Submission for Homework 3: Energy-Based Models

Jay Nitin Kaoshik
jnk7726@nyu.edu
N13143537

CSCI-GA 2572 Deep Learning

Fall 2024

1 Theory (50pt)

1.1 Energy Based Models Intuition (15pts)

This question tests your intuitive understanding of Energy-based models and their properties.

(a) (1pts) How do energy-based models allow for modeling situations where the mapping from input x_i to output y_i is not 1 to 1, but 1 to many, or even 1 to an infinite continuum of y?

Answer for 1.1 (a):

Energy-based models (EBMs) facilitate modeling scenarios where the relationship between the input x_i and output y_i is not strictly one-to-one by **associating input-output pairs with an energy** score. Instead of predicting a unique output for a given input, EBMs learn to assign a **lower energy** to more likely (**well-behaved**) plausible pairs (x, y), indicating that these combinations are preferable or have a higher probability. This approach inherently supports one-to-many mappings since multiple outputs can correspond to low-energy configurations for the same input. In fact, it can even model a continuum of possible y values as any number of y's may have sufficiently low energy, representing a range of feasible outcomes.

(b) (1pts) How do energy-based models differ from models that output probabilities?

Answer for 1.1 (b):

EBMs differ from probabilistic models in the sense that they **do not** directly **produce a probability distribution** over the possible outputs given an input. Instead of outputting the probability $p(y \mid x)$ of a specific label y for a given input x, EBMs provide a **raw energy score** $F_W(x, y)$ for the input-output pair (no normalization). This energy score reflects how compatible or plausible the pair is, with lower energy values indicating higher compatibility. In probabilistic models, the output is normalized to sum up to one across all possible classes, representing a true probability distribution. EBMs offer a more flexible approach, where the focus is on learning a function that assigns relative scores to different outputs, rather than directly modeling their probabilities since the output space may not be fixed.

(c) (2pts) How can you use energy function $F_W(x, y)$ to calculate a probability $p(y \mid x)$?

Answer for 1.1 (c):

To calculate the probability $p(y \mid x)$ using the energy function $F(x, y)$, we need to convert the energy values into a normalized probability distribution. This can be done using the **Gibbs-Boltzmann Distribution**, which defines the probability of a particular output y given an input x as:

$$p(y \mid x) = \frac{\exp(-\beta F(x, y))}{\int_{y'} \exp(-\beta F(x, y'))}$$

Here, β is a positive scaling parameter that controls the smoothness of the distribution. The numerator $\exp(-\beta F_W(x, y))$ represents the unnormalized probability associated with the energy of the pair (x, y), while the denominator (known as the partition function) normalizes these probabilities across all possible output values y'. This ensures that the probabilities sum up to one, converting energy-based scores into a probability distribution.

(d) (1pt) Is there a way to control the smoothness of the probability distribution $p(y|x)$ estimated from the energy function $F_W(x, y)$? How do we reduce the variance of $p(y|x)$?

Answer for 1.1 (d):

To control the smoothness of the probability distribution $p(y \mid x)$ derived from the energy function $F_W(x, y)$, the scaling parameter β in the Gibbs distribution can be adjusted as it serves as the **temperature** parameter influencing the distribution. By increasing β, the energy differences between different pairs are emphasized, resulting in a sharper distribution with lower variance (i.e., the model becomes more confident in a few outcomes). Conversely, decreasing β smooths the distribution, making it more uniform and increasing the variance. To reduce the variance of $p(y \mid x)$, we should **increase** the **value of** β. This adjustment will make the distribution more peaked around the most likely outputs, effectively decreasing the probability spread over other outputs and thus reducing the variance.

(e) (2pts) What are the roles of the loss function and energy function?

Answer for 1.1 (e):

Energy Function: The energy function evaluates how well a given **input-output pair fits** according to the model, assigning a numerical value that represents the "quality" of that pairing. Lower energy values signify higher **compatibility**, meaning the pair is more likely or plausible (**well-behaved**) under the model. The energy function thus acts as a scoring mechanism that ranks different input-output pairs based on their compatibility.

Loss Function: The loss function, on the other hand, is used during the training process to optimize the parameters of the model. It **quantifies the discrepancy** between the predicted outputs (or energy values) and the desired target values. By minimizing the loss function, we adjust the model parameters to **shape** the **energy landscape**, ensuring that lower energy values correspond to more desirable or correct outputs. Thus, the loss function guides the learning process, refining the energy function to distinguish between compatible and incompatible pairs during EBM training.

(f) (2pts) What problems can be caused by using only positive examples for energy (pushing down energy of correct inputs only)? How can it be avoided?

Answer for 1.1 (f):

Problems: Using only positive examples to push down the energy of correct inputs may lead to a corrupted solution where the model simply assigns a low (0) energy to all possible inputs, resulting in a **flat energy surface**. In such a case, the model would not effectively distinguish between correct and incorrect outputs because it does not have any incentive to push the energy of incorrect or less desirable outputs higher. This happens because the loss function only encourages lowering the energy for positive examples without any constraint or penalty for other cases, allowing the model to trivially minimize the loss by making the energy uniformly low.

Solution: To avoid the aforementioned issue, it is important that we also consider negative examples during training. This can be done by **pushing up the energy of incorrect** or less likely outputs while lowering the energy for correct outputs. A common approach is to use **contrastive** loss or margin-based objectives, which explicitly separate the energy levels of positive and negative examples by a certain **margin**. Another method may involve incorporating regularization terms to ensure that the energy function remains well-structured and meaningful across the entire input space.

(g) (2pts) Briefly explain the three methods that can be used to shape the energy function.

Answer for 1.1 (g):

1. **Contrastive Methods**: These methods involve adjusting the energy values based on positive and negative examples. The energy for positive examples (correct outputs) is pushed down, making them more likely, while

the energy for negative examples (incorrect outputs) is pushed up, making them less likely. This separation helps shape the energy landscape such that correct solutions have lower energy, and incorrect ones have higher energy, ensuring the model effectively distinguishes between them.

2. **Regularization Methods**: Regularization techniques add extra terms in the loss function to prevent a collapse condition. As explained in lecture, we can use a regularization term that measures the volume of space that has low energy: Sparse coding, sparse auto-encoder, LISTA, Variational Auto-Encoders, discretization/VQ/VQVAE. A simple example of **Sparse Coding** can be to encourage sparsity in the hidden layers, forcing the model to use fewer features, thus avoiding uniformly low energy across all inputs. This sparsity can be introduced by trivial methods like least absolute shrinkage and selection operator (LASSO) or via more complex methods.

3. **Architectural Methods**: These methods involve designing the model architecture in a way that directly limits the capacity to assign low energy to a large number of inputs. For instance, using very **small hidden layers** which restrict the representation space. Because the hidden representation is constrained, the model cannot perfectly represent all data points, which prevents it from assigning low energy everywhere and encourages it to prioritize lower energy for the most plausible configurations only.

(h) (2pts) Provide an example of a loss function that uses negative examples. The format should be as follows $\ell_{\text{example}}(x, y, W) = F_W(x, y)$.

Answer for 1.1 (h):

One example of a loss function that incorporates negative examples is the negative log-likelihood loss, which helps to separate correct predictions from incorrect ones by **penalizing high energy values for positive** examples and adjusting for negative examples. It can be expressed as:

$$\ell_{\text{example}}(x, y, W) = F_W(x, y) + \frac{1}{\beta} \log \left[\int_{y'} \exp(-\beta F_W(x, y')) \right]$$

In this formulation, $F_W(x, y)$ represents the energy associated with the correct input-output pair (x, y), and the second term accounts for all possible outputs y', effectively serving as a normalization factor (often referred to as the partition function). The term $\exp(-\beta F_W(x, y'))$ assigns a probability-like score to each alternative output based on its energy value. By taking the log of the sum of these scores, we incorporate the influence of negative examples into the loss function, encouraging the model to not only lower the energy for correct examples but also increase it for less likely outputs. The parameter β controls the sharpness of this adjustment, influencing how strictly the model separates correct and incorrect predictions.

(i) (2pts) Say we have an energy function $F(x, y)$ with images x, classification for this image y. Write down the mathematical expression for doing infer-

ence given an input x. Now say we have a latent variable z, and our energy is $G(x, y, z)$. What is the expression for doing inference then?

Answer for 1.1 (i):

In the context of inference using an EBM, we often aim to find the output y that minimizes the energy function, which corresponds to finding the most likely prediction given the input x. We consider both cases i.e. with and without latent variables below.

1. **Inference with an Energy Function** $F(x, y)$:

When we have an energy function $F(x, y)$, the goal is to find the value of y that minimizes the energy for a given input x. Thus the equation is simply:

$$\hat{y} = \arg\min_{y} F(x, y)$$

Here, $\hat{y}$ represents the predicted output that minimizes the energy function, indicating the most likely classification for the input image x.

2. **Inference with an Energy Function** $G(x, y, z)$ involving a **Latent** z:

When the energy function depends on a latent variable z, the inference process involves finding the output y that minimizes the marginalized energy, which accounts for all possible values of z. The equation thus becomes:

$$\hat{y} = \arg\min_{y} \left(\int_{z} G(x, y, z) \right)$$

In this case, we marginalize (integrate) the energy function over all possible configurations of the latent variable z to obtain the effective energy for each y, and then find the y that minimizes this marginalized energy.

1.2 Negative log-likelihood loss (20 pts)

Let's consider an energy-based model we are training to do classification of input between n classes. $F_W(x, y)$ is the energy of input x and class y. We consider n classes: $y \in \{1, \ldots, n\}$.

(i) (2pts) For a given input x, write down an expression for a Gibbs distribution over labels $p(y|x)$ that this energy-based model specifies. Use β for the constant multiplier.

Answer for 1.2 (i):

$$p(y \mid x) = \frac{\exp(-\beta F_W(x, y))}{\int_{y'} \exp(-\beta F_W(x, y'))}$$

(ii) (5pts) Let's say for a particular data sample x, we have the label y. Give the expression for the negative log likelihood loss, i.e. negative log likelihood of the correct label (show step-by-step derivation of the loss function from the expression of the previous subproblem). For easier calculations in the following subproblem, multiply the loss by $\frac{1}{\beta}$.

Answer for 1.2 (ii):

We Know, the Negative Log Likelihood for Label y, given the Input x,

$$\ell(x, y, W) = -\log p(y \mid x)$$

Substituting the Expression for $p(y \mid x)$ from 1.2 (i),

$$\ell(x, y, W) = -\log\left(\frac{\exp(-\beta F_W(x, y))}{\int_{y'} \exp(-\beta F_W(x, y'))}\right)$$

Using $\log \frac{a}{b} = \log a - \log b$ Property,

$$\ell(x, y, W) = -\left[\log\left(\exp(-\beta F_W(x, y))\right) - \log\left(\int_{y'} \exp(-\beta F_W(x, y'))\right)\right]$$

Assuming log has base e, We Can Simplify Above,

$$\ell(x, y, W) = \beta F_W(x, y) + \log\left(\int_{y'} \exp(-\beta F_W(x, y'))\right)$$

Multiplying the Loss by $\frac{1}{\beta}$ for Easier Calculations,

$$\ell(x, y, W) = \frac{1}{\beta}\left[\beta F_W(x, y) + \log\left(\int_{y'} \exp(-\beta F_W(x, y'))\right)\right]$$

Thus, the Final Expression of Loss is,

$$\ell(x, y, W) = F_W(x, y) + \frac{1}{\beta}\log\left(\int_{y'} \exp(-\beta F_W(x, y'))\right)$$

(iii) (8pts) Now, derive the gradient of that expression with respect to W (just providing the final expression is not enough). Your final answer may contain the expression $\frac{\partial F_W(\ldots)}{\partial W}$. Why can it be intractable to compute it, and how can we get around the intractability?

Answer for 1.2 (iii):

From 1.2 (ii), We Know, the Negative Log Likelihood Loss,

$$\ell(x, y, W) = F_W(x, y) + \frac{1}{\beta}\log\left(\int_{y'} \exp(-\beta F_W(x, y'))\right)$$

Differentiating the Loss Function w.r.t. W,

$$\frac{\partial \ell(x,y,W)}{\partial W} = \frac{\partial F_W(x,y)}{\partial W} + \frac{1}{\beta}\frac{\partial}{\partial W}\left(\log\left(\int_{y'} \exp(-\beta F_W(x,y'))\right)\right)$$

Differentiating the Second Term using Chain Rule,

$$\frac{\partial}{\partial W}\left(\log\left(\int_{y'} \exp(-\beta F_W(x,y'))\right)\right) = \frac{1}{\int_{y'} \exp(-\beta F_W(x,y'))} \cdot \frac{\partial}{\partial W}\left(\int_{y'} \exp(-\beta F_W(x,y'))\right)$$

Differentiating the Integral Component using Chain Rule,

$$\frac{\partial}{\partial W}\left(\int_{y'} \exp(-\beta F_W(x,y'))\right) = \int_{y'} \frac{\partial}{\partial W}\left(\exp(-\beta F_W(x,y'))\right)$$

$$= \int_{y'} \exp(-\beta F_W(x,y'))\left(-\beta \frac{\partial F_W(x,y')}{\partial W}\right)$$

Substituting Intermediate Results,

$$\frac{\partial \ell(x,y,W)}{\partial W} = \frac{\partial F_W(x,y)}{\partial W} + \frac{1}{\beta} \cdot \frac{1}{\int_{y'} \exp(-\beta F_W(x,y'))} \int_{y'} \exp(-\beta F_W(x,y'))\left(-\beta \frac{\partial F_W(x,y')}{\partial W}\right)$$

On Simplification,

$$\frac{\partial \ell(x,y,W)}{\partial W} = \frac{\partial F_W(x,y)}{\partial W} - \int_{y'} P_W(y' \mid x) \frac{\partial F_W(x,y')}{\partial W}$$

Here, $P_W(y' \mid x)$ represents the Gibbs-Boltzmann Distribution:

$$P_W(y' \mid x) = \frac{\exp(-\beta F_W(x,y'))}{\int_{y''} \exp(-\beta F_W(x,y''))}$$

Case for Intractability of Computation: The integral over y' can be computationally intractable when the number of possible values for y' is very large, especially in continuous or high-dimensional spaces.

Addressing Intractability: One way to get around this issue is by using sampling methods like Markov Chain Monte Carlo (MCMC), which can approximate the integral by sampling from the distribution $P_W(y' \mid x)$. This allows us to estimate gradient without computing full integral explicitly.
Reference: https://en.wikipedia.org/wiki/Markov_chain_Monte_Carlo

(iv) (5pts) Explain why negative log-likelihood loss pushes the energy of the correct example to negative infinity, and all others to positive infinity, no matter how close the two examples are, resulting in an energy surface with really sharp edges in case of continuous y (this is usually not an issue for discrete y because there's no distance measure between different classes).

Answer for 1.2 (iv):

The negative log-likelihood loss forces the model to minimize the energy for the correct example while increasing the energy for all other outputs. The way it pushes the energy up for incorrect examples is **independent of the distance** from the correct output; it is based solely on the probabilities assigned to the different outputs. As a result, even if two outputs are very close to each other in the continuous y case, the model will still push up the energy for the incorrect output as much as it does for an output that is far from the correct value. This behavior leads to a very steep and abrupt change in the energy surface, creating sharp edges or discontinuities.

1.3 Comparing Contrastive Loss Functions (15pts)

In this problem, we're going to compare a few contrastive loss functions. We are going to look at the behavior of the gradients, and understand what uses each loss function has. In the following subproblems, m is a margin, $m \in \mathbb{R}$, x is input, y is the correct label, $\bar{y}$ is the incorrect label. Define the loss in the following format: $\ell_{example}(x, y, \bar{y}, W) = F_W(x, y)$.

(a) (2pts) **Simple loss function** is defined as follows:

$$\ell_{\text{simple}}(x, y, \bar{y}, W) = [F_W(x, y)]^+ + [m - F_W(x, \bar{y})]^+$$

Where $[z]^+ = max(0, z)$

Assuming we know the derivative $\frac{\partial F_W(x,y)}{\partial W}$ for any x, y, give an expression for the partial derivative of the ℓ_{simple} with respect to W.

Answer for 1.3 (a):

$$\ell_{\text{simple}}(x, y, \bar{y}, W) = [F_W(x, y)]^+ + [m - F_W(x, \bar{y})]^+$$

Computing Partial Derivative of Simple Loss Function w.r.t. W,

$$\frac{\partial \ell_{\text{simple}}}{\partial W} = \frac{\partial [F_W(x, y)]^+}{\partial W} + \frac{\partial [m - F_W(x, \bar{y})]^+}{\partial W}$$

Partial Derivative of $[F_W(x, y)]^+$ w.r.t. W is,

$$\frac{\partial [F_W(x, y)]^+}{\partial W} = \begin{cases} 0, & \text{if } F_W(x, y) < 0 \\ \frac{\partial F_W(x,y)}{\partial W}, & \text{in all other cases} \end{cases}$$

Partial Derivative of $[m - F_W(x, \bar{y})]^+$ w.r.t. W is,

$$\frac{\partial [m - F_W(x, \bar{y})]^+}{\partial W} = \begin{cases} 0, & \text{if } F_W(x, \bar{y}) > m \\ -\frac{\partial F_W(x, \bar{y})}{\partial W}, & \text{in all other cases} \end{cases}$$

(b) (2pts) **Hinge Loss** is defined as follows:

$$\ell_{\text{hinge}}(x, y, \bar{y}, W) = [m + F_W(x, y) - F_W(x, \bar{y})]^+$$

Assuming we know the derivative $\frac{\partial F_W(x,y)}{\partial W}$ for any x, y, give an expression for the partial derivative of the ℓ_{hinge} with respect to W.

Answer for 1.3 (b):

$$\ell_{\text{hinge}}(x, y, \bar{y}, W) = [m + F_W(x, y) - F_W(x, \bar{y})]^+$$

Partial Derivative of the Hinge Loss Function w.r.t. W is,

$$\frac{\partial \ell_{\text{hinge}}}{\partial W} = \frac{\partial [m + F_W(x, y) - F_W(x, \bar{y})]^+}{\partial W}$$

The partial derivative of $[m + F_W(x, y) - F_W(x, \bar{y})]^+$ thus depends on whether $m + F_W(x, y) - F_W(x, \bar{y})$ is greater than zero or not. Thus, the partial derivative of hinge loss with respect to W is determined based on the condition whether the difference between $F_W(x, \bar{y})$ and $F_W(x, y)$ exceeds margin m.

$$\frac{\partial \ell_{\text{hinge}}}{\partial W} = \begin{cases} 0, & \text{if } F_W(x, \bar{y}) - F_W(x, y) > m \\ \frac{\partial F_W(x,y)}{\partial W} - \frac{\partial F_W(x,\bar{y})}{\partial W}, & \text{in all other cases} \end{cases}$$

(c) (2pts) **Log loss** is defined as follows:

$$\ell_{\log}(x, y, \bar{y}, W) = \log\left(1 + e^{F_W(x,y) - F_W(x,\bar{y})}\right)$$

Assuming we know the derivative $\frac{\partial F_W(x,y)}{\partial W}$ for any x, y, give an expression for the partial derivative of the $\ell_{\log}$ with respect to W.

Answer for 1.3 (c):

We Know Equation for Log Loss,

$$\ell_{\log}(x, y, \bar{y}, W) = \log\left(1 + e^{F_W(x,y) - F_W(x,\bar{y})}\right)$$

Computing Partial Derivative $\ell_{\log}$ w.r.t. W,

$$\frac{\partial \ell_{\log}}{\partial W} = \frac{\partial}{\partial W}\left(\log\left(1 + e^{F_W(x,y) - F_W(x,\bar{y})}\right)\right)$$

Using Chain Rule,

$$\frac{\partial \ell_{\log}}{\partial W} = \frac{1}{1+e^{F_W(x,y)-F_W(x,\bar{y})}} \cdot \frac{\partial}{\partial W}\left(e^{F_W(x,y)-F_W(x,\bar{y})}\right)$$

Computing Partial Derivative of Exponential Term $e^{F_W(x,y)-F_W(x,\bar{y})}$ w.r.t. W,

$$\frac{\partial}{\partial W}\left(e^{F_W(x,y)-F_W(x,\bar{y})}\right) = e^{F_W(x,y)-F_W(x,\bar{y})}\left(\frac{\partial F_W(x,y)}{\partial W} - \frac{\partial F_W(x,\bar{y})}{\partial W}\right)$$

Substituting Intermediate Results,

$$\frac{\partial \ell_{\log}}{\partial W} = \frac{e^{F_W(x,y)-F_W(x,\bar{y})}}{1+e^{F_W(x,y)-F_W(x,\bar{y})}}\left(\frac{\partial F_W(x,y)}{\partial W} - \frac{\partial F_W(x,\bar{y})}{\partial W}\right)$$

Also,

$$\frac{e^{F_W(x,y)-F_W(x,\bar{y})}}{1+e^{F_W(x,y)-F_W(x,\bar{y})}} = \sigma(F_W(x,y)-F_W(x,\bar{y}))$$

Characteristics: The aforementioned term behaves like a sigmoid function. It **smoothly maps** the difference $F_W(x,y)-F_W(x,\bar{y})$ to values between 0 and 1. When $F_W(x,y)$ is much larger than $F_W(x,\bar{y})$, the term approaches 1, meaning the gradient primarily depends on the difference $\frac{\partial F_W(x,y)}{\partial W} - \frac{\partial F_W(x,\bar{y})}{\partial W}$. Conversely, if $F_W(x,\bar{y})$ is much larger than $F_W(x,y)$, the term approaches 0, making the gradient close to zero. Unlike hinge loss, the log loss provides a smoother transition because the gradient changes gradually as $F_W(x,y)$ and $F_W(x,\bar{y})$ vary. This leads to a **smoother energy landscape**, avoiding sharp discontinuities or steep edges. If $F_W(x,y)$ and $F_W(x,\bar{y})$ are close, the gradient will still produce non-zero updates because the sigmoid term will be around 0.5. This is different from hinge loss, where if the margin is not exceeded, the gradient can be 0, leading to no updates.

(d) (2pts) **Square-Square loss** is defined as follows:

$$\ell_{\text{square-square}}(x,y,\bar{y},W) = \left([F_W(x,y)]^+\right)^2 + \left([m-F_W(x,\bar{y})]^+\right)^2$$

Assuming we know the derivative $\frac{\partial F_W(x,y)}{\partial W}$ for any x,y, give an expression for the partial derivative of the $\ell_{\text{square-square}}$ with respect to W.

Answer for 1.3 (d):

$$\ell_{\text{square-square}}(x,y,\bar{y},W) = \left([F_W(x,y)]^+\right)^2 + \left([m-F_W(x,\bar{y})]^+\right)^2$$

Computing Partial Derivative of $\ell_{\text{square-square}}$ w.r.t. W,

$$\frac{\partial \ell_{\text{square-square}}}{\partial W} = \frac{\partial \left([F_W(x,y)]^+\right)^2}{\partial W} + \frac{\partial \left([m-F_W(x,\bar{y})]^+\right)^2}{\partial W}$$

Computing Partial Derivative of $\left([F_W(x,y)]^+\right)^2$ w.r.t. W,

$$\frac{\partial\left([F_W(x,y)]^+\right)^2}{\partial W} = \begin{cases} 0, & \text{if } F_W(x,y) < 0 \\ 2F_W(x,y)\frac{\partial F_W(x,y)}{\partial W}, & \text{in all other cases} \end{cases}$$

Computing Partial Derivative of $\left([m - F_W(x,\bar{y})]^+\right)^2$ w.r.t. W,

$$\frac{\partial\left([m - F_W(x,\bar{y})]^+\right)^2}{\partial W} = \begin{cases} 0, & \text{if } F_W(x,\bar{y}) > m \\ -2(m - F_W(x,\bar{y}))\frac{\partial F_W(x,\bar{y})}{\partial W}, & \text{in all other cases} \end{cases}$$

(e) (7pts) **Comparison.**

(i) (2pts) Explain how NLL loss is different from the three losses above.

(ii) (2pts) The hinge loss $[F_W(x,y) - F_W(x,\bar{y}) + m]^+$ has a margin parameter m, which gives 0 loss when the positive and negative examples have energy that are m apart. The log loss is sometimes called a "soft-hinge" loss. Why? What is the advantage of using a soft hinge loss?

(iii) (2pts) How are the simple loss and square-square loss different from the hinge/log loss?

(iv) (1pt) In what situations would you use the simple loss, and in what situations would you use the square-square loss?

Answer for 1.3 (e) (i):

The Negative Log-Likelihood (NLL) loss fundamentally differs from the losses discussed above in its approach to penalizing incorrect predictions. While the simple, hinge, square-square, and log losses typically involve comparing a correct label with a single incorrect label, NLL considers all possible incorrect labels simultaneously. It aims to increase the energy for all incorrect examples while lowering the energy for the correct example. This characteristic makes NLL loss more computationally intensive because it requires calculating the normalization factor over all possible output labels, often through an integral or summation. In cases where the output space is large or continuous, this integral can be intractable, posing challenges for direct computation. In contrast, the other loss functions focus on simpler pairwise comparisons, which do not demand heavy compute.

Answer for 1.3 (e) (ii):

Role of Margin in Hinge Loss: In hinge loss, the margin parameter m defines a threshold for how far apart the energies of the positive example $F_W(x,y)$ and the negative example $F_W(x,\bar{y})$ should be. When the difference between these energies is at least m, the loss becomes zero, indicating that the separation is sufficient. The margin ensures that not only the positive example has lower energy than the negative, but also by a significant

amount. This adds a level of robustness to the classification, as it avoids decision boundaries that are too close to the examples.

Log Loss as "Soft Hinge" Loss & Advantages: The log loss is sometimes referred to as a "soft hinge" loss because, unlike the standard hinge loss, it doesn't abruptly drop to zero when the margin is exceeded. Instead, it decreases smoothly, gradually reducing the loss as the margin increases. The "soft" nature of log loss helps with optimization because it provides a smoother gradient, avoiding sharp transitions. This makes the training process more stable, particularly for gradient-based methods. We also refer to this in 1.3 (c). Using a soft hinge loss like log loss allows for better handling of cases where the positive and negative examples are close in terms of energy. Even if the margin is exceeded, the log loss still contributes to weight updates, encouraging further separation. The smoothness of the log loss thus leads to more gradual and consistent updates to the model parameters, helping to avoid oscillations or sudden changes during training.

Answer for 1.3 (e) (iii):

Hinge and log losses focus on enforcing a margin between the energies of the correct and incorrect examples. For hinge loss, the goal is to ensure that the difference between the correct example's energy and the incorrect example's energy is at least m. If this condition is met, the loss is zero. Log loss, acting as a "soft hinge" loss, has a smoother gradient and continues to penalize even when the margin is exceeded, offering a more gradual decrease in loss. Simple loss and square-square loss, unlike hinge or log loss, focus on directly pushing the energy of the correct example down and the energy of the incorrect example up. They do not rely on satisfying a margin condition but aim for a direct separation. Simple loss pushes linearly, meaning the updates to the weights are proportional to the magnitude of the energy difference. In contrast, square-square loss applies a quadratic penalty, leading to larger updates for larger differences, similar to L2 & L1.

Answer for 1.3 (e) (iv):

We use Simple Loss when we want to avoid large penalties for high-energy discrepancy points. Simple loss, with its linear behavior, does not disproportionately penalize larger errors compared to smaller ones. We can also use Simple Loss if we prefer a straightforward and less computationally demanding loss function. We use Square-Square Loss when we want to heavily penalize larger errors. Since square-square loss increases quadratically, it is suitable when we need a stronger push to separate examples with higher discrepancies. Similarly, we can also use this if we can afford the potentially higher sensitivity to extremes, as this approach emphasizes correcting larger deviations. The choice between these losses often mirrors the trade-off between L1 and L2 regularization, where L1 (simple loss) is more robust to high discrepancy, while L2 (square-square loss) is commonly used due to its smoothness and strong penalization of larger errors.