Mixture of Experts

Seungjin Choi

Department of Computer Science and Engineering
Pohang University of Science and Technology
77 Cheongam-ro, Nam-gu, Pohang 37673, Korea
seungjin@postech.ac.kr

Supervised Learning and Function Approximation

- The supervised learning involves estimating a function which relates inputs $\{\boldsymbol{x}_l\}$ to their associated targets $\{\boldsymbol{t}_l\}$, given a data set $\mathcal{D} = \{\boldsymbol{x}_l, \boldsymbol{t}_l\}$.
- We wish to learn either $p(\boldsymbol{t}|\boldsymbol{x})$ or $p(\boldsymbol{t}, \boldsymbol{x}) = p(\boldsymbol{t}|\boldsymbol{x})p(\boldsymbol{x})$ if we care to model the input distribution too.
- Once we have these probabilistic models, we can always choose a deterministic function through

$$f(\boldsymbol{x}) = \langle \boldsymbol{t}|\boldsymbol{x} \rangle ,$$

or if the density function is nicely unimodal, the average could be a point of low probability and in such a case we might choose the maximum

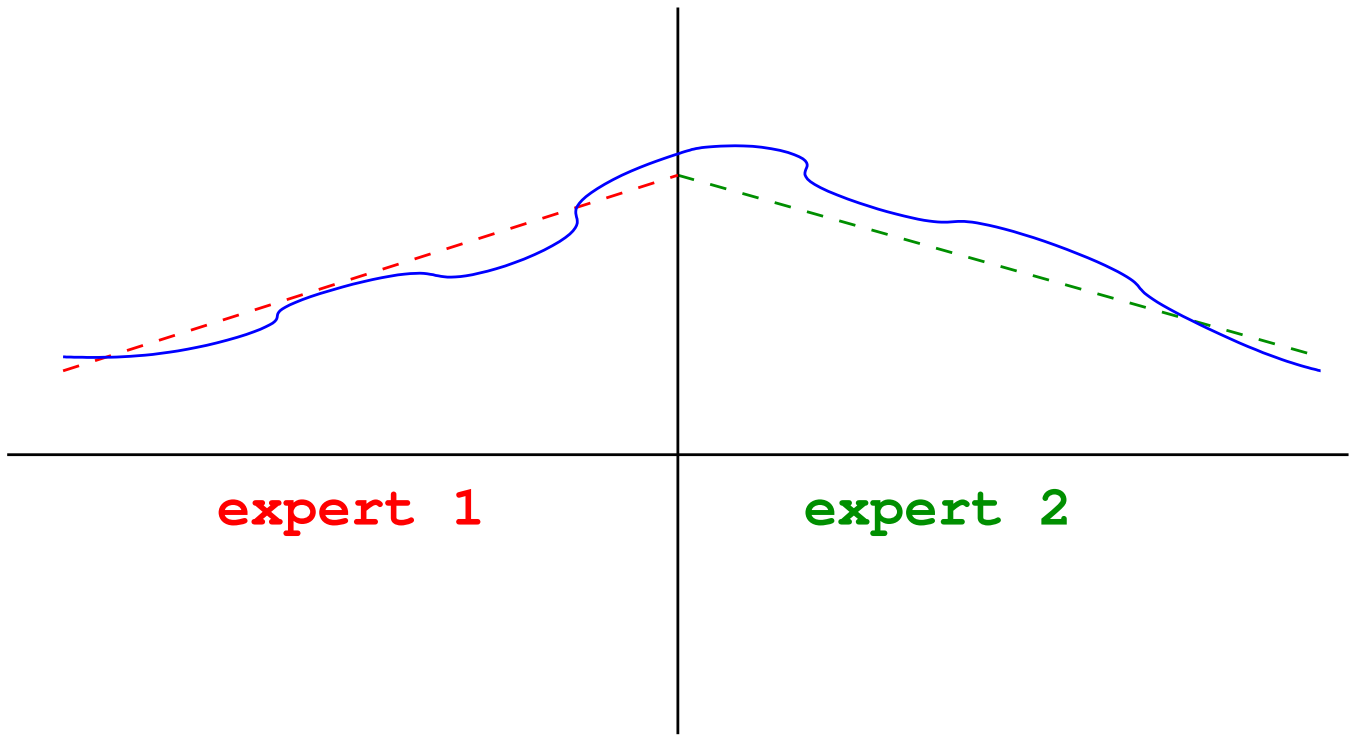
$$f(\boldsymbol{x}) = \arg\max_{\boldsymbol{t}} p(\boldsymbol{t}|\boldsymbol{x}).$$

Modeling Nonlinear Functions

- A powerful principle to model nonlinear functions is by approximating it through many local linear functions.
- The same strategy can be adopted for the density estimation problem.
- We employ many simple local models and combine them to approximate the density, very much in the same spirit as MoG.
- This leads to mixture of experts.

Mixture of Experts: A Model

- Model $p(\boldsymbol{t}|\boldsymbol{x})$ as $\boxed{p(\boldsymbol{t}|\boldsymbol{x}) = \sum_{i=1}^{M} p(\boldsymbol{t}, i|\boldsymbol{x}) = \sum_{i=1}^{M} p(\boldsymbol{t}|i, \boldsymbol{x}) p(i|\boldsymbol{x})}$.
- The basic idea is to make local models to specialize on a certain task.
  - Experts: $p(\boldsymbol{t}|i, \boldsymbol{x})$ (local models).
  - Gatekeepers: $p(i|\boldsymbol{x})$ (what region experts should model, is regulated by the gatekeepers)



Models for Experts and Gates

- For the gates, we use the softmax functions,

$$p(i|\boldsymbol{x}) = g_i = \frac{e^{\xi_i}}{\sum_{j=1}^{M} e^{\xi_j}},$$

where $\xi_i = \boldsymbol{v}_i^\top \boldsymbol{x} + \boldsymbol{v}_i^0 = \widetilde{\boldsymbol{v}}_i^\top \widetilde{\boldsymbol{x}}$.

- Modelling experts is as follows. First we linear combine the inputs

$$\boldsymbol{\mu}_i(\boldsymbol{x}) = \boldsymbol{W}_i \boldsymbol{x} + \boldsymbol{w}_i^0 = \widetilde{\boldsymbol{W}}_i \widetilde{\boldsymbol{x}},$$

which will be used as mean values for a Gaussian distribution modelling the target data, i.e.,

$$p(\boldsymbol{t}|\boldsymbol{x}, i) = \mathcal{N}(\boldsymbol{t} \mid \left[\boldsymbol{\mu}_i(\boldsymbol{x}), \sigma^2 \boldsymbol{I}\right],$$

where we treat the noise as a fixed parameter, but we could estimate it from the data.

Learn MoE

- How to learn MoE? $\Rightarrow$ EM!
- In E-step, we compute responsibilities and calculate the expected complete-data log-likelihood.
- In M-step, we calculate the partial derivatives of the expected complete-data log-likelihood with respect to associated parameters, in order to derive updating rules for those parameters.

EM-MoE: E-Step

We compute responsibilities:

$$\begin{aligned} R_{in} &= p(i|\boldsymbol{x}_n, \boldsymbol{t}_n) \\ &= \frac{p(\boldsymbol{t}_n|\boldsymbol{x}_n, i)\, p(i|\boldsymbol{x}_n)}{\sum_{j=1}^{M} p(\boldsymbol{t}_n|\boldsymbol{x}_n, j)\, p(j|\boldsymbol{x}_n)}. \end{aligned}$$

The complete-data log-likelihood is

$$\begin{aligned} \mathcal{L}_c &= \sum_{n=1}^{N}\sum_{i=1}^{M} \log p(\boldsymbol{x}_n, \boldsymbol{t}_n, i) \\ &= \sum_{n=1}^{N}\sum_{i=1}^{M} \log \left[p(\boldsymbol{t}_n|\boldsymbol{x}_n, i) p(i|\boldsymbol{x}_n)\right]. \end{aligned}$$

Hence, the expected complete-data log-likelihood is

$$\langle \mathcal{L}_c \rangle = \sum_{n=1}^{N}\sum_{i=1}^{M} R_{in} \left[\log p(\boldsymbol{t}_n|\boldsymbol{x}_n, i) + \log p(i|\boldsymbol{x}_n)\right].$$

EM-MoE: M-Step

- Solving $\frac{\partial \langle \mathcal{L}_c \rangle}{\partial \widetilde{\boldsymbol{W}}_i} = 0$ leads to $\widetilde{\boldsymbol{W}}_i = \left(\sum_{n=1}^{N} R_{in} \boldsymbol{t}_n \widetilde{\boldsymbol{x}}_n^\top \right) \left(\sum_{n=1}^{N} R_{in} \widetilde{\boldsymbol{x}}_n \widetilde{\boldsymbol{x}}_n^\top \right)^{-1}$.
- We have $\frac{\partial \langle \mathcal{L}_c \rangle}{\partial \widetilde{\boldsymbol{v}}_i} = \sum_{i=1}^{N} (R_{in} - g_{in}) \widetilde{\boldsymbol{x}}_n$, which is hard to optimize since the softmax function depends on the parameters $\{\widetilde{\boldsymbol{v}}_i\}$.
- We see that g_{in} is optimized to be as close as possible to the R_{in}, leading to

$$\log R_{in} \approx \widetilde{\boldsymbol{v}}_i^\top \widetilde{\boldsymbol{x}}_n - \underbrace{\log \left[\sum_j \exp \left\{ \widetilde{\boldsymbol{v}}_j^\top \widetilde{\boldsymbol{x}}_n \right\} \right]}_{\text{normalization constant}}.$$

- This is reduced to a simple regression problem. Minimizing $\sum_n \left(\log R_{in} - \widetilde{\boldsymbol{v}}_i^\top \widetilde{\boldsymbol{x}}_n \right)^2$, leads to

$$\widetilde{\boldsymbol{v}}_i = \left(\sum_{n=1}^{N} \log [R_{in}] \widetilde{\boldsymbol{x}}_n^\top \right) \left(\sum_{n=1}^{N} \widetilde{\boldsymbol{x}}_n \widetilde{\boldsymbol{x}}_n^\top \right)^{-1}.$$

Algorithm Outline: EM-MoE

- E-Step: Compute responsibilities

$$\begin{aligned} R_{in} &= p(i|\boldsymbol{x}_n, \boldsymbol{t}_n) \\ &= \frac{p(\boldsymbol{t}_n|\boldsymbol{x}_n, i)\, p(i|\boldsymbol{x}_n)}{\sum_{j=1}^{M} p(\boldsymbol{t}_n|\boldsymbol{x}_n, j)\, p(j|\boldsymbol{x}_n)}, \end{aligned}$$

  where $p(\boldsymbol{t}_n|\boldsymbol{x}_n, i) = \mathcal{G}_{t_n}\left[\boldsymbol{\mu}_i(\boldsymbol{x}_n), \sigma^2 \boldsymbol{I}\right]$ with $\boldsymbol{\mu}_i(\boldsymbol{x}_n) = \widetilde{\boldsymbol{W}}_i \widetilde{\boldsymbol{x}}_n$, and $p(i|\boldsymbol{x}_n) = g_{in} = \frac{e^{\xi_{in}}}{\sum_j e^{\xi_{jn}}}$ with $\xi_{in} = \widetilde{\boldsymbol{v}}_i^{\top} \widetilde{\boldsymbol{x}}_n$.

- M-Step: Update parameters

$$\begin{aligned} \widetilde{\boldsymbol{W}}_i &\leftarrow \left(\sum_{n=1}^{N} R_{in} \boldsymbol{t}_n \widetilde{\boldsymbol{x}}_n^{\top}\right) \left(\sum_{n=1}^{N} R_{in} \widetilde{\boldsymbol{x}}_n \widetilde{\boldsymbol{x}}_n^{\top}\right)^{-1}, \\ \widetilde{\boldsymbol{v}}_i &\leftarrow \left(\sum_{n=1}^{N} \log\left[R_{in}\right] \widetilde{\boldsymbol{x}}_n^{\top}\right) \left(\sum_{n=1}^{N} \widetilde{\boldsymbol{x}}_n \widetilde{\boldsymbol{x}}_n^{\top}\right)^{-1}. \end{aligned}$$