Energy-Based Models

Part 1

Introduction

Yann LeCun

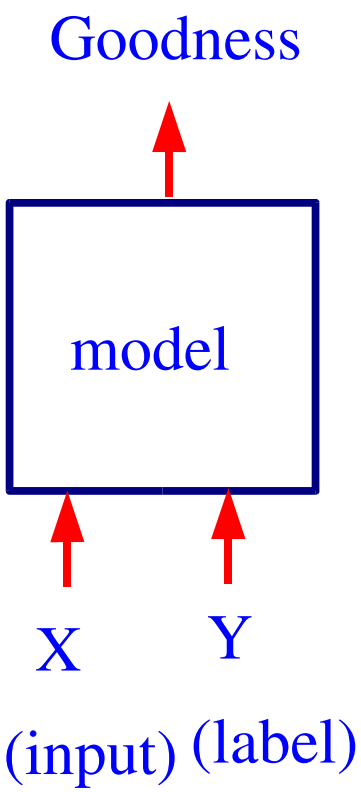
The Courant Institute of Mathematical Sciences

New York University

http://yann.lecun.com

http://www.cs.nyu.edu/~yann

A Model is Designed and trained to Answer Questions



- **Example:** X is an image from a camera; Y is a discrete variable e.g. Y in {animal, human, plane, truck, car}.
- 1. **Best Guess for Y:** which category best describes X?
- 2. **Ranking on Y:** Is X more a car than an airplane?
- 3. **Distribution on Y:** give an estimate of P(animal | X)
- 4. **Best Guess for X:** among all images of airplane, give me the best one.
- 5. **Ranking on X:** is this image more of a truck than that one?
- 6. **Distribution on X:** among all images of airplanes, how likely is this one?

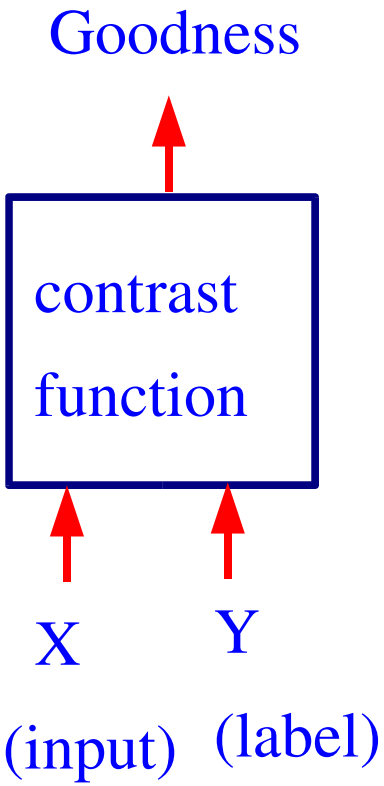
For each question, a different learning strategy is required

Do not answer a more complex question than necessary

- 1. Best Guess for Y: which category best describes X?
- 2. Ranking on Y: Is X more a car than an airplane?
- 3. Distribution on Y: give an estimate of P(animal | X)
- 4. Best Guess for X: among all images of airplane, give me the best one.
- 5. Ranking on X: is this image more of a truck than that one?
- 6. Distribution on X: among all images of airplanes, how likely is this one?

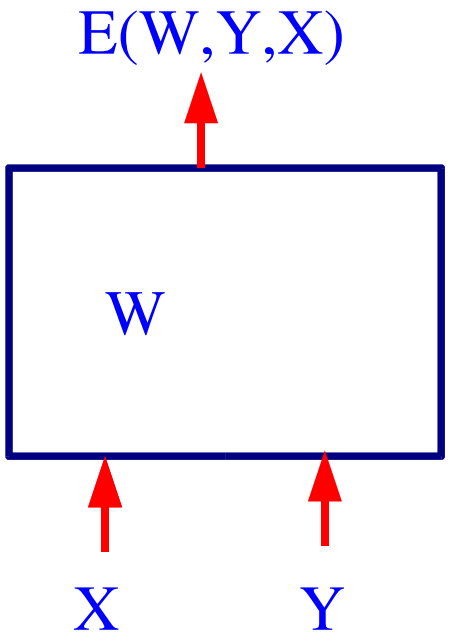
- The questions are in increasing order of complexity.
- The machine should be designed and trained to answer the simplest question possible.

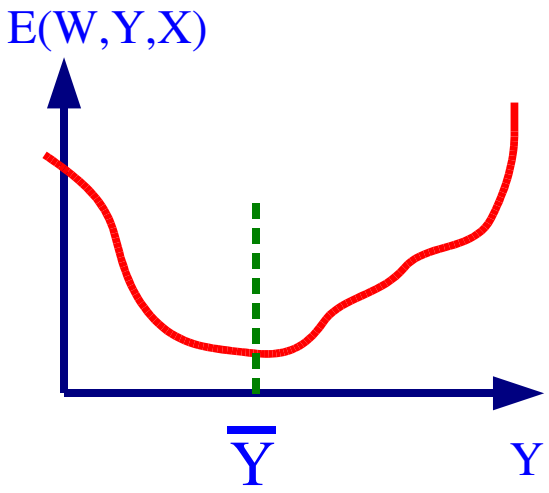
What is a model?



- A model measures the goodness of a combination of observed variables (X) and variables to be predicted (Y).
- Probabilistic approaches compute the distribution P(Y|X), and choose the Y that maximizes it.
- Sometimes, we do not need probabilities.
- Example: driving a robot. When the robot faces an obstacle, it MUST turn left of right. Computing a distribution of steering angles is of little use.
- Question: why estimate the whole distribution P(Y|X) when we are only interested in picking the best value of Y?

Energy-Based Models





- **E(W,Y,X):** is a scalar energy function (a.k.a. Contrast function) that measures the "compatibility" between Y and X.
- **W** is the parameter to be learned.
- **MAP Inference**: Given an input X, find the value of Y that minimizes the energy:

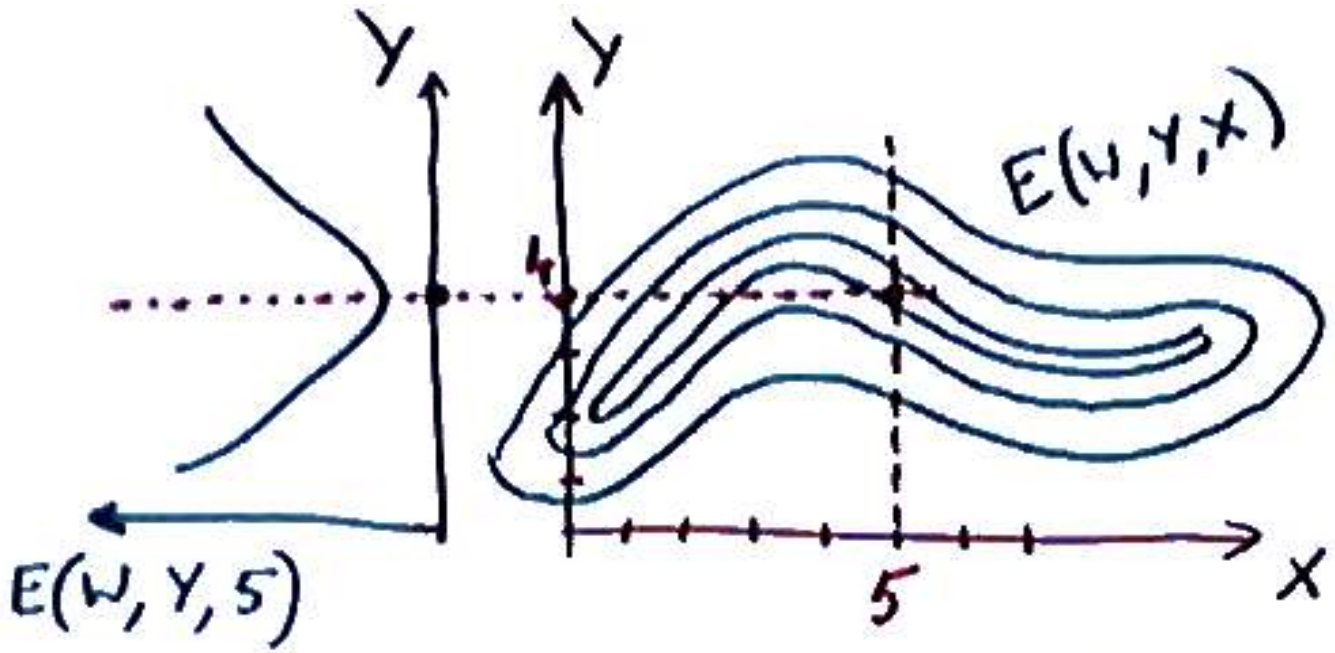
$$\check{Y} = \text{argmin}_{y \in \{Y\}} E(W, y, X)$$

- **Probabilistic Prediction**: Given an input X, compute the conditional distribution over Y (Gibbs Distribution):

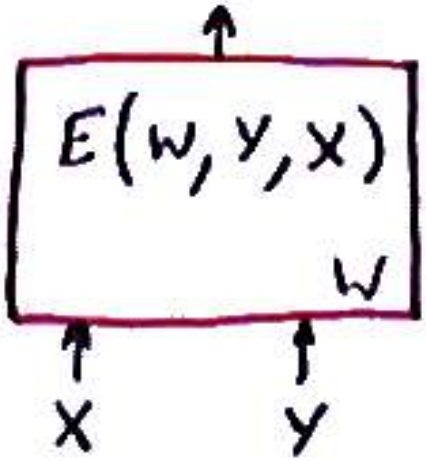
$$P(Y|X) = \frac{\exp\left(-\beta E(W, Y, X)\right)}{\sum_{y \in \{Y\}} \exp\left(-\beta E(W, y, X)\right)}$$

- **For decision making, we need no normalization.**

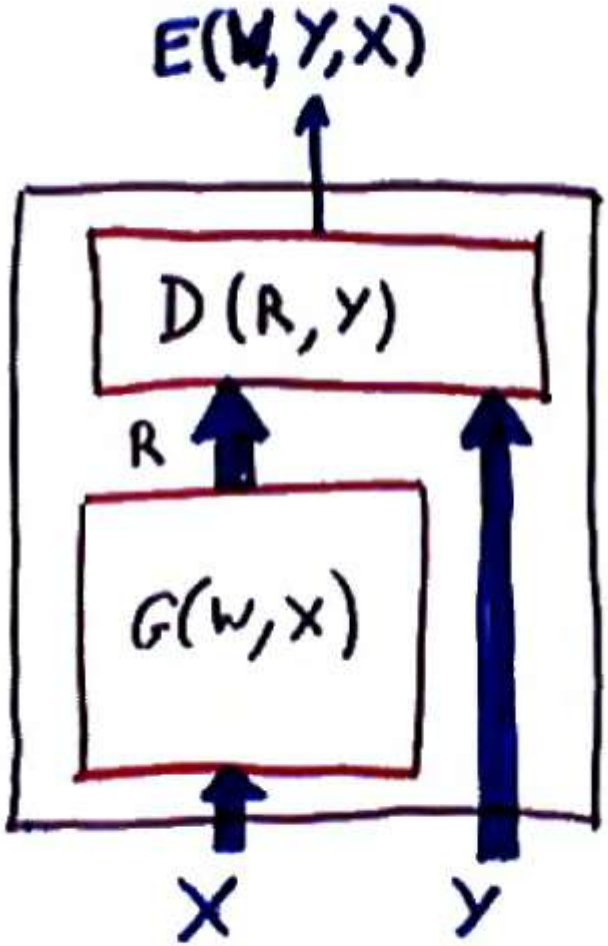
MAP Inference with Energy-Based Models



$$\check{y} = \min_{y} E(w, y, x)$$

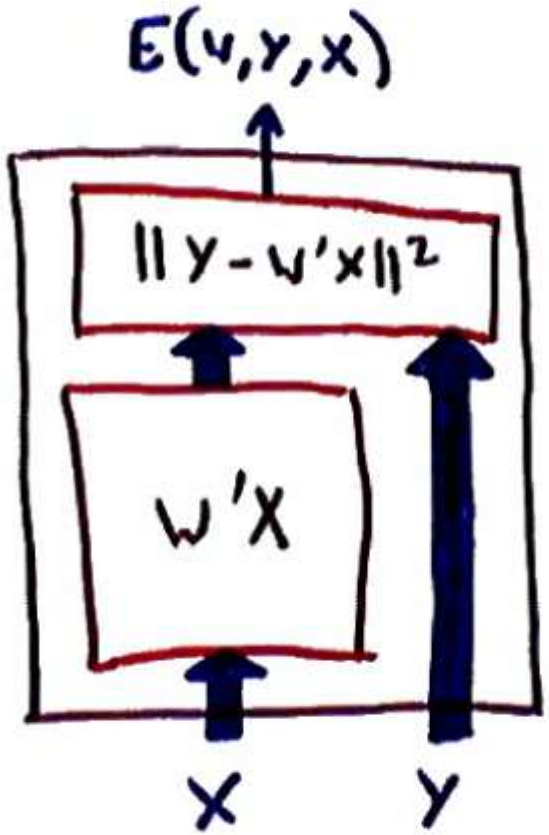


Examples of EBM: Regressor



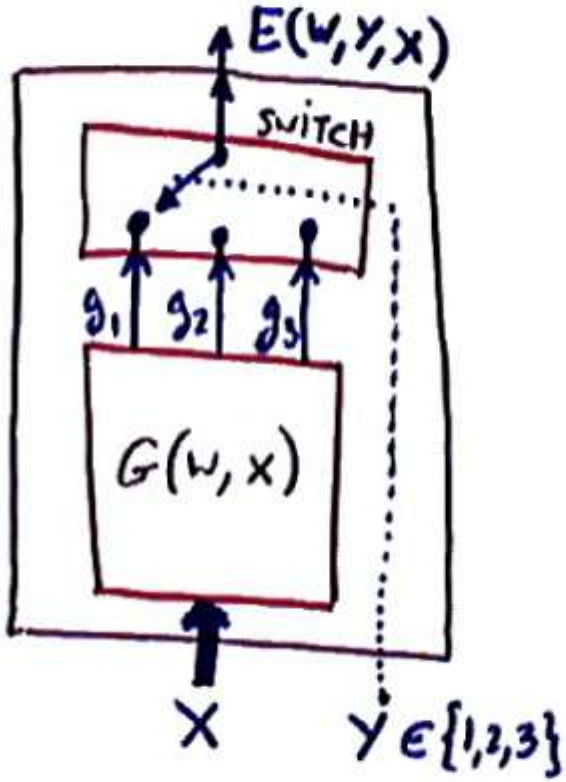
- X and Y are vectors or other entities
- Energy: $E(W, Y, X) = D(Y, G(W, X))$ where $D(Y, R)$ is a distance or dissimilarity measure.
- Best output: $\check{Y} = \min_Y E(W, Y, X) = G(W, X)$.

Examples of EBM Regressor: Linear Regression



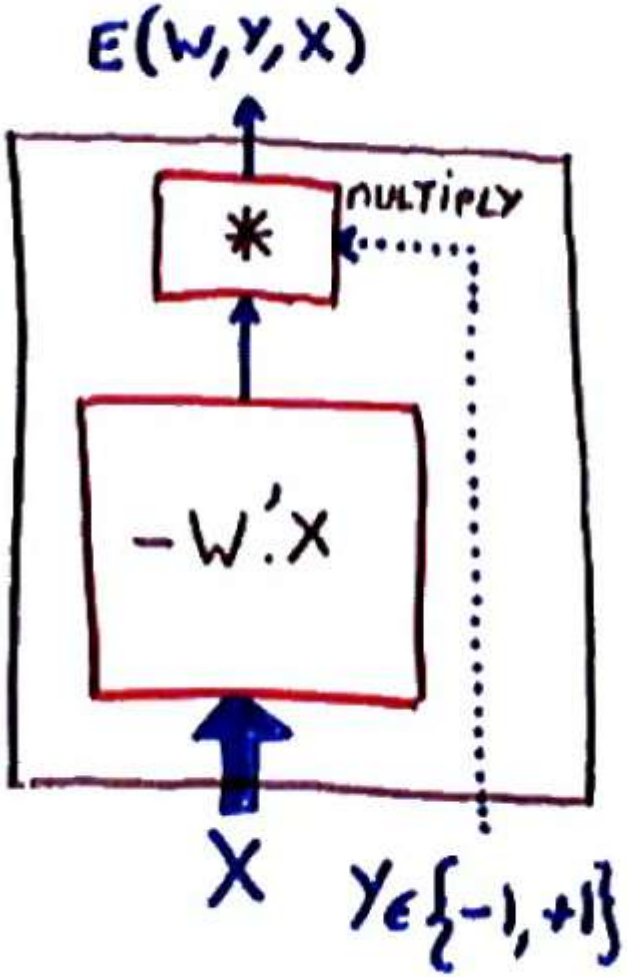
- X and Y are vectors
- Energy: $E(W, Y, X) = ||Y - W'X)||^2$.
- Best output: $\check{Y} = \min_Y E(W, Y, X) = W'X$.

Examples of EBM: Classifier



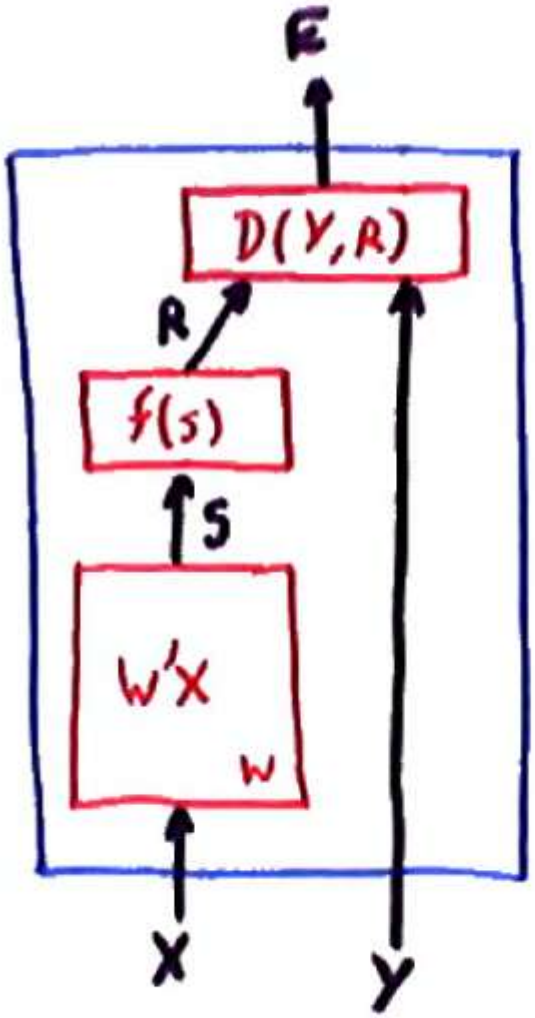
- Y is a discrete variable, $\{Y\} = \{1, 2, 3\}$.
- Energy: $E(W, Y, X) = \sum_k G_k(W, X)\delta(k, Y)$, where $\delta(k, Y) = 1$ iff $k = Y$ and 0 otherwise.
- $G_k(W, X)$, the k-th component of the output vector of $G(W, X)$ is interpreted as the "cost" of classifying X into category k.
- Best output: $\check{Y} = \min_{Y \in \{Y\}} E(W, Y, X) = \min_k G_k(W, X)$.

Examples of EBM Classifier: Perceptron



- Y is a discrete variable, $\{Y\} = \{-1, +1\}$.
- Energy: $E(W, Y, X) = -Y.W'X$.
- Best output: $\check{Y} = \text{sign}(W'X)$, where $\text{sign}(R) = +1$ iff $R > 0$ and -1 otherwise.

Linear Machines

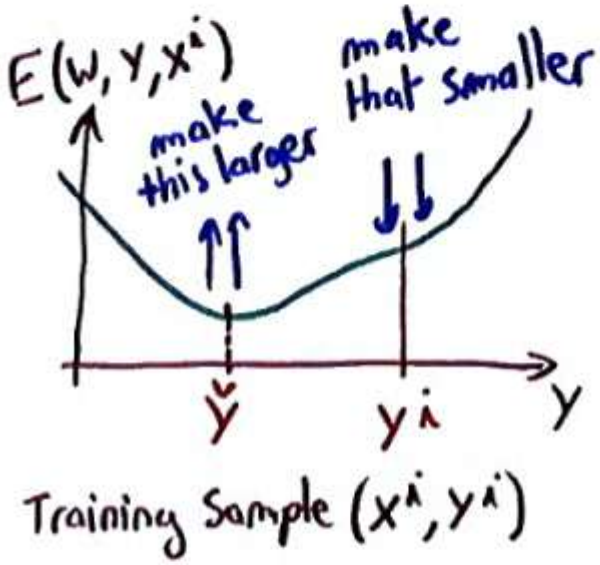


- The learning algorithms we have seen so far (perceptron, linear regression) are of that form, with the assumption that $G(W, X)$ only depends on the dot product of W and X.
- In other words, The E function of 2-class linear classifiers can be written as:

$$E(Y, X, W) = D(Y, f(W'X))$$

  where $W'X$ is the dot product of vectors W and X, and f is a monotonically increasing scalar function.
- in the following, we assume $Y = -1$ for class 1, and $Y = +1$ for class 2.

Training Energy-Based Models



- To train an EBM, we minimize a **loss function**, which is an average over training samples of a **per-sample loss function** $L(W, Y^i, X^i)$:

$$\mathcal{L}(W, \mathcal{S}) = \frac{1}{P} \sum_{i=1}^{P} L(W, Y^i, X^i)$$

- The loss function must be designed so that minimizing it with respect to W will make the machine approach the desired behavior.

To ensure this, we pick loss functions that, for a given training input X^i, will drive the energies $E(W, Y^i, X^i)$ associated with the desired output Y^i to be lower than the energies associated with all other (undesired) outputs values $E(W, Y, X^i)$ for all $Y \neq Y^i, Y \in \{Y\}$.

Form of the Loss Function

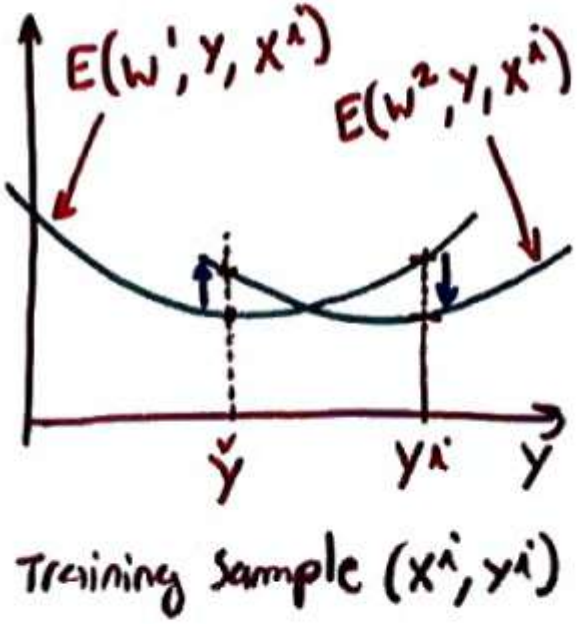
- We assume that the per-sample loss function $L(W, Y^i, X^i)$ has a lower bound over W for all Y^i, X^i.
- We assume that L depends on X^i only indirectly through the set of energies $\{E(W, Y, X^i)\ ,\ Y \in \{Y\}\}$.
- For example, if $\{Y\}$is the set of integers between 0 and $k-1$ (as would be the case for a classifier with k categories), the per-sample loss for sample (X^i, Y^i) should be of the form:

$$L(W, Y^i, X^i) = L(Y^i, E(W, 0, X^i), E(W, 1, X^i), \ldots, E(W, k-1, X^i))$$

- With this assumption, we separate the choice of the loss function from the details of the internal structure of the machine, and limit the discussion to how minimizing the loss function affects the energies.

Examples of Loss: Energy Loss

Energy Loss, the simplest of all losses: $L_{\text{energy}}(W, Y^i, X^i) = E(W, Y^i, X^i)$. This loss only works if $E(W, Y, X^i)$ has a special form which guarantess that making $E(W, Y^i, X^i)$ lower will automatically make $E(W, Y, X^i)$ for $Y \neq Y^i$ larger than the minimum.

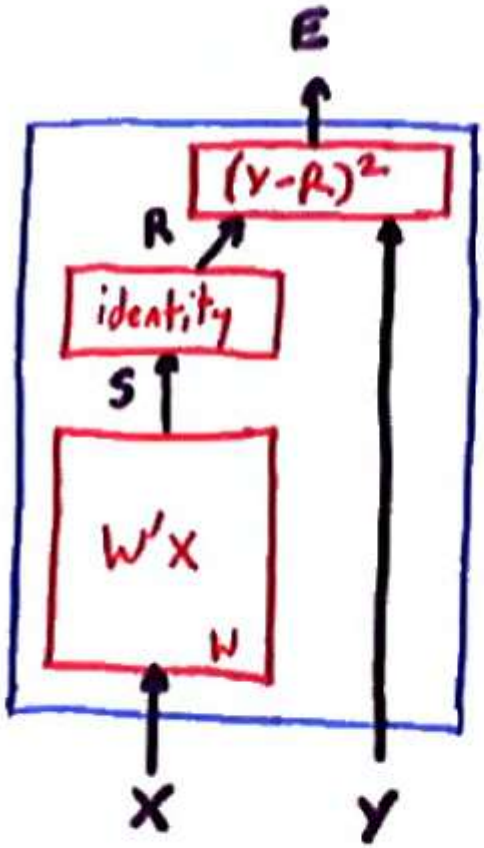


Example: if $E(W, Y, X)$ is quadratic in Y, as is the case for regression with squared error: $E(W, Y, X) = ||Y - G(W, X)||^2$,
Let $W(1)$ is the parameter before a learning update, and $W(2)$ the parameter after the learning update, and let $\check{Y} = \min_Y E(W(1), Y, X)$. Then,

$$E(W(2), Y^i, X^i) - E(W(2), \check{Y}, X^i) < E(W(1), Y^i, X^i) - E(W(1), \check{Y}, X^i)$$

Linear Regression

Linear regression uses the Energy loss

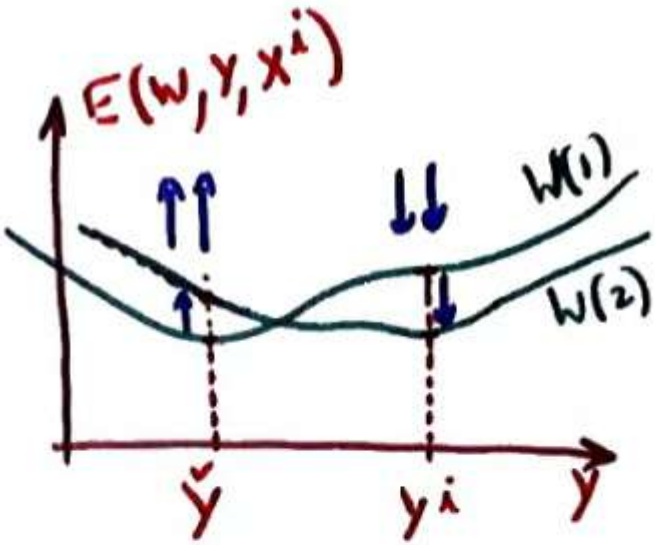


- $R = W'X$
- $E(W, Y, X) = D(Y, R) = \frac{1}{2}||Y - R||^2$
- $L(W, Y^i, X^i) = D(Y^i, W'X^i)$
- $\frac{\partial L}{\partial W} = \frac{\partial D(Y^i,R)}{\partial R} \frac{\partial R}{\partial W}$
- $\frac{\partial L}{\partial W} = \frac{\partial D(Y^i,R)}{\partial R} \frac{\partial (W'X^i)}{\partial W} = (R - Y^i)X^i$
- descent: $W \leftarrow W + \eta(Y^i - R)X^i$

Examples of Loss: Perceptron Loss

Perceptron Loss:

$$L_{\text{perceptron}}(W, Y^i, X^i) = E(W, Y^i, X^i) - \min_{Y \in \{Y\}} E(W, Y, X^i)$$

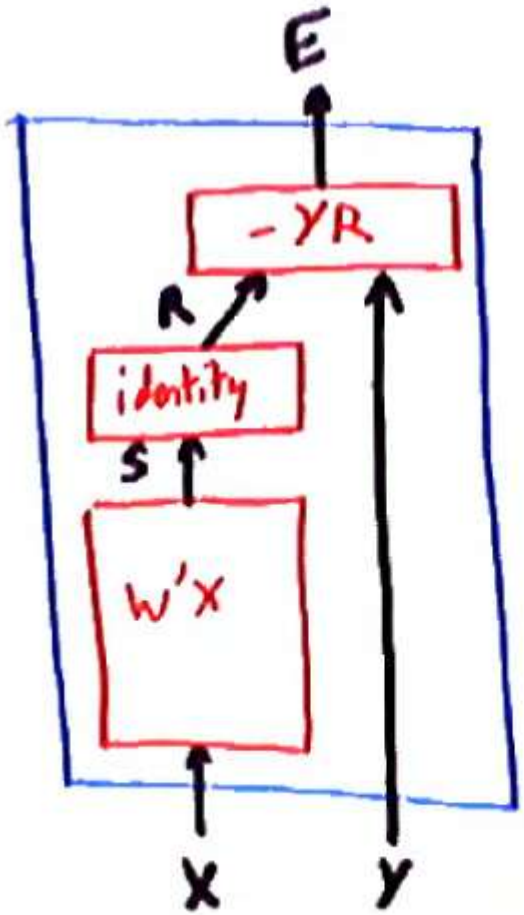


Adjust W so that $E(W, Y^i, X^i)$ gets smaller, while $\check{Y} = \min_{Y \in \{Y\}} E(W, Y, X^i)$ gets bigger (or more precisely, so that the difference decreases).
This algorithm makes no update whenever the energy of the desired Y is lower than all the others.

Perceptron

$$L_{\text{perceptron}}(W, Y^i, X^i) = E(W, Y^i, X^i) - \min_{Y \in \{Y\}} E(W, Y, X^i)$$

$\{Y\} = \{-1, +1\}$.



- $R = W'X$
- $E(Y, X, W) = D(Y, R) = -YR$
- $Y \in \{-1, +1\}$, hence $\min_Y -YR = -\text{sign}(R)R$ where $\text{sign}(R) = 1$ iff $R > 0$, and -1 otherwise.
- $L(W, Y^i, X^i) = -(Y^i - \text{sign}(R))R$
- $\frac{\partial L}{\partial W} = \frac{\partial -(Y^i - \text{sign}(R))R}{\partial R} \frac{\partial R}{\partial W}$
- $\frac{\partial L}{\partial W} = -(Y^i - \text{sign}(W'X^i))X^i$
- descent: $W \leftarrow W + \eta(Y^i - \text{sign}(W'X^i))X^i$

Examples of Loss: Log-Likelihood Loss

Log-Likelihood Loss:

$$L_{\text{ll}}(W, Y^i, X^i) = E(W, Y^i, X^i) + \frac{1}{\beta} \log \left(\sum_{Y \in \{Y\}} \exp(-\beta E(W, Y, X^i)) \right)$$

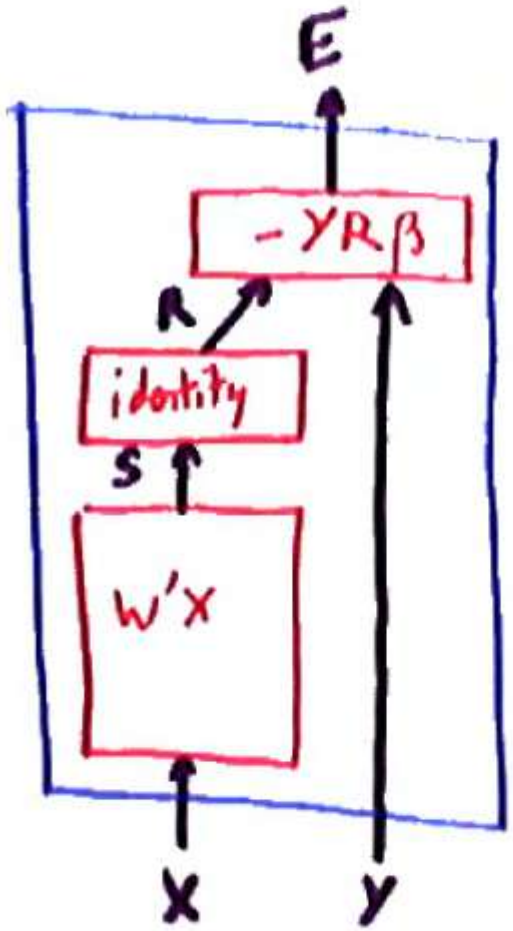
where β is a positive constant.

- The function $\mathcal{F}_\beta(\{Y\}) = \frac{1}{\beta} \log \left(\sum_{Y \in \{Y\}} \exp(-\beta E(W, Y, X^i)) \right)$ is called the **free energy** of the ensemble $\{Y\}$ for temperature $1/\beta$.
- We define $\mathcal{Z}_\beta(\{Y\}) = \sum_{Y \in \{Y\}} \exp(-\beta E(W, Y, X^i))$ as the **partition function** of ensemble $\{Y\}$.
- Interesting property # 1: $\mathcal{F}_\beta(\{Y\}) = \frac{1}{\beta} \log \mathcal{Z}_\beta(\{Y\})$
- Interesting property # 2: $\lim_{\beta \to \infty} \mathcal{F}_\beta(\{Y\}) = \min_{Y \in \{Y\}} E(W, Y, X^i)$

For very large β, the log-likelihood loss reduces to the Perceptron loss.

Logistic Regression (a.k.a MaxEnt)

$$L_{\text{ll}}(W) = E(Y^i, X^i, W) + \log\left(\sum_{Y \in \{Y\}} \exp(-E(W, Y, X^i))\right)$$



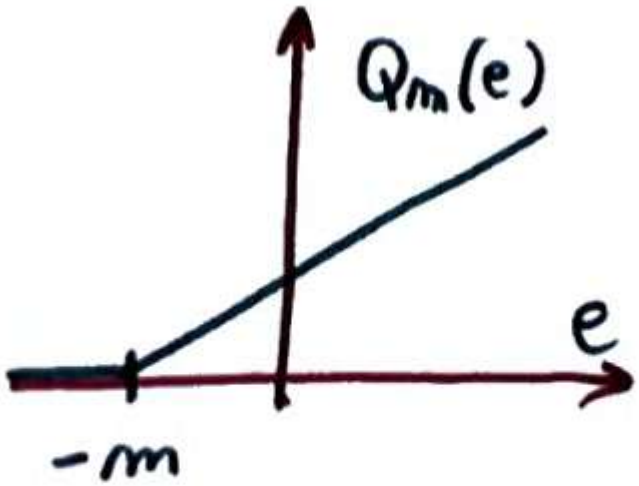
- $R = \frac{1}{2} W'X$
- $E(Y, X, W) = D(Y, R) = -\frac{1}{2} YR = -\frac{1}{2} YW'X$
- $L(W) = \log(1 + \exp(-Y^i W' X^i))$
- $\frac{\partial L}{\partial W} = \frac{\partial D(Y^i, R)}{\partial R} \frac{\partial S}{\partial W}$
- $\frac{\partial L}{\partial W} = -\left(\frac{Y^i + 1}{2} - \frac{1}{1 + \exp(-W' X^i)}\right) X^i$
- descent: $W \leftarrow W + \eta \left(\frac{Y^i + 1}{2} - \frac{1}{1 + \exp(-W' X^i)}\right) X^i$

Examples of Loss: Margin Loss

Margin Loss: for discrete output set $\{Y\}$:

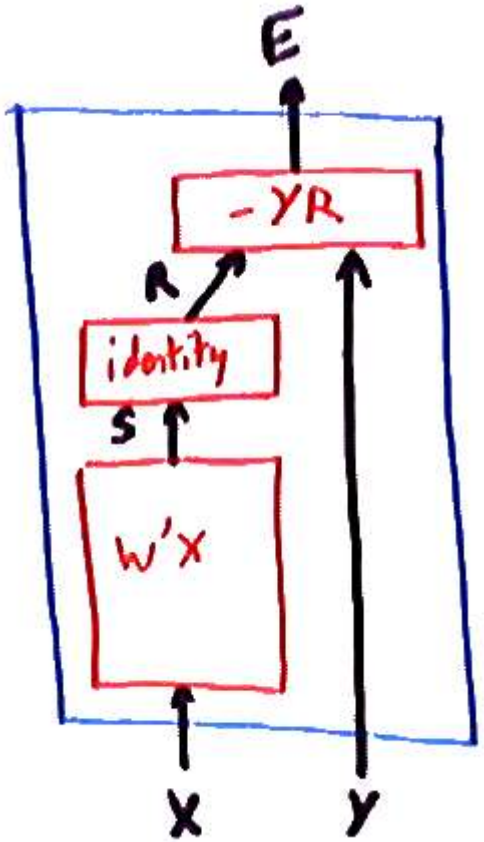
$$L_{\text{margin}}(W, Y^i, X^i) = Q_m\left(E(W, Y^i, X^i) - \min_{Y \in \{Y\}, Y \neq Y^i} E(W, Y, X^i)\right)$$

where $Q_m(e)$ is any function that is monotonically increasing for $e > -m$, where m is a constant called the **margin**.

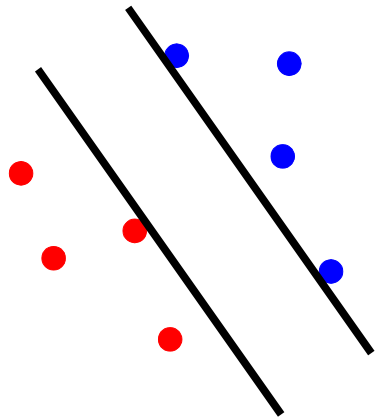


Adjust W so that $E(W, Y^i, X^i)$ gets smaller, while all $E(W, Y, X^i)$ for which $E(W, Y, X^i) - E(W, Y^i, X^i) < m$ get bigger. This guarantees that the energy of the desired Y will be smaller than all other energies by at least m.

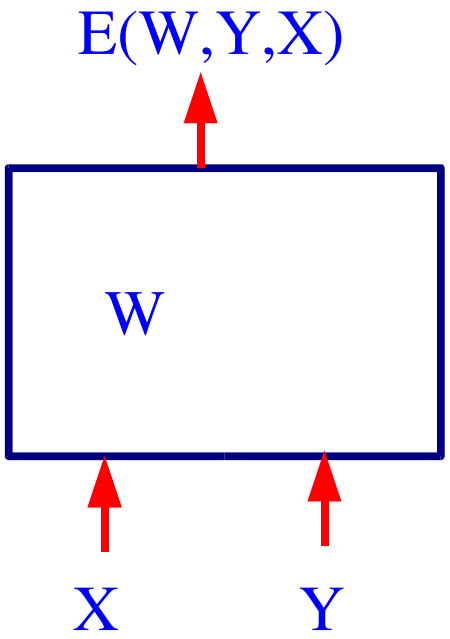
Linear Model + Margin Loss + Regularization = SVM



- Minimize the hinge loss: make the energy of all the "good" answers smaller that the energy of any "bad" answer by at least m (the margin).
- Minimize the Regularization term: Make W as short as possible.
- This is equivalent to keeping ||W|| constant, while maximizing m.

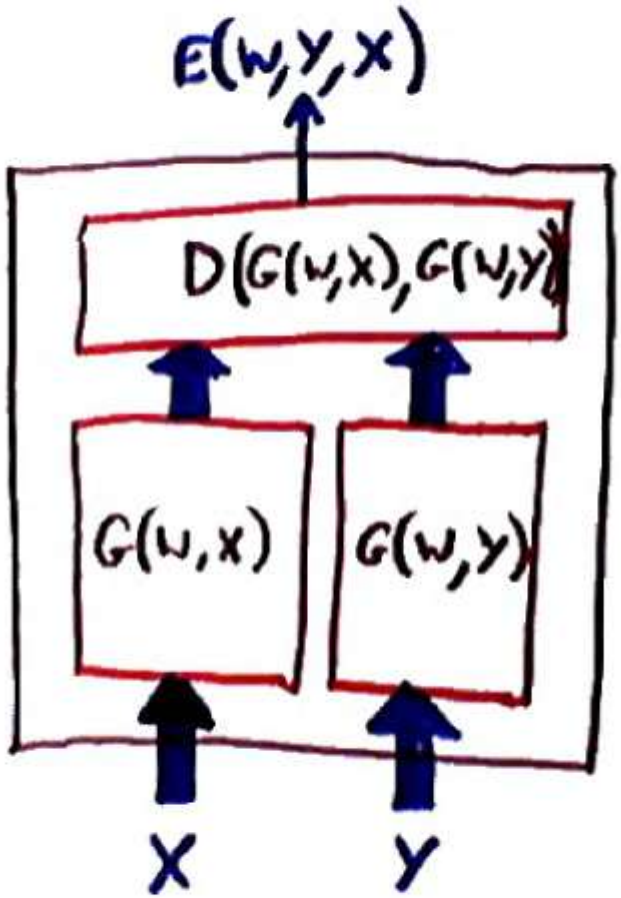


Architecture



- We can put anything we want in the box.
- The energy can be a very complicated non-linear function of X,Y, and W (e.g. A neural net, a graphical model, an HMM, Markov Random Field,....).
- The internal structure of the box is called the architecture of the model.

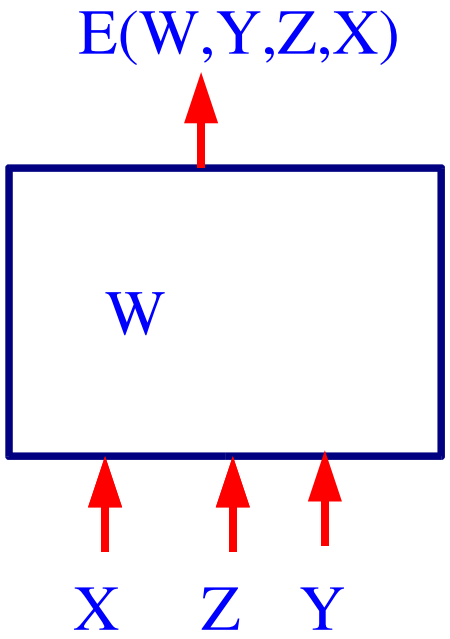
Examples of EBM: Matcher



- X and Y are vectors of the same dimension.
- Energy:
  $E(W, Y, X) = D(G(W, Y), G(W, X))$ where $D(.,.)$ is a distance or dissimilarity measure.
- Best output: $\check{Y} = \min_Y E(W, Y, X) = G^{(} - 1)(G(W, X))$.

Finding the Y that minimizes the energy may be non-trivial

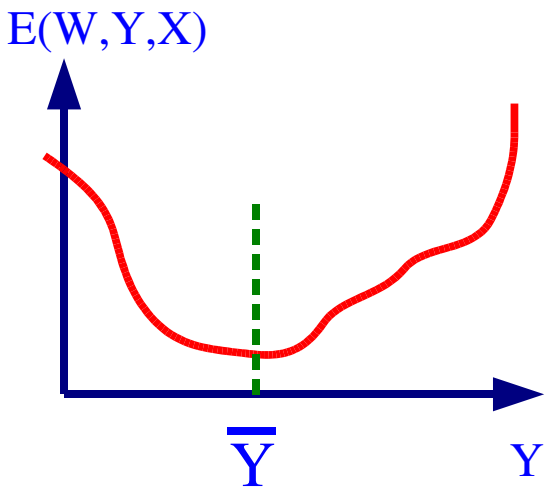
Energy-Based Models with Latent Variables



- **Z:** is the latent variable (never observed).
- **MAP inference:** given an input X, find the value of Y that minimizes the energy:

$$\check{Y} = \text{argmin}_{y \in \{Y\}} \check{E}(W, y, X)$$

$$\check{E}(W, y, X) = \text{argmin}_{z \in \{Z\}} E(W, y, z, X)$$



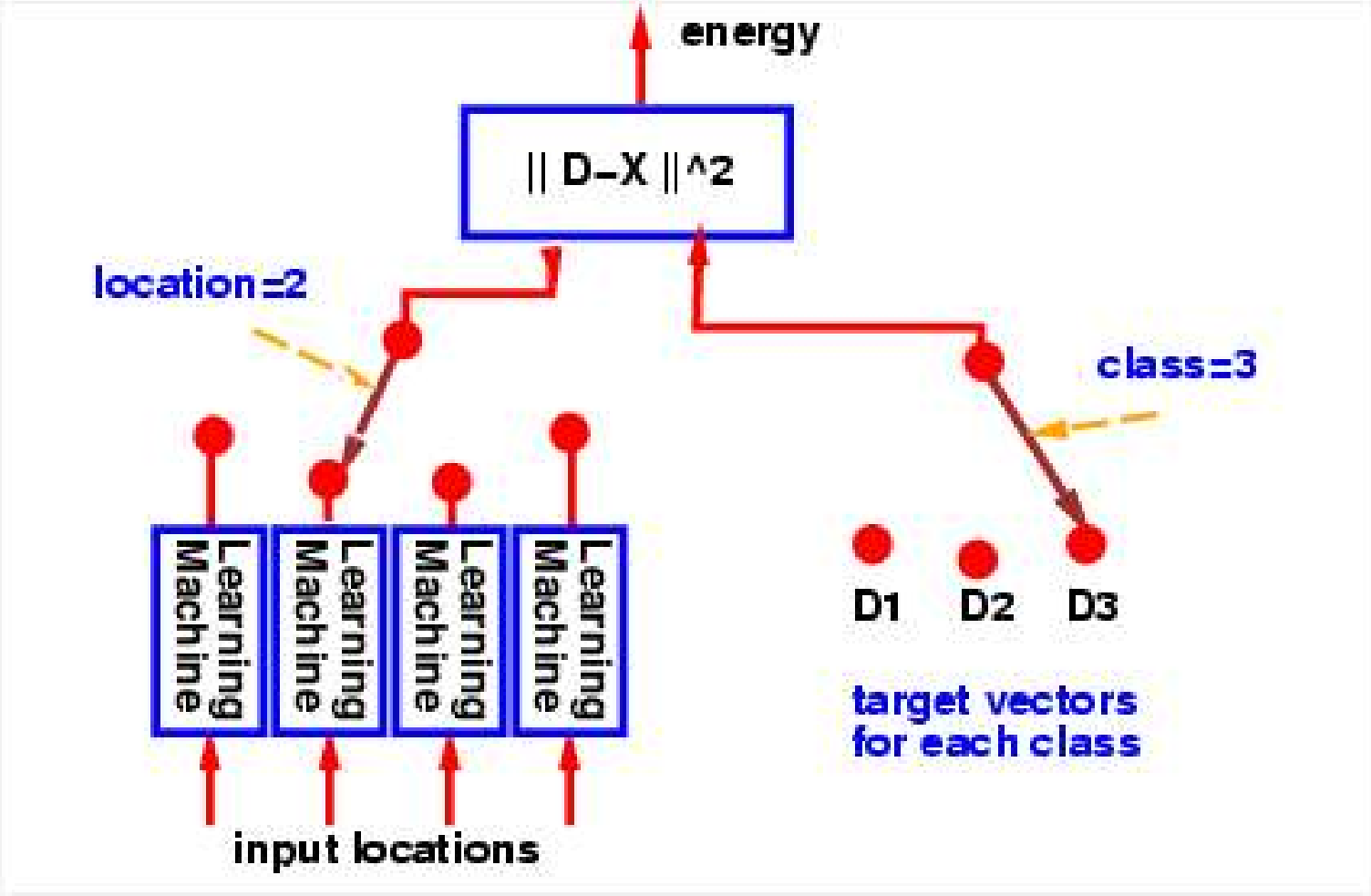
- **Probabilistic Inference**: simply marginalize over Z.

$$P(Y|X) = \frac{\int_{z \in \{Z\}} \exp\left(-\beta E(W, Y, z, X)\right)}{\sum_{y \in \{Y\}} \int_{z \in \{Z\}} \exp\left(-\beta E(W, y, z, X)\right)}$$

What can the latent variables represent?

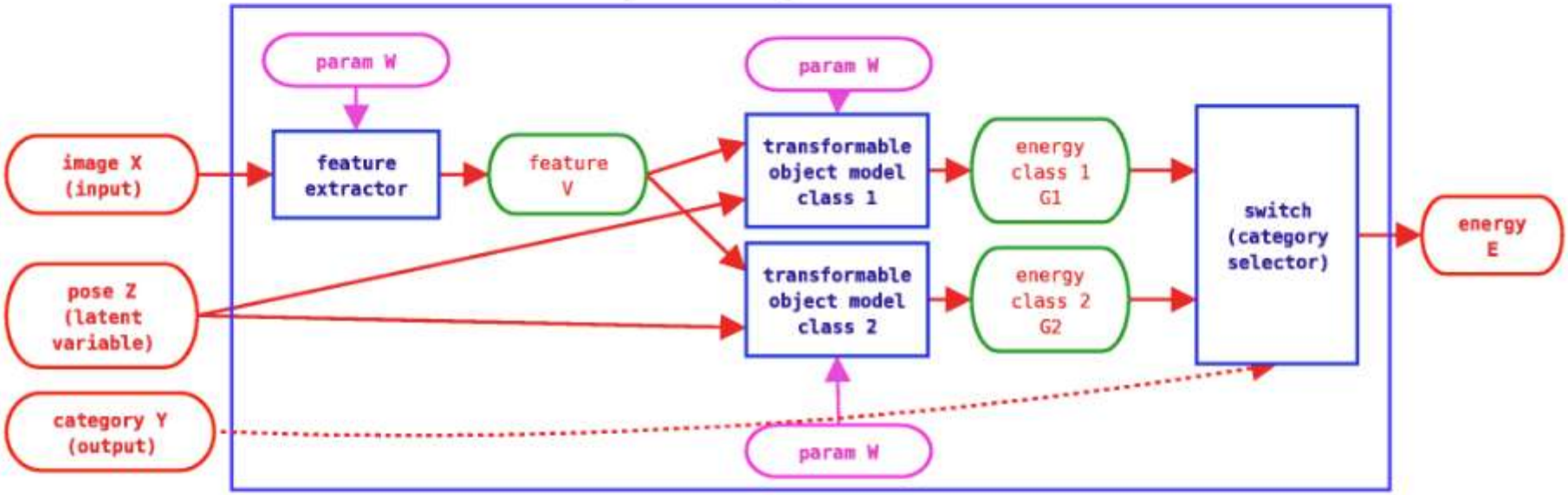
- **Variables that would make the task easier if they were known:**
  - **Face recognition**: the gender of the person, the orientation of the face.
  - **Object recognition**: the pose parameters of the object (location, orientation, scale), the lighting conditions.
  - **Parts of Speech Tagging**: the segmentation of the sentence into syntactic units, the parse tree.
  - **Speech Recognition**: the segmentation of the sentence into phonemes or phones.
  - **Handwriting Recognition**: the segmentation of the line into characters.
- **In general, we will search for the value of the latent variable that allows us to get an answer (Y) of smallest energy.**

Example of latent variable: location



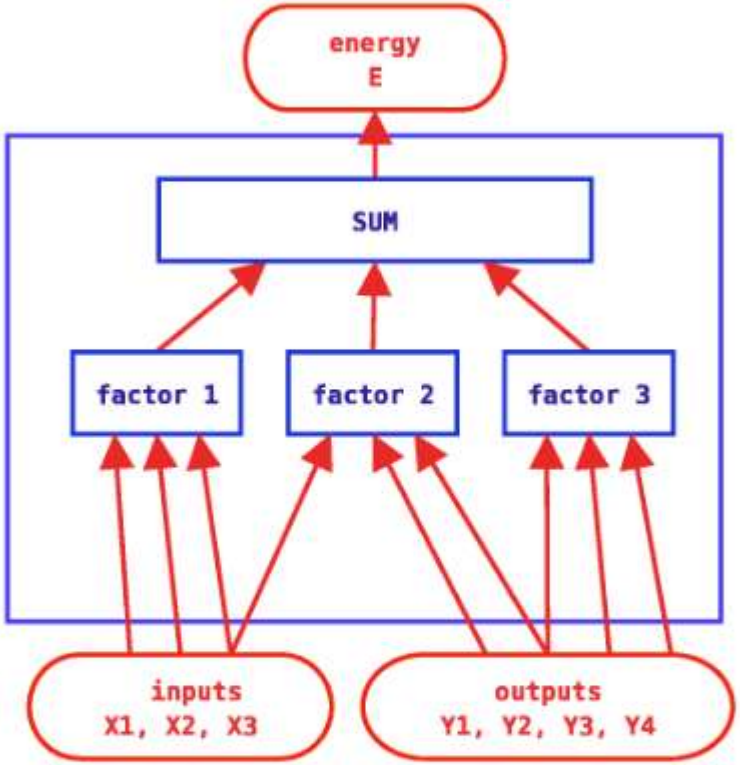
Example of latent variable: pose

EBM Architecture for invariant object recognition



Each object model matches the output of the feature extractor to a reference representation that is transformed by the pose parameters.
Inference finds the category and the pose that minimize the energy.

EBMs as Factor Graphs



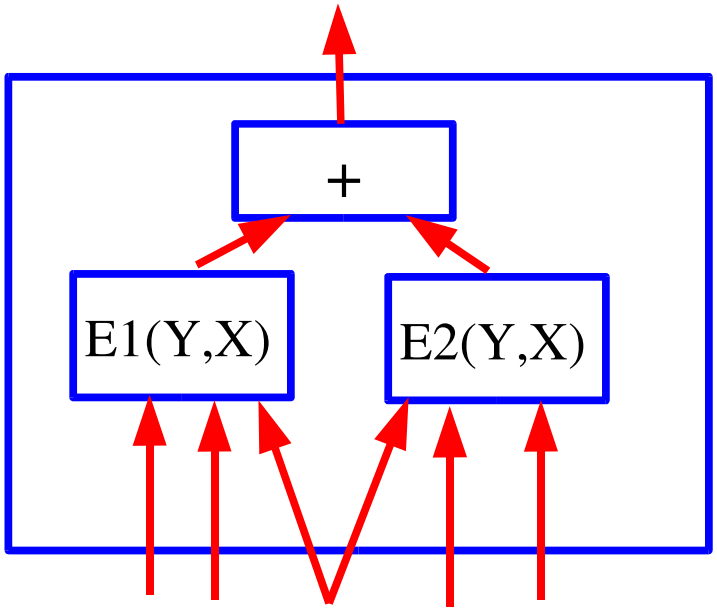
An EBM whose energy function can be "factorized" as a sum of individual functions (factors) is equivalent to a graphical model represented as a factor graph.
Any traditional graphical model can be formulated as a factor graph, but the converse is not true. Each factor is akin to $-log$ of the potential functions of a clique of variable nodes.

Efficient inference algorithms such as (loopy) belief propagation can be used to compute the marginals of Y, or the lowest energy (MAP) configuration [Kschischang, Frey, Loeliger, 2001].

Energy-Based Graphical Models

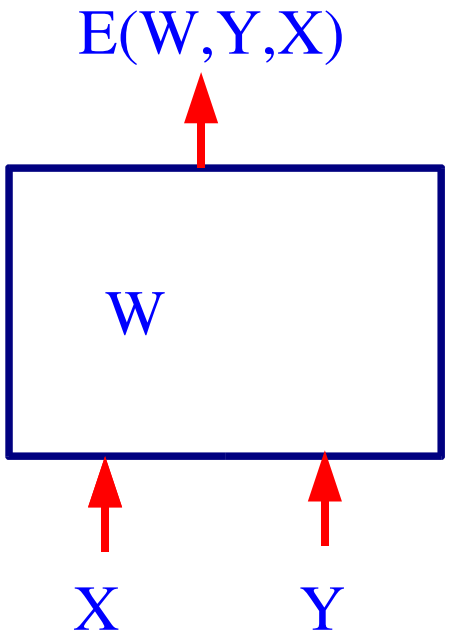
- **A factor graph is a general way to represent a graphical model.**
- **Probabilistic Factor Graph:**

$$P(Y|X) = \frac{\prod_i \Psi^i(Y,X)}{\int_y \prod_i \Psi^i(y,X)}$$



- **Energy-Based Factor Graph:**
  - No latent vars: $P(Y|X) = \frac{\exp(-\beta \sum_i E^i(Y,X))}{\int_y \exp(-\beta \sum_i E^i(y,X))}$
  - Latent vars $P(Y|X) = \frac{\int_z \exp(-\beta \sum_i E^i(Y,z,X))}{\int_{y,z} \exp(-\beta \sum_i E^i(y,z,X))}$

Architecture + Inference Algo + Loss Function = Model



- **1. Design an architecture:** a particular form for E(W,Y,X).
- **2. Pick an inference algorithm for Y:** MAP or conditional distribution, belief prop, min cut, variational methods, gradient descent, MCMC, HMC.....
- **3. Pick a loss function:** in such a way that minimizing it with respect to W over a training set will make the inference algorithm find the correct Y for a given X.
- **4. Pick an optimization method.**

- **PROBLEM: What loss functions will make the machine approach the desired behavior?**