Deep Learning

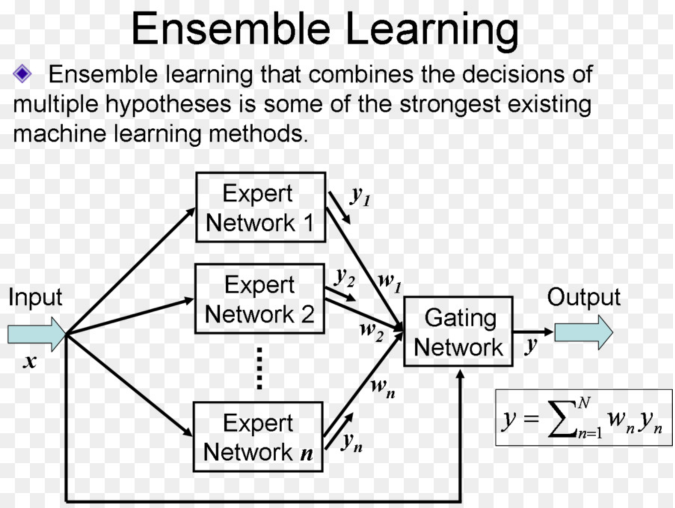
Vazgen Mikayelyan

YSU, Krisp

December 18, 2019

Outline

Ensemble of Neural Networks



Ensemble of Neural Networks

Note that

$$(y - \hat{y})^2 = \left(\sum_{i=1}^{n} w_i y_i - \hat{y} \right)^2$$

Ensemble of Neural Networks

Note that

$$(y - \hat{y})^2 = \left(\sum_{i=1}^{n} w_i y_i - \hat{y} \right)^2 = \left(\sum_{i=1}^{n} w_i \left(y_i - \hat{y} \right) \right)^2$$

Note that

$$(y-\hat{y})^2 = \left(\sum_{i=1}^{n} w_i y_i - \hat{y}\right)^2 = \left(\sum_{i=1}^{n} w_i \left(y_i - \hat{y}\right)\right)^2 \leq \sum_{i=1}^{n} w_i^2 \sum_{i=1}^{n} \left(y_i - \hat{y}\right)^2.$$

Note that

$$(y-\hat{y})^2 = \left(\sum_{i=1}^{n} w_i y_i - \hat{y}\right)^2 = \left(\sum_{i=1}^{n} w_i \left(y_i - \hat{y}\right)\right)^2 \leq \sum_{i=1}^{n} w_i^2 \sum_{i=1}^{n} (y_i - \hat{y})^2 .$$

Let $w_1 = \ldots = w_n = \frac{1}{n}$, then

$$(y-\hat{y})^2 \leq \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y})^2$$

Ensemble of Neural Networks

Note that

$$(y-\hat{y})^2 = \left(\sum_{i=1}^{n} w_i y_i - \hat{y}\right)^2 = \left(\sum_{i=1}^{n} w_i \left(y_i - \hat{y}\right)\right)^2 \le \sum_{i=1}^{n} w_i^2 \sum_{i=1}^{n} \left(y_i - \hat{y}\right)^2.$$

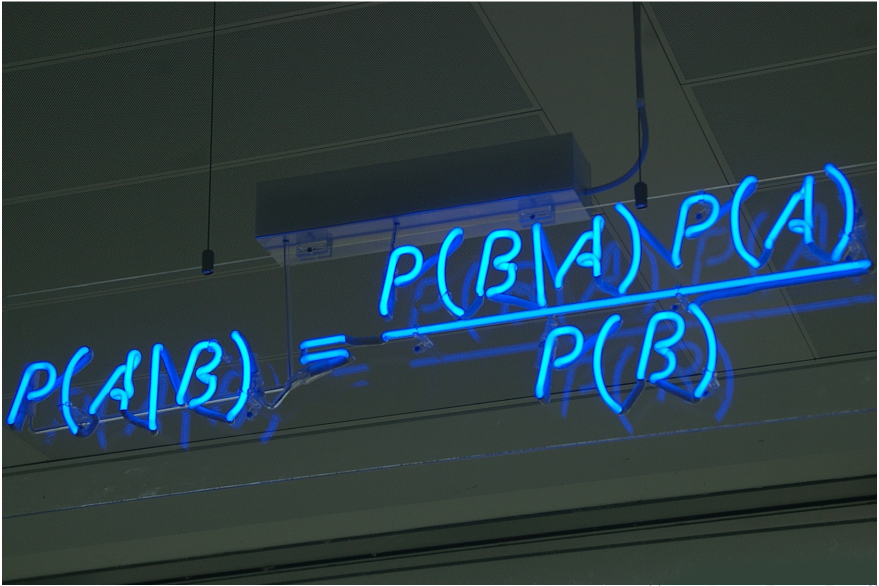
Let $w_1 = \ldots = w_n = \dfrac{1}{n}$, then

$$(y-\hat{y})^2 \le \frac{1}{n} \sum_{i=1}^{n} \left(y_i - \hat{y}\right)^2$$

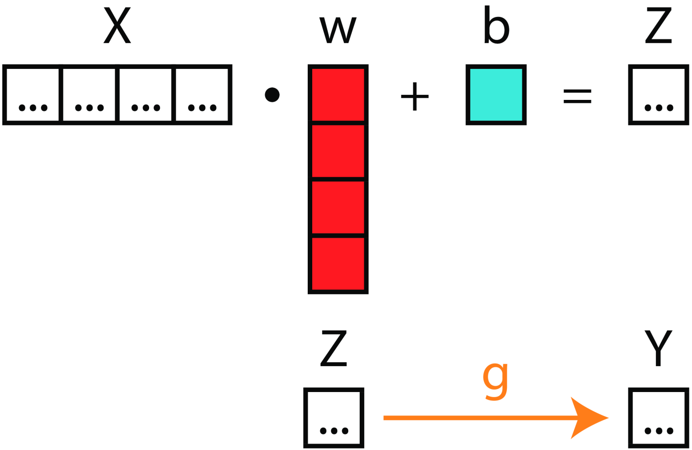
Can we do ensemble learning with infinite number of neural networks?

Outline

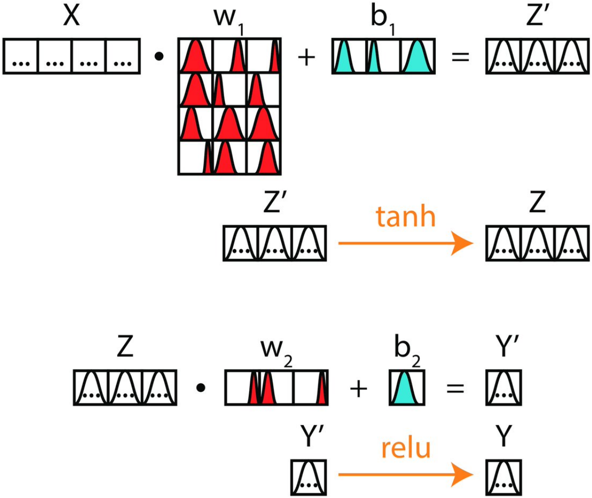
BNNs



BNNs



BNNs



BNNs

Let $\mathcal{D} = \{(x_i, y_i) : i = 1, \dots, n\}$ be our training data.

BNNs

Let $\mathcal{D} = \{(x_i, y_i) : i = 1, \ldots, n\}$ be our training data. Recall MLE:

$$w^{MLE} = \underset{w}{\text{argmax}}\ p\left(\mathcal{D}|w\right) = \underset{w}{\text{argmax}} \prod_{i=1}^{n} p\left(y_i|x_i, w\right)$$

$$= \underset{w}{\text{argmax}} \sum_{i=1}^{n} \log p\left(y_i|x_i, w\right)$$

BNNs

Let $\mathcal{D} = \{(x_i, y_i) : i = 1, \ldots, n\}$ be our training data. Recall MLE:

$$w^{MLE} = \underset{w}{\operatorname{argmax}}\ p(\mathcal{D}|w) = \underset{w}{\operatorname{argmax}} \prod_{i=1}^{n} p(y_i|x_i, w)$$

$$= \underset{w}{\operatorname{argmax}} \sum_{i=1}^{n} \log p(y_i|x_i, w)$$

Here the weights of our model are fixed, but the data is viewed as a random variable. If we instead view the data as being fixed and the model weights as being a random variable, we can train to maximize the posterior distribution $p(w|\mathcal{D})$:

BNNs

Let $\mathcal{D} = \{(x_i, y_i) : i = 1, \ldots, n\}$ be our training data. Recall MLE:

$$w^{MLE} = \underset{w}{\operatorname{argmax}}\ p\left(\mathcal{D}|w\right) = \underset{w}{\operatorname{argmax}} \prod_{i=1}^{n} p\left(y_i|x_i, w\right)$$

$$= \underset{w}{\operatorname{argmax}} \sum_{i=1}^{n} \log p\left(y_i|x_i, w\right)$$

Here the weights of our model are fixed, but the data is viewed as a random variable. If we instead view the data as being fixed and the model weights as being a random variable, we can train to maximize the posterior distribution $p\left(w|\mathcal{D}\right)$:

$$w^{MAP} = \underset{w}{\operatorname{argmax}}\ p\left(w|\mathcal{D}\right) = \underset{w}{\operatorname{argmax}}\ \frac{p\left(\mathcal{D}|w\right) p\left(w\right)}{p\left(\mathcal{D}\right)}$$

$$= \underset{w}{\operatorname{argmax}} \left(\log p\left(\mathcal{D}|w\right) + \log p\left(w\right)\right).$$

BNNs

We will construct a new distribution for $q\left(w|\theta\right)$, for approximating $p\left(w|\mathcal{D}\right)$.

We will construct a new distribution for $q\left(w|\theta\right)$, for approximating $p\left(w|\mathcal{D}\right)$. So we need to do the following optimization:

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\ KL\left(q\left(w|\theta\right)||p\left(w|\mathcal{D}\right)\right)$$

BNNs

We will construct a new distribution for $q(w|\theta)$, for approximating $p(w|\mathcal{D})$. So we need to do the following optimization:

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\ KL\left(q\left(w|\theta\right)||p\left(w|\mathcal{D}\right)\right)$$

$$= \underset{\theta}{\operatorname{argmin}} \left(KL\left(q\left(w|\theta\right)||p\left(w\right)\right) - \mathbb{E}_{q(w|\theta)}\left[\log p\left(\mathcal{D}|w\right)\right]\right)$$

We will construct a new distribution for $q(w|\theta)$, for approximating $p(w|\mathcal{D})$. So we need to do the following optimization:

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \; KL\left(q(w|\theta) \,||\, p(w|\mathcal{D})\right)$$

$$= \underset{\theta}{\operatorname{argmin}} \left(KL\left(q(w|\theta) \,||\, p(w)\right) - \mathbb{E}_{q(w|\theta)}\left[\log p(\mathcal{D}|w)\right]\right)$$

We will assume that prior $p(w)$ is mixture of two Gaussians:

$$p(w) = \prod_j \left(\alpha \mathcal{N}\left(w_j|0, \sigma_1^2\right) + (1-\alpha)\mathcal{N}\left(w_j|0, \sigma_2^2\right)\right)$$

where the first mixture component of the prior is given a larger variance than the second: $\sigma_1 > \sigma_2$.

BNNs

1. Sample $\epsilon \sim \mathcal{N}(0, I)$.
2. Let $\mathbf{w} = \mu + \log(1 + \exp(\rho)) \circ \epsilon$.
3. Let $\theta = (\mu, \rho)$.
4. Let $f(\mathbf{w}, \theta) = \log q(\mathbf{w}|\theta) - \log P(\mathbf{w})P(\mathcal{D}|\mathbf{w})$.
5. Calculate the gradient with respect to the mean

$$\Delta_\mu = \frac{\partial f(\mathbf{w}, \theta)}{\partial \mathbf{w}} + \frac{\partial f(\mathbf{w}, \theta)}{\partial \mu}. \tag{3}$$

6. Calculate the gradient with respect to the standard deviation parameter ρ

$$\Delta_\rho = \frac{\partial f(\mathbf{w}, \theta)}{\partial \mathbf{w}} \frac{\epsilon}{1 + \exp(-\rho)} + \frac{\partial f(\mathbf{w}, \theta)}{\partial \rho}. \tag{4}$$

7. Update the variational parameters:

$$\mu \leftarrow \mu - \alpha \Delta_\mu \tag{5}$$

$$\rho \leftarrow \rho - \alpha \Delta_\rho. \tag{6}$$

Outline

- We can easily collect very large amounts of unlabeled text data.

- We can easily collect very large amounts of unlabeled text data.
- Can we learn useful representations (e.g., word embeddings) from unlabeled data?

Bigrams from Unlabeled Data

- Given a corpus, extract a training set $(x_i, y_i)_{i=1}^{n}$, where $x_i, y_i \in \mathcal{V}$ and $\mathcal{V}$ is the vocabulary.

Bigrams from Unlabeled Data

- Given a corpus, extract a training set $(x_i, y_i)_{i=1}^{n}$, where $x_i, y_i \in \mathcal{V}$ and $\mathcal{V}$ is the vocabulary.
- For example:

Hispaniola quickly became an important base from which Spain expanded its empire into the rest of the Western Hemisphere.

Given a window size of $+/-$ 3, for $x =$ base we get the pairs

(base, became), (base, an), (base, important),
(base, from), (base, which), (base, Spain).

The Skip-gram Model

- The training objective of the Skip-gram model is to find word representations that are useful for predicting the surrounding words in a sentence or a document.

The Skip-gram Model

- The training objective of the Skip-gram model is to find word representations that are useful for predicting the surrounding words in a sentence or a document.
- More formally, given a sequence of training words $w_1, \ldots, w_T$, the objective of the Skip-gram model is to maximize the average log probability

$$\frac{1}{T} \sum_{t=1}^{T} \sum_{-c \leq j \leq c, j \neq 0} \log p\left(w_{t+j} | w_t\right)$$

where c is the size of the training context.

The Skip-gram Model

- The training objective of the Skip-gram model is to find word representations that are useful for predicting the surrounding words in a sentence or a document.
- More formally, given a sequence of training words $w_1, \ldots, w_T$, the objective of the Skip-gram model is to maximize the average log probability

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p\left(w_{t+j}|w_t\right)$$

  where c is the size of the training context.
- We model $p\left(w_{t+j}|w_t\right)$ using the softmax function:

$$p\left(w_O|w_I\right) = \frac{\exp\left(v'^{T}_{w_O} v_{w_I}\right)}{\sum\limits_{w=1}^{W}\exp\left(v'^{T}_{w} v_{w_I}\right)},$$

  where v_w and v'_w are the “input” and “output” vector representations of w, and W is the number of words in the vocabulary.

Negative Sampling

Negative Sampling

- We define Negative sampling by the objective

$$\log \sigma \left(v'^{T}_{w_O} v_{w_I} \right) + \sum_{i=1}^{k} \mathbb{E}_{w_i \sim P_n(w)} \left[\log \sigma \left(-v'^{T}_{w_i} v_{w_I} \right) \right]$$

which is used to replace every $\log p \left(w_O | w_I \right)$ term in the Skip-gram objective.

Negative Sampling

- We define Negative sampling by the objective

$$\log \sigma\left(v_{w_O}'^{T} v_{w_I}\right) + \sum_{i=1}^{k} \mathbb{E}_{w_i \sim P_n(w)}\left[\log \sigma\left(-v_{w_i}'^{T} v_{w_I}\right)\right]$$

  which is used to replace every $\log p\left(w_O | w_I\right)$ term in the Skip-gram objective.
- Thus the task is to distinguish the target word w_O from draws from the noise distribution $P_n\left(w\right)$ using logistic regression, where there are k negative samples for each data sample.

Negative Sampling

- We define Negative sampling by the objective

$$\log \sigma\left(v_{w_O}'^{T} v_{w_I}\right) + \sum_{i=1}^{k} \mathbb{E}_{w_i \sim P_n(w)}\left[\log \sigma\left(-v_{w_i}'^{T} v_{w_I}\right)\right]$$

  which is used to replace every $\log p\left(w_O | w_I\right)$ term in the Skip-gram objective.
- Thus the task is to distinguish the target word w_O from draws from the noise distribution $P_n\left(w\right)$ using logistic regression, where there are k negative samples for each data sample.
- In the original paper authors chose P_n to be the unigram distribution raised to the $3/4$rd power.

Thank you!