A Supplementary Materials

A.1 Dataset Description

We describe the additional details of each dataset in the followings.

For `electricity`, we take 500k training windows between 2014-01-01 to 2014-09-01 by reference [14, 24]. And we use the first 90% for the training set and the last 10% as the validation set. Testing set is the next 7 days after the training set. We apply the z-score normalization to the real-valued inputs of each time series. For the input covariates, we consider the identifier of time series, day-of-week, hour-of-day and age which is the number of time-steps from the first observation of this time series.

For `traffic`, also to keep consistent with previous work, we take 500k training windows between 2018-01-02 and 2018-06-15. And we use the same split method, normalization and time-related covariates as `electricity`.

For `wind, M4-Hourly` and `solar`, we choose 10k, 50k, 125k training windows respectively refer to[14, 24]. And the train/validation split method, choose of test set and normalization are the same as `electricity` and `traffic`. And for the `wind` and `M4-Hourly`, we use the same time-related covariates as `electricity`. For `wind`, we use day-of-week, week-of-year, month-of-year and the age as time-related input covariates. And we leave the last 210 days as the test set.

A.2 Experiment Details

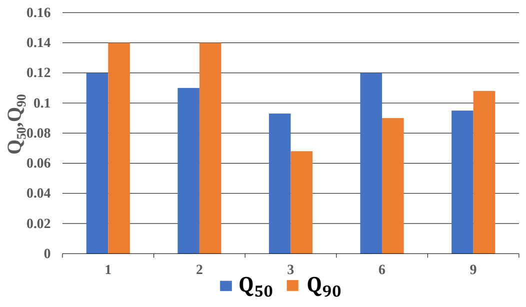
We implement our model by PyTorch on a single RTX 2080Ti GPU. For the Sparse Transformer, H=8 , $d_k = d_v = 20$. For other hyperparameters (e.g. learning rate and the number of layers), a grid-search is used to find best values. The training set is used to fit our models and the validation set is used to choose the optimal parameters. And then the evaluation metrics(Q_{50} and Q_{90}) can be computed on the test set. We train the Sparse Transformer by Adam optimizer[10] with default parameters, lr=6.25e-5, and a warm-up scheduler[26]. While the discriminator is trained by RMSprop[16] with a learning rate of 0.0001.

We further evaluate the sensitivities of different hyper-parameters. We mainly consider d_{model}, which is the dimension of Sparse Transformer, and N, the number of layers of encoder and decoder. We explore different layers $N \in \{1, 2, 3, 6, 9\}$ and $d_{model} \in \{40, 80, 160, 320, 512\}$ with all the other parameters fixed, in which h=8. We predict for one day ahead on `traffic`. The results are shown in the Figure 1. We find that when the number of layers N is 3 and model dimension d_{model} is 160, our model achieves the best performance. In addition, our model is not very sensitive to hyper-parameters from the results.

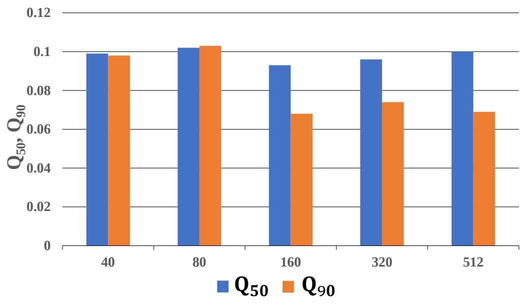
A.3 Attention Weight Density Analysis

Here we show an example of learned attention weight density as a complementary of the Ablation Study in Section 5.3. The distribution of attention densities of different α is shown in Figure 2.

The results show that compared to softmax(α=1), entmax(α=1.5) have higher density on the range 0.9-1.0 and less density on the range 0-0.1, which have revealed that entmax can allocate more attention weight to correlated steps and ignore the uncorrelated steps of the prediction range.



(a) Performance of different N



(b) Performance of different d_{model}

Figure 1: Illustration of performance with different parameter settings

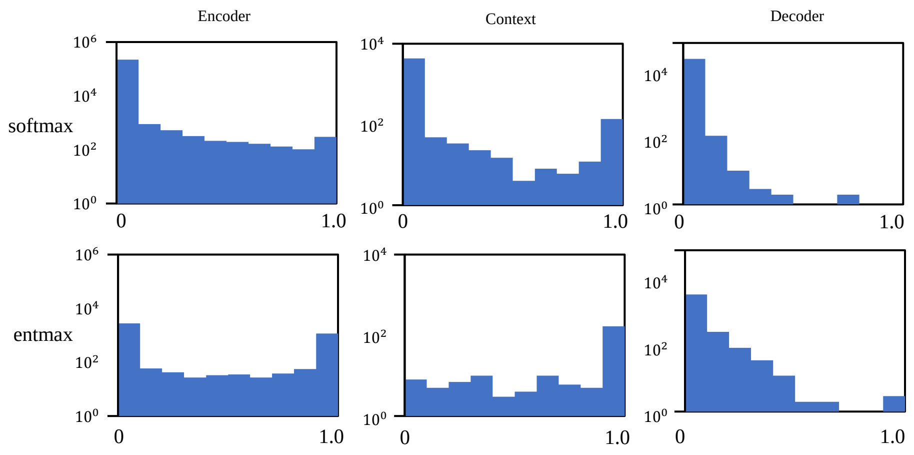


Figure 2: Distribution of attention densities (average number of samples receiving non-zero attention weight) for all attention heads and all validation sentences trained on `traffic` dataset. Compared to softmax, entmax have higher density on the range 0.9-1.0 and less density on the range 0-0.1.