Clustering is crucial for load forecasting which is essential for moral power operation and management. It’s hard to build a general model that can accurately predict load profile in all conditions. While build a good model for load profiles shares similar features are much easier.

For our Pecan street data, we want to cluster the residents with similar load profiles into k-partitions.

Distance is one of the most straightforward way to measure similarity. Thus we choose one of the most widely used model, k-means cluster, whose objective function is distortion – the average distances of each points to its clustering centroid.

While k-means might be unstable for high-dimensional clustering. It is proved that all pairwise distances in space concentrated to a same value as the dimension goes to infinity. When all distances are similar, noises become extremely matter, as any slight variation in data while changing the relationships among distances. Thus all clustering methods based on distance, including k-means cluster that based on Euclidean distance, generally shown unstable cluster results. Load profile, as a time series, obviously, are high dimensional.

To generate a stable cluster result for load forecasting. We need a dimension reduction method to capture the essential features internal the high dimensional vectors and eliminate noise in data.

Here we introduce auto-encoder, a dimension reduction method shown superior performance in the image classification field.

The general idea is like this. An auto-encoder is a neural network with a “bottleneck”. And the neural networks are trained to learn the input itself. During this process, the “bottleneck” layer captures the most essential features in the original data. From these features, we can reconstruct the original vector with the highest quality, i.e. these features contain the most information in the input. Usually, we call the "bottleneck" vector latent vector, because it contains hidden features in the input.

An example can be seen in our ppt. The left input is a digit from the MNIST dataset. The right is the reconstructed figure from the latent vector. The right one pretty good, it is exactly a 4. So you can see the power of the auto-encoder, even a 2-d latent vector captures most information contains in this figure.

We extend this idea to time series. A time series can be considered as a 1-d image. Mimicking the idea of image classification, we build a 1-d convolutional auto-encoder to extract hidden features.

Here is the architecture of our neural network, it contains

To see if the k-means cluster with auto-encoder is more robust to noise. We test this idea on a simple problem. Use k-means cluster on a Gaussian contaminated, unbalanced sine dataset. This dataset contains a hundred noisy sine series and one negative sine series. As we increase the noise level on the negative sine series. K-means cluster's performance deteriorates soon. It fails to correctly cluster the data in the high dimension. While our 1-d convolutional auto-encoder can afford higher-level noise.

We then applied our auto-encoder to our Pecan Street data. We first calculate the average daily profile for each resident. Then we use our auto-encoder to extract crucial features from the daily load profile.

Here is a record of the training process. The red line denotes the reconstructed load profile. During the training process, our latent vector captures more and more information in the input load profile.

As find the best cluster is an NP-hard problem. K-means algorithm only guaranteed the result is a local minimum. We expect the cluster on the latent vector can result in lower distortions, i.e. a denser cluster result. This means it is easier for the K-means algorithm to find the global minimum in the latent space.

To see if we are correct, we then implemented k-means clustering based on learned latent vector and original data to compare the performance. Surprisingly, the two methods showed very similar results. Here are the clusters given by k-means clusters on original data and latent data respectively. You can see that the clustering result is almost the same. We also compared distortions for different k. The results are also similar. Notice that the distortion line for clustering on latent space even higher than the original one. The loss of information during the encoding even harms the clustering.

This is just upsetting, but the more important thing is, why our method fails in the Pecan Street dataset. Because the phenomenon shown in Pecan street data is exactly the same as the theory predicted. High-dimensionality, strong noise. Auto-encoder should have better performance.

What goes wrong?

We re-check the related theorems. And we noticed that the curse of dimensionality depends on really strong assumptions. It assumes, each element, each dimension, follows the independent and identical distribution. The Pecan Street data comes from the real world and depicts a load profile. Obviously elements are not i.i.d distributions. Thus the curse of dimensionality possibly not real holds. This also explains why our auto-encoder works well in sine experiments. Each entry in our sine experiments is independent and all from Gaussian distribution. The only difference is the mean on each dimension. The condition for the curse of dimensionality generally holds. Thus auto-encoder outperforms the naïve k-means cluster.

Latent doable

Latent can allieviate curse of dimensionality

Real world time series data contains dependents data thus may not suffer from curse of dimensionality.