Homework 5 Solutions

1. **Regularized Linear Regression**

(a) Find an expression for the weight w which minimizes the L_2-regularized loss:

$$\mathcal{E}_{L_2} = \mathcal{E} + \frac{\lambda}{2} w^2$$

We have to minimize $\mathcal{E}_{L_2}$ with respect to w; that is, we have to find the value of w which yields the minimal value of $\mathcal{E}_{L_2}$. To do this, we solve $\frac{\partial \mathcal{E}_{L_2}}{\partial w} = 0$:

$$\mathcal{E}_{L_2} = \mathcal{E} + \frac{\lambda}{2} w^2$$

$$\mathcal{E}_{L_2} = \frac{1}{2N} \left[\sum_{i=1}^{N} (wx^{(i)} - t^{(t)})^2 \right] + \frac{\lambda}{2} w^2$$

$$\frac{\partial \mathcal{E}_{L_2}}{\partial w} = \frac{1}{2N} \left[\sum_{i=1}^{N} \frac{\partial}{\partial w} (wx^{(i)} - t^{(t)})^2 \right] + \frac{\partial}{\partial w} \frac{\lambda}{2} w^2$$

Now, for each $i = 1, \ldots, N$, we have:

$$\begin{aligned} \frac{\partial}{\partial w} (wx^{(i)} - t^{(t)})^2 &= 2(wx^{(i)} - t^{(i)}) \frac{\partial}{\partial w} (wx^{(i)} - t^{(i)}) \\ &= 2x^{(i)} (wx^{(i)} - t^{(i)}) \end{aligned}$$

For the regularization term, we find:

$$\frac{\partial}{\partial w} (\frac{\lambda}{2} w^2) = \lambda w$$

Putting these two parts together, we have:

$$\frac{\partial \mathcal{E}_{L_2}}{\partial w} = \frac{1}{2N} \sum_{i=1}^{N} 2x^{(i)} (wx^{(i)} - t^{(i)}) + \lambda w$$

To find the value(s) of w that yield minimal values of $\mathcal{E}_{L_2}$, we just set this partial derivative to 0, and solve for w:

$$\frac{1}{2N} \sum_{i=1}^{N} 2x^{(i)} (wx^{(i)} - t^{(i)}) + \lambda w = 0$$

$$\frac{1}{N} \sum_{i=1}^{N} (w(x^{(i)})^2 - t^{(i)} x^{(i)}) + \lambda w = 0$$

Breaking apart the sum:

$$\frac{1}{N} \sum_{i=1}^{N} (w(x^{(i)})^2) - \frac{1}{N} \sum_{i=1}^{N} (t^{(i)} x^{(i)}) + \lambda w = 0$$

Pulling w out of the first sum, and factoring it out of the regularization term:

$$w\left[\frac{1}{N}\sum_{i=1}^{N}[(x^{(i)})^2]+\lambda\right]-\frac{1}{N}\sum_{i=1}^{N}(t^{(i)}x^{(i)})=0$$

Now, we rearrange terms to solve for w:

$$w=\frac{\frac{1}{N}\sum_{i=1}^{N}(t^{(i)}x^{(i)})}{\frac{1}{N}\sum_{i=1}^{N}[(x^{(i)})^2]+\lambda}$$

(b) Find an expression for the weight w which minimizes the L_1-regularized loss:

$$\mathcal{E}_{L_1}=\mathcal{E}+\lambda|w|$$

Note that $\mathcal{E}_{L_1}$is convex, so there is a unique minimum. This minimum can be either at a critical point (derivative exists and equals zero) or at the point where $\mathcal{E}_{L_1}$ is non-differentiable (i.e. 0).

$$\mathcal{E}_{L_1}=\frac{1}{2N}\left[\sum_{i=1}^{N}(wx^{(i)}-t^{(t)})^2\right]+\lambda|w|$$

Taking the gradient with respect to w, we have:

$$\mathcal{E}_{L_1}=\frac{1}{2N}\left[\sum_{i=1}^{N}2x^{(i)}(wx^{(i)}-t^{(t)})\right]+\lambda\frac{\partial}{\partial w}|w|$$

Now, the derivative of $|w|$ with respect to w is defined piecewise:

$$\frac{\partial}{\partial w}|w|=\begin{cases}1 & \text{if } w>0\\ -1 & \text{if } w<0\end{cases}$$

Note that $|w|$ is *not differentiable* at $w=0$.

If w > 0:

$$\frac{\partial}{\partial w}\mathcal{E}_{L_1}=\frac{1}{2N}\left[\sum_{i=1}^{N}2x^{(i)}(wx^{(i)}-t^{(t)})\right]+\lambda=0$$

$$\frac{1}{N}\left[\sum_{i=1}^{N}[w(x^{(i)})^2-x^{(i)}t^{(t)}]\right]+\lambda=0$$

Rearranging, we solve for w:

$$w=\frac{\frac{1}{N}\sum_{i=1}^{N}x^{(i)}t^{(t)}-\lambda}{\frac{1}{N}\sum_{i=1}^{N}(x^{(i)})^2}$$

Remember that we assumed $w>0$, so we must check if this formula evaluates to something positive in order for it to be a valid critical point. We get $w>0$ if:

$$\frac{1}{N}\sum_{i=1}^{N}x^{(i)}t^{(t)}>\lambda.$$

If w < 0:

$$\frac{\partial}{\partial w}\mathcal{E}_{L_1} = \frac{1}{2N}\left[\sum_{i=1}^{N} 2x^{(i)}(wx^{(i)} - t^{(t)})\right] - \lambda = 0$$

$$\frac{1}{N}\left[\sum_{i=1}^{N}[w(x^{(i)})^2 - x^{(i)}t^{(t)}]\right] - \lambda = 0$$

Rearranging, we solve for w:

$$w = \frac{\frac{1}{N}\sum_{i=1}^{N} x^{(i)}t^{(t)} + \lambda}{\frac{1}{N}\sum_{i=1}^{N}(x^{(i)})^2}$$

For this to be a valid critical point, we need $w < 0$. This is true if:

$$\frac{1}{N}\sum_{i=1}^{N} x^{(i)}t^{(t)} < -\lambda.$$

In all other cases, the minimum must occur at $w = 0$. Putting this all together,

$$\begin{cases} \frac{\frac{1}{N}\sum_{i=1}^{N} x^{(i)}t^{(t)} - \lambda}{\frac{1}{N}\sum_{i=1}^{N}(x^{(i)})^2} & \text{if } \frac{1}{N}\sum_{i=1}^{N} x^{(i)}t^{(t)} > \lambda \\ 0 & \text{if } -\lambda \leq \frac{1}{N}\sum_{i=1}^{N} x^{(i)}t^{(t)} \leq \lambda \\ \frac{\frac{1}{N}\sum_{i=1}^{N} x^{(i)}t^{(t)} + \lambda}{\frac{1}{N}\sum_{i=1}^{N}(x^{(i)})^2} & \text{if } \frac{1}{N}\sum_{i=1}^{N} x^{(i)}t^{(t)} < -\lambda \end{cases}$$

2. **Dropout**

(a) Find expressions for $\mathbb{E}[y]$ and $\mathrm{Var}[y]$ for a given data point.
We can determine $\mathbb{E}[y]$ and $\mathrm{Var}[y]$ using the properties of expectation and variance.

$$\begin{aligned} \mathbb{E}[y] &= \mathbb{E}\left[\sum_j m_j w_j x_j\right] \\ &= \sum_j w_j x_j \mathbb{E}[m_j] && \text{by linearity of expectation} \\ &= \frac{1}{2}\sum_j w_j x_j && \text{by the expectation formula for a Bernoulli r.v.} \\ \mathrm{Var}[y] &= \mathrm{Var}\left[\sum_j m_j w_j x_j\right] \\ &= \sum_j \mathrm{Var}\left[m_j w_j x_j\right] && \text{by independence} \\ &= \sum_j w_j^2 x_j^2 \mathrm{Var}[m_j] && \text{by the scalar multiplication rule for variance} \\ &= \frac{1}{4}\sum_j w_j^2 x_j^2 && \text{by the variance formula for a Bernoulli r.v.} \end{aligned}$$

(b) Determine $\tilde{w}_j$ as a function of w_j such that

$$\mathbb{E}[y] = \tilde{y} = \sum_j \tilde{w}_j x_j$$

Based on the expectation derived in Part (a), we have:

$$\begin{aligned}\mathbb{E}[y] &= \frac{1}{2}\sum_j w_j x_j^{(i)} \\ &= \sum_j (\frac{1}{2}w_j) x_j^{(i)}\end{aligned}$$

Thus,

$$\tilde{w}_j = \frac{1}{2}w_j$$

(c) Using the model from the previous section, show that the cost $\mathcal{E}$ can be written as:

$$\mathcal{E} = \frac{1}{2N}\sum_{i=1}^{N}(\tilde{y}^{(i)} - t^{(i)})^2 + \mathcal{R}(\tilde{w}_1, \ldots, \tilde{w}_D)$$

Equation 1 in the homework states:

$$\mathcal{E} = \frac{1}{2N}\sum_{i=1}^{N}\mathbb{E}[(y^{(i)} - t^{(i)})^2]$$

Using the fact that the expectation is a linear operation, we can expand it as follows:

$$\mathbb{E}[(y^{(i)} - t^{(i)})^2] = \mathbb{E}[(y^{(i)})^2] - 2\mathbb{E}[y^{(i)}t^{(i)}] + \mathbb{E}[(t^{(i)})^2]$$

We can express $\mathbb{E}[(y^{(i)})^2]$ in terms of the variance as follows:

$$\mathbb{E}[(y^{(i)})^2] = \mathrm{Var}[y^{(i)}] + \mathbb{E}[y^{(i)}]^2$$

Since $\tilde{y}^{(i)} = \mathbb{E}[y^{(i)}]$, we have:

$$\mathbb{E}[(y^{(i)})^2] = \mathrm{Var}[y^{(i)}] + (\tilde{y}^{(i)})^2$$

Since $t^{(i)}$ is not a function of the $m_j^{(i)}$'s, $t^{(i)}$ is treated as a constant in the expectation $\mathbb{E}[y^{(i)}t^{(i)}]$, so we have:

$$\begin{aligned}\mathbb{E}[y^{(i)}t^{(i)}] &= t^{(i)}\mathbb{E}[y^{(i)}] \\ &= t^{(i)}\tilde{y}^{(i)}\end{aligned}$$

Similarly, since $t^{(i)}$ is not a function of the $m_j^{(i)}$'s, the expectation of $(t^{(i)})^2$ with respect to the $m_j^{(i)}$'s is $(t^{(i)})^2$:

$$\mathbb{E}[(t^{(i)})^2] = (t^{(i)})^2$$

Putting these terms together, we have:

$$\begin{aligned}\mathbb{E}[(y^{(i)} - t^{(i)})^2] &= \mathrm{Var}[y^{(i)}] + (\tilde{y}^{(i)})^2 - 2t^{(i)}(\tilde{y}^{(i)})^2 + (t^{(i)})^2 \\ &= (\tilde{y}^{(i)} - t^{(i)})^2 + \mathrm{Var}[y^{(i)}]\end{aligned}$$

Plugging this derivation of $\mathbb{E}[(y^{(i)} - t^{(i)})^2]$ into the original expression for $\mathcal{E}$ yields:

$$\begin{aligned}\mathcal{E} &= \frac{1}{2N}\sum_{i=1}^{N}\left((\tilde{y}^{(i)} - t^{(i)})^2 + \mathrm{Var}[y^{(i)}]\right) \\ &= \frac{1}{2N}\sum_{i=1}^{N}(\tilde{y}^{(i)} - t^{(i)})^2 + \frac{1}{2N}\sum_{i=1}^{N}\mathrm{Var}[y^{(i)}]\end{aligned}$$

Finally, we can substitute the expression for the variance that we derived in Part (a) to obtain a regularization term that does not involve any expectations:

$$\begin{aligned}\mathcal{E} &= \frac{1}{2N}\sum_{i=1}^{N}(\tilde{y}^{(i)} - t^{(i)})^2 + \frac{1}{2N}\sum_{i=1}^{N}\frac{1}{4}\sum_{j} w_j^2 (x_j^{(i)})^2 \\ &= \frac{1}{2N}\sum_{i=1}^{N}(\tilde{y}^{(i)} - t^{(i)})^2 + \frac{1}{8N}\sum_{i=1}^{N}\sum_{j} w_j^2 (x_j^{(i)})^2\end{aligned}$$

3. **Neural Language Model**

(a) **What is the total number of trainable parameters in the model? Which layer has the largest number of trainable parameters?**

Let V be the vocabulary size (i.e. the number of words in the dictionary), D be the word embedding dimension, and H be the dimension of the hidden layer.

The `word_embedding_weights` matrix stores the vector representations of each word, and functions as a lookup table. Each row of the matrix is a D-dimensional embedding of one of the V words in the vocabulary; thus, this matrix has dimension $V \times D = 250 \times 16$, yielding 4,000 trainable parameters.

The `embed_to_hid_weights` matrix takes a vector representing the concatenation of the three word embeddings for the context words, and produces a vector of the same dimension as the `hidden_layer`. That is, the `embed_to_hid_weights` matrix takes a $(3 \cdot D)$-dimensional vector as input, and produces an H-dimensional vector. Thus, this matrix must have dimension $H \times (3 \cdot D) = 128 \times 48$, yielding 6,144 trainable parameters. The `hid_bias` vector must have the same dimension as the hidden layer, so it is $H \times 1 = 128 \times 1$, and has 128 trainable parameters.

The output layer is a softmax over the 250 words; that is, the layer outputs normalized probabilities for each word in the vocabulary. The "`word 4`" box in the diagram represents a vector of probabilities, which has dimension $V \times 1 = 250 \times 1$. The matrix `hid_to_output_weights` takes the 128-dimensional hidden representation and produces a 250-dimensional vector; it must therefore have dimension $V \times H = 250 \times 128$, yielding 32,000 trainable parameters. Finally, the `output_bias` vector must have the same dimension as the output layer, so it is $V \times 1 = 250 \times 1$, and has 250 trainable parameters.

The dimensions of each of the vectors and matrices used in this model are summarized in Table 1.

Table 1: Dimensions and Number of Parameters for each Matrix and Vector

Matrix/Vector	Dimension	# Parameters
`word_embedding_weights`	$V \times D = 250 \times 16$	4,000
`embed_to_hid_weights`	$H \times 3D = 128 \times 48$	6,144
`hid_bias`	$H \times 1 = 128 \times 1$	128
`hid_to_output_weights`	$V \times H = 250 \times 128$	32,000
`output_bias`	$V \times 1 = 250 \times 1$	250

In total, we have **42,522 trainable parameters**. The `hid_to_output_weights` layer has the largest number of trainable parameters.

(b) **How many add-multiply operations are needed to make predictions, assuming the first layer is implemented using a lookup table?**

A *dot product* (inner product) between two M-dimensional vectors requires M multiplications and $M-1$ additions. Given an $N \times M$ matrix A and an $M \times 1$ vector $\mathbf{x}$, the matrix-vector product $A\mathbf{x}$ involves N dot products (each one between a *row* of A and the vector $\mathbf{x}$); this amounts to NM multiplications and $N(M-1)$ additions. Also, clearly, adding two M-dimensional vectors requires M additions and no multiplications.

Since `embed_to_hid_weights` has dimension 128×48, matrix multiplication by `embed_to_hid_weights` requires $128 \cdot 48 = 6,144$ multiplications and $128 \cdot (48-1) = 6,016$ additions. Adding the `hid_bias` vector takes 128 additions, and no multiplications. Matrix multiplication by `hid_to_output_weights` requires $250 \cdot 128 = 32,000$ multiplications and $250 \cdot (128-1) = 31,750$ additions. Finally, adding the `output_bias` vector takes 250 additions and no multiplications.

Thus, we need $6,144 + 32,000 = \mathbf{38,144}$ **multiplications** and $6,106 + 128 + 31,750 + 250 = \mathbf{38,144}$ **additions** to make a prediction, for a total of **38,144 add-multiply operations**.