Mixture Density Network for Joint Position Coordinates Prediction

Chen Xupeng

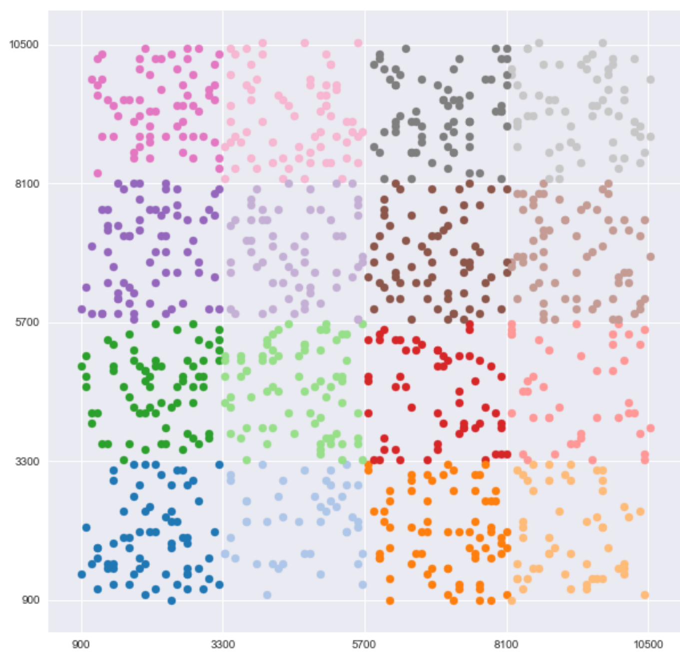
Zhang Zehua

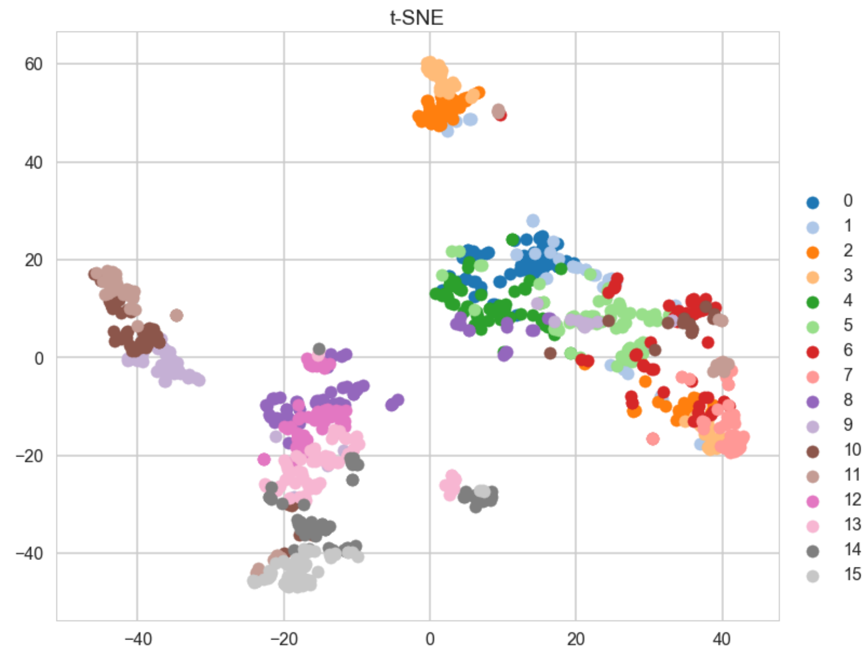
Outline

- Data
  - Visualization
  - Analysis: NaN, correlation
- Model
  - Machine Learning model
  - MLP based Mixture Density Network
  - Attention Model

Mountain data

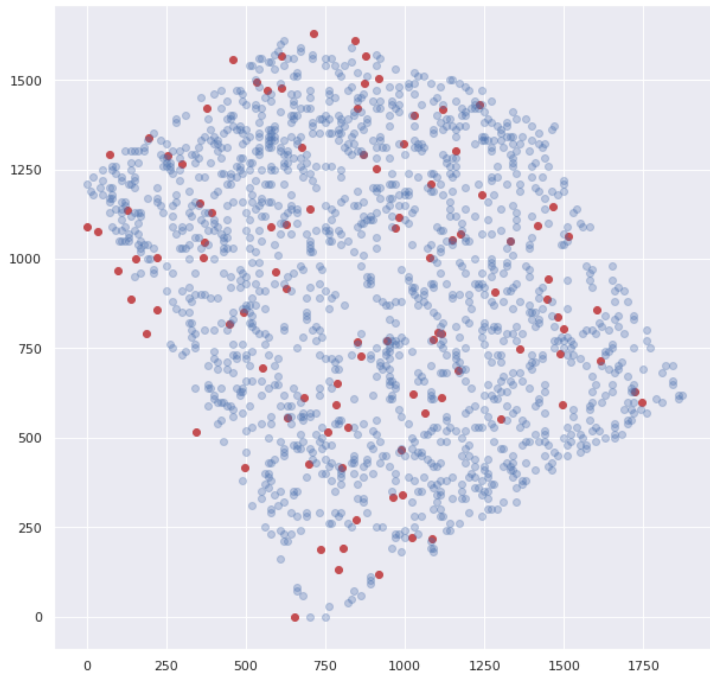
- 1000 receivers with 6 transmitters.

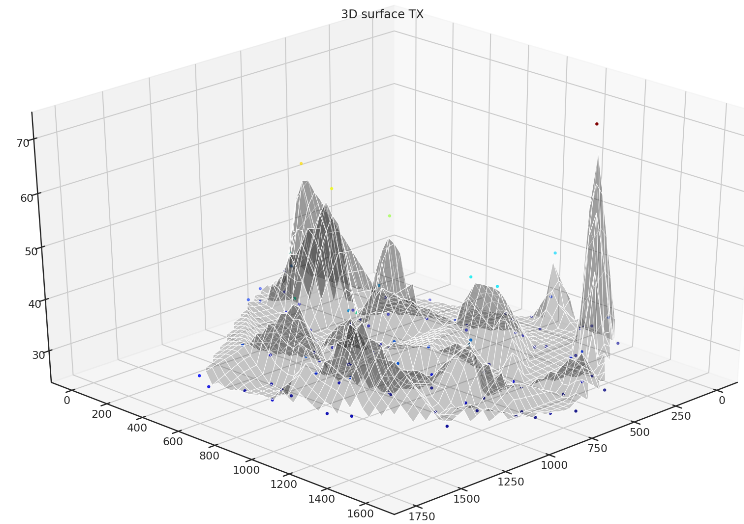


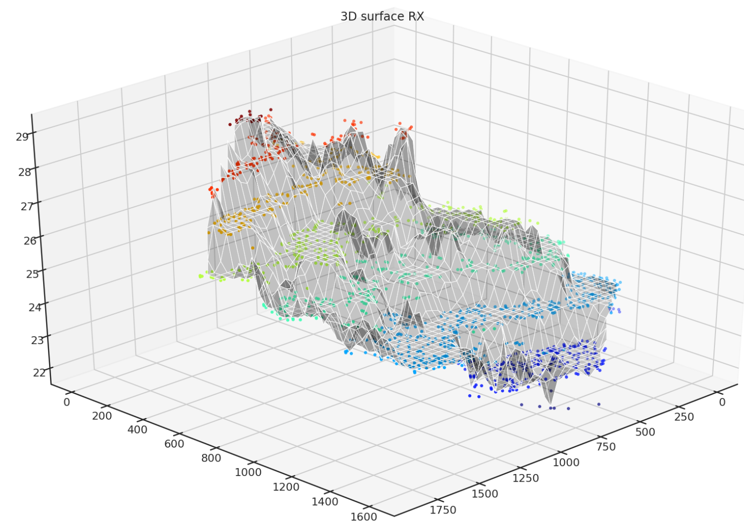


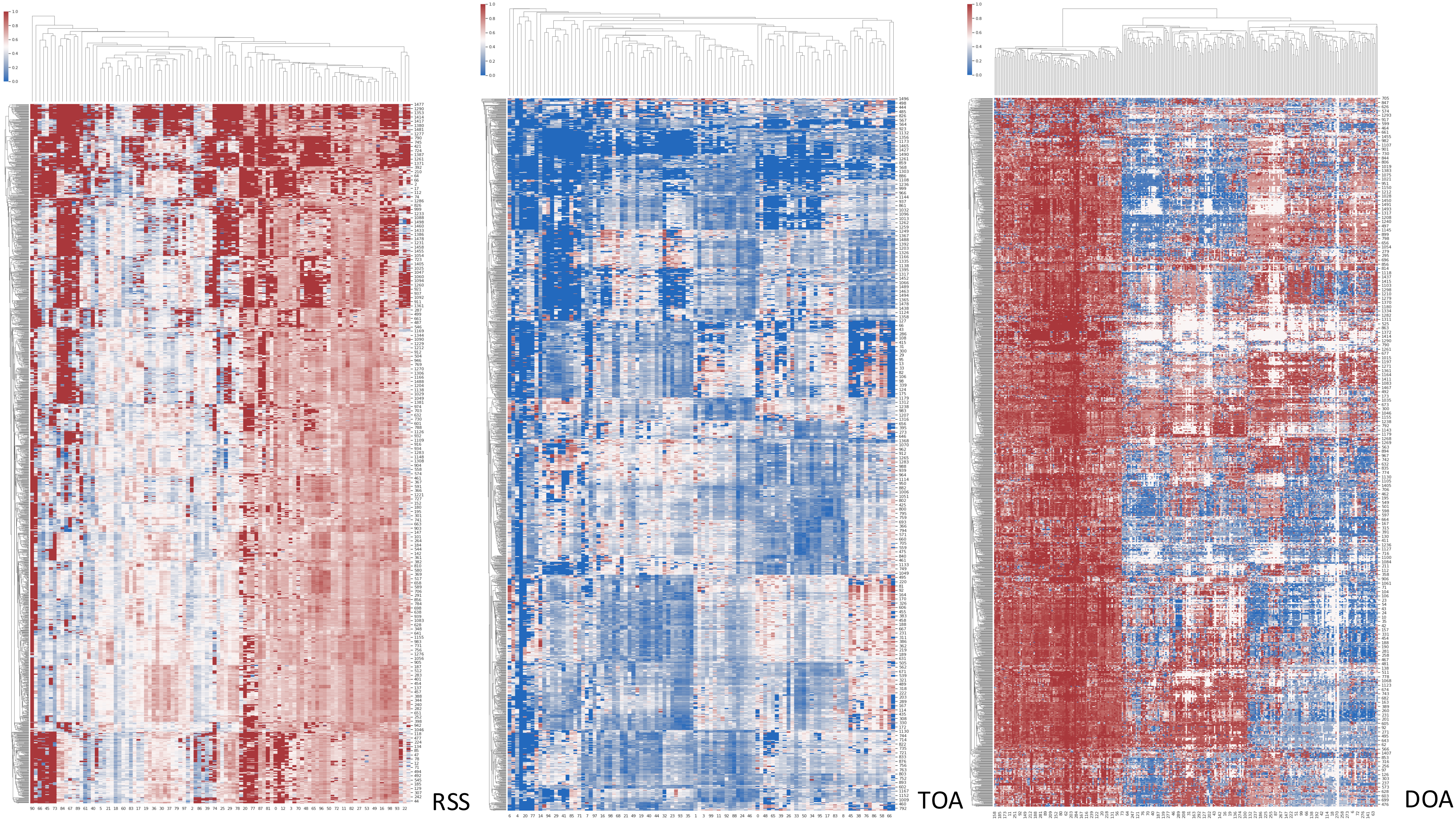
City data

There are 100 transmitters and 1500 receivers
RSS, TOA and DOA(3 angles) can be detected. Each receivers have 500 features.
Due to barriers, there are many missing values.








RSS
TOA
DOA

Mixture Density Network

- The original mixture density model combined several (k) one dimensional normal distribution together.
- Combine several 2D normal distribution together. So that we can jointly predict coordinates, and benefit from mixture model.
- Use a multi layer perceptron to generate and train parameters.
- Use EM algorithm to optimize the model

$$L(\mathbf{w}) = \frac{-1}{N} \sum_{n=1}^{N} \log \Big(\sum_{k} \pi_k(\mathbf{x_n}, \mathbf{w}) N(\mathbf{y_n} | \mu_k(\mathbf{x_n}, \mathbf{w}), I\sigma_k^2(\mathbf{x_n}, \mathbf{w})) \Big).$$

Tricks to avoid gradient vanishing: use log and cutting method to restrict parameters

Mode Finding

- We should find mode (the maximum) in mixture distribution by following methods:

1 Mode finding for Isotropic Gaussian mixture model using mean-shift algorithm

The probability density function of a isotropic Gaussian mixture is:

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{D}{2}}} \sum_{k=1}^{K} \frac{\pi_k}{\prod_{d=1}^{D} \sigma_{kd}} \exp\left(- \sum_{d=1}^{D} \frac{(x_d - \mu_{kd})^2}{2\sigma_{kd}^2} \right)$$

- D: number of dimensions for each Gaussian components
- K: number of components

The gradient of $p(\mathbf{x})$ w.r.t x_d is:

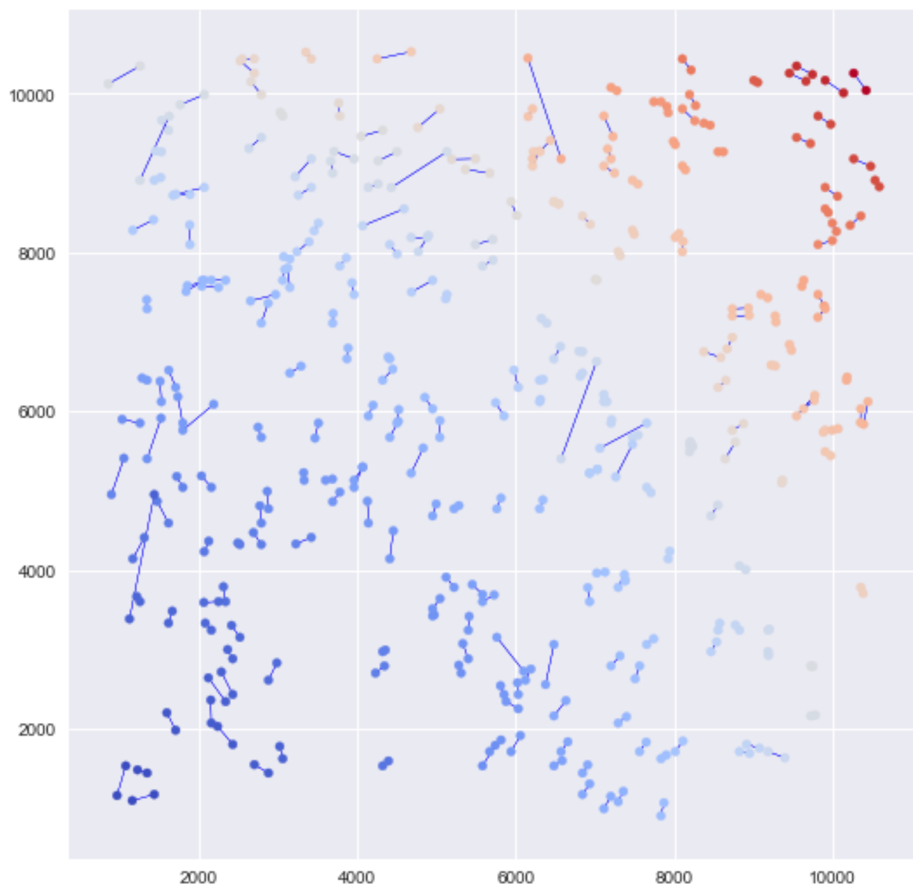
$$\frac{\partial p(\mathbf{x})}{\partial x_d} = \frac{1}{(2\pi)^{\frac{D}{2}}} \sum_{k=1}^{K} \frac{\pi_k}{\prod_{j=1}^{D} \sigma_{kj}} \exp\left(- \sum_{j=1}^{D} \frac{(x_d - \mu_{kj})^2}{2\sigma_{kj}^2} \right) \frac{1}{\sigma_{kd}^2}(x_d - \mu_{kd})$$

The modes of a Gaussian mixture model is found by setting $\frac{\partial p(\mathbf{x})}{\partial x_d} = 0$.

The update equation for x_d is:

$$x_d^{(t+1)} = \frac{\sum_{k=1}^{K} \frac{\pi_k}{\sigma_{kd}^2 \prod_{j=1}^{D} \sigma_{kj}} \exp\left(- \sum_{j=1}^{D} \frac{(x_d^{(t)} - \mu_{kj})^2}{2\sigma_{kj}^2} \right) \mu_{kd}}{\sum_{k=1}^{K} \frac{\pi_k}{\sigma_{kd}^2 \prod_{j=1}^{D} \sigma_{kj}} \exp\left(- \sum_{j=1}^{D} \frac{(x_d^{(t)} - \mu_{kj})^2}{2\sigma_{kj}^2} \right)}$$

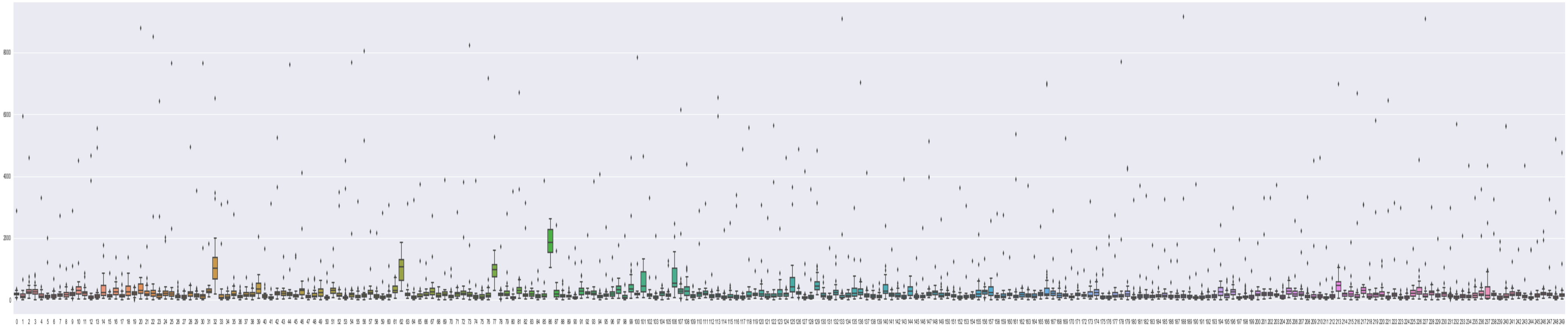
Results for mountain data



Most samples are predicted well, some have a bigger shift

Hard to predict samples

- Test different hyper-parameters combination and calculate RMSE in many rounds.
- Each box is one sample being tested many rounds.



- Some samples are always harder to predict in many rounds.

Data imputation

- For missing values in city data, we should do data imputation to fill in.
- It’s hard to fill in data before we learn the relation between position and feature values.
- Further analyze data and find the most reliable features.

Attention Method

- Borrowed from NLP. Assign different weights to different features.
- For city data, there are feature missing and feature redundant problem.
- To decide which feature is reliable to use, we use an attention model to assign weights to features.
- Use prior experience to restrict some weights.

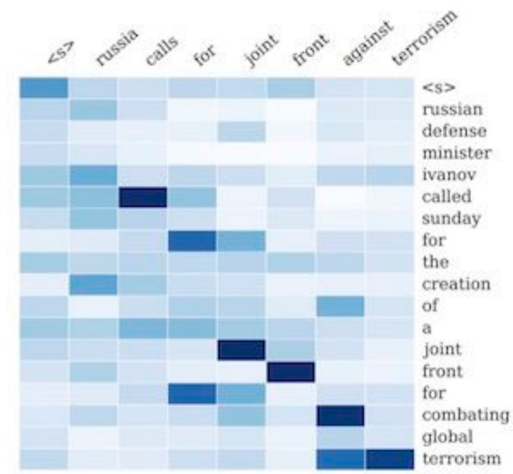


Figure 1: Example output of the attention-based summarization (ABS) system. The heatmap represents a soft alignment between the input (right) and the generated summary (top). The columns represent the distribution over the input after generating each word.