catch long- and short-term dependencies. However, RNN-based models have been proved to be inefficient in dealing with long-term dependencies[Vaswani A, Neurips 2017]. Compared to LSTNet, AST has better capibilities of capturing long-term dependencies.

[Reviewer4] • **Supplementary** We will add a legend and more clear description to Figure 2. The prediction of time series using statics and other neural network approaches on several datasets will be added to the supplementary in the next version, as well as the visualization of average attention weights.