1st Iran AI Olympiad — Problems and Solutions

2nd Round — Part II (Descriptive Section)
Academic Year 1403–1404 / 2024–2025

Shayan Shahrabi
`shayanshahrab@gmail.com`

Last updated: 30 May 2025

This is a compilation containing problems and solutions from the Descriptive Round of the **1st IOAI (Iranian Olympiad in Artificial Intelligence)**. The ideas of the solution are a mix of my own work and solutions found by the community. However, all the writing and translation of the questions to English is maintained by me. I've tried to answer in a neat, easy to understand way, while mentioning some notes and/or resources for future reading for some of the questions.
Corrections and comments are welcome!

"That's one small step for [a] man, one giant leap for mankind."
— Neil Armstrong

Introduction

The second round of the 1st Iran AI Olympiad took place on April 17, 2025. This round consisted of two parts. The first part was a multiple-choice exam with 15 questions and a duration of 120 minutes. The second part, held two hours after the first, was a written section featuring 4 descriptive questions, with a time limit of 150 minutes.

Problems and Solutions

Problem 1

The Islamic Republic of Iran Broadcasting (IRIB) intends to use a neural network to classify movie clips based on their genres. The network is designed to assign multiple genre labels to each movie. For instance, movie a might be categorized as both *Comedy* and *Action*, while movie b might fall under *Crime*, *Mystery*, and *Contagion*. Assume there are K possible genres in total. IRIB has tasked Mohammad Javad with training this network, and he proceeds with the following approach:

- For each data record, a K-dimensional vector is used as the label. In this vector, if the movie belongs to the i-th genre, the i-th entry is set to 1 ($y_i = 1$); otherwise, it is set to 0 ($y_i = 0$).
- The neural network has an output layer with K units. The raw outputs of this layer are passed through a softmax function to produce a probability distribution over the genres. The softmax function is defined as:

$$p_i = \frac{e^{a_i}}{\sum_{j=1}^{K} e^{a_j}} \tag{1}$$

  where a_i denotes the activation (raw score) corresponding to the i-th genre, and p_i is the predicted probability for that genre.
- To train the network, the following loss function is used:

$$L = -\frac{1}{N}\sum_{s=1}^{N}\sum_{i=1}^{K} y_i^{(s)} \log(p_i^{(s)}) \tag{2}$$

  where N is the number of samples in the dataset. The term $y_i^{(s)}$ is the ground-truth label for the i-th genre in the s-th sample, and $p_i^{(s)}$ is the predicted probability for the same.

Based on the above setup, answer the following questions:

1. Suppose the dataset is very large, and the model has sufficient capacity to overfit the training data. If the number of training epochs is increased significantly, will the loss ont the training data approach zero? Justify your answer.
2. Assume a model is trained as described above. Show that if Movie A has more genre labels (i.e., belongs to more genres) than Movie B, then the loss for Movie A will be greater than that for Movie B.
3. Can you propose a better modeling approach than Mohammad Javad's for this problem? If so, describe your approach and explain how it addresses the limitations in his model.

Solution to Part 1

Well, this a tricky question to start with! Pay great attention that the loss on the **training data** (**not** the test data) is questioned. Here is the answer:

The training loss will **not** approach zero (YES! Even the **training loss** won't approach zero, let alone the **validation/test** data), even if the dataset is large and the model has enough capacity to overfit — and here is why:

There is a fundamental mismatch in the modeling approach. The network uses a **softmax activation** function (Equation (1)) in the output layer, which is suitable for **multi-class classification problems** where each instance belongs to **exactly one** class. (Here, a movie can belong to multiple genres, violating the fundamental assumption of the softmax function.) In such cases, softmax ensures the outputs form a probability distribution (i.e., they sum to 1), and the model can be trained to assign high probability to the correct label.

However, this problem is a **multi-label classification** problem — that is, each movie can belong to **multiple genres** simultaneously. In this case, using softmax creates **competition** between output neurons. **Assigning a high probability to one genre necessarily reduces the probabilities assigned to others, making it impossible to assign high confidence to multiple genres at once.**

As a result, even with enough training time and model capacity, the network **cannot** represent the true label distribution properly, and the loss will converge to a **non-zero** value.

Solution to Part 2

The loss function used is:

$$L = -\frac{1}{N}\sum_{s=1}^{N}\sum_{i=1}^{K} y_i^{(s)} \log(p_i^{(s)})$$

This loss adds up the **negative log-probabilities** of the predicted values for the true genres — that is, only the entries where $y_i^{(s)} = 1$ contribute to the loss for sample s.

Suppose Movie A belongs to m_1 genres and Movie B belongs to m_2 genres,such that $m_1 > m_2$.

Since the softmax function ensures that the total output probability sums to 1, the model must distribute that probability mass across all the predicted labels equally . If a movie has m correct genres, and the model is trying to treat them equally, it will assign each one about $\hat{p} = \frac{1}{m}$.

So, for Movie A, the predicted probability for each of the m_1 correct genres is roughly $\hat{p}_1 = \frac{1}{m_1}$, and for Movie B, it's $\hat{p}_2 = \frac{1}{m_2}$. The corresponding losses are:

$$L_1 = -m_1 \cdot \log\left(\frac{1}{m_1}\right) = m_1 \log(m_1)$$

$$L_2 = -m_2 \cdot \log\left(\frac{1}{m_2}\right) = m_2 \log(m_2)$$

Because $m_1 > m_2$ and the function $m \log m$ is **strictly increasing** for $m > 1$, it easily follows that $L_1 > L_2$.

This means that even though each individual genre's predicted probability becomes **smaller** as the number of true genres increases, the overall loss becomes **larger**, because the model gets penalized once for each correct label — and those penalties add up.

In summary, the loss scales non-linearly with the number of labels. So, even if the model performs equally well on each label, examples with more labels are still penalized more. This highlights a major weakness of using softmax for multi-label classification: not only does it make it difficult to predict multiple genres at once (softmax forces probabilities to compete as discussed in part 1), but it also distorts the loss landscape by unfairly favoring samples with fewer labels.

Solution to Part 3

A better modeling approach would be to use a **sigmoid activation** function in the output layer instead of softmax, and switch to a **binary cross-entropy loss function**.

Revised approach:

- Use a sigmoid function for each output unit:

$$p_i = \frac{1}{1+e^{-a_i}}, \quad \text{for } i = 1, \ldots, K$$

- Use binary cross-entropy loss:

$$L = -\frac{1}{N}\sum_{s=1}^{N}\sum_{i=1}^{K}\left[y_i^{(s)}\log(p_i^{(s)}) + (1-y_i^{(s)})\log(1-p_i^{(s)})\right]$$

This approach treats each genre as an **independent** binary classification problem, allowing the network to assign high confidence to multiple genres for the same movie. It correctly captures the multi-label nature of the task and avoids the normalization constraint imposed by softmax.

Additional Notes

- The original loss function used by Mohammad Javad (equation 2) is a variant of the **categorical cross-entropy loss**.

- Softmax is typically used in multi-class classification, where each input belongs **to exactly one class**. (Always remember the **competition** between outputs to obtain a higher value. This concept really helps to better understand the true role of a softmax function applied to outputs)

  I'll call it the **Fundamental Assumption of Softmax Activation Function** from now on.

- Multi-label problems should be modeled **independently**. Our approach in the 3rd part was to use sigmoid activations with binary cross-entropy losses. With sigmoid + binary cross-entropy, each genre prediction is independent, which better fits the real-world structure of genre classification.

- A **softmax** activation forces the output probabilities to sum to 1, making it unsuitable when **multiple labels** need to be predicted simultaneously.

- This question reminds me of the classic recommendation system example presented by Yaser Abu-Mostafa in the early sessions of his machine learning class at Caltech[a]. In that example, each movie was represented by a vector, and the task was to recommend movies to people by using the vector representation of each movie.

[a]The course webpage is accessible at work.caltech.edu/telecourse

Problem 2

A logistic regression model is able to present a linear decision boundary for the data points of Figure 1 as below:

$$w_2x_2 + w_1x_1 + w_0 = 0 \tag{3}$$

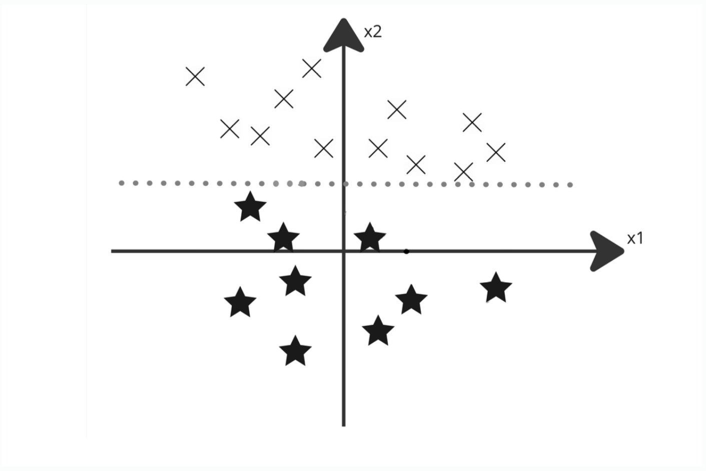


Figure 1: Data Points Related to the Linear Decision Making Question

As it's apparent by Figure 1, the data points are linearly separable. Suppose we call the loss function of this model $\mathcal{L}_0$. We define the new loss function $\mathcal{L}_i$ as following:

$$\mathcal{L}_i = \mathcal{L}_0 + \lambda w_i^2, \quad i \in \{0, 1, 2\} \tag{4}$$

where w_i are the model parameters and λ is a hyperparameter that can be chosen arbitrarily.
If λ is chosen to be very large, which of the loss functions allows the model to separate the two classes better? Explain your reasoning.

Solution

[Your solution to Problem 2 here]

Problem 3

Maryam is interested in converting decision trees used for classification problems into neural networks. For this, she uses networks of multi-layer perceptrons (MLPs) with a step activation function for the hidden units:

$$\sigma(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{5}$$

She also uses a softmax layer to produce the probability vector of the classifier. Finally, the model outputs the category with the highest probability.
A decision tree T has been given to Maryam. The nodes of this tree contain conditions of the form $x_i \geq C$ or $x_i \leq C$, where C is a constant specific to each node. This decision tree takes a four-dimensional real-valued input and classifies it into one of three categories.
Maryam has constructed a neural network with two hidden layers that mimics the behavior of this decision tree. The input to the neural network is the vector

$$x = [x_1, x_2, x_3, x_4]^T,$$

and the output is the probability vector

$$y = [y_1, y_2, y_3]^T,$$

which represents class probabilities.
The mathematical equations of this neural network are shown in Figure 2:

1. Assume that the decision tree T has no duplicate or redundant nodes. In this case, find T. A **duplicate node** means a condition is repeated multiple times in the tree — for example, $x_1 \geq 2$ or an equivalent appears multiple times. Conditions such as $x_1 \geq 1$ and $x_1 \geq 2$ are not duplicates.

   A **redundant node** is one whose output is always positive or always negative, and can therefore be pruned from the tree.

2. Maryam wants to use the ReLU activation function instead of σ in her neural network:

   $$\mathrm{ReLU}(x) = \max(0, x) \tag{6}$$

   Help Maryam design a neural network equivalent to the decision tree T. (Assume the input values x_i are integers.)

$$\mathbf{h}_1 = \sigma\left(\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 \\ -1 & 0 & 0 & 0 \end{bmatrix}\mathbf{x} + \begin{bmatrix} -5 \\ 4 \\ -7 \\ -1 \\ 3 \\ 10 \end{bmatrix}\right),$$

$$\mathbf{h}_2 = \sigma\left(\begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & -1 & 1 \\ 1 & 0 & 1 & 0 & -1 & -1 \\ 1 & 0 & -1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 1 & 0 & 0 \\ -1 & -1 & 0 & -1 & 0 & 0 \end{bmatrix}\mathbf{h}_1 + \begin{bmatrix} -3 \\ -3 \\ -2 \\ -1 \\ -1 \\ -1 \\ 0 \end{bmatrix}\right),$$

$$\mathbf{y} = \mathsf{softmax}\left(\begin{bmatrix} 10 & 0 & 0 & 0 & 10 & 0 & 10 \\ 0 & 0 & 10 & 0 & 0 & 10 & 0 \\ 0 & 10 & 0 & 10 & 0 & 0 & 0 \end{bmatrix}\mathbf{h}_2\right)$$

Figure 2: Mathematical Equations Used for the Neural Network

Solution to part 1

Solution to part 2

Problem 4

In the city of Nostradamia, three predictors with extraordinary abilities forecast important future events to whom the people of the city refer to for their predictions. Each predictor answer every question about the events that might happen in the furture in one of the 2 following ways:

$$\text{"Event will happen" (1),} \quad \text{"Event will not happen" (0).}$$

However, each predictor sometimes makes mistakes in certain time intervals and gives incorrect predictions! To reduce errors, the city council decides to use a majority vote rule: if at least two predictors agree on a prediction, that prediction is accepted as the final forecast.
The exact definition of the problem: We have three predictors M_1, M_2, and M_3, each making forecasts for each time frame in the future. These predictors want to predict the happening of a phenomenom in the time interval $[0, 1]$. The predictions are certain, meaning that if a question about an exact time frame is ask multiple times, the answers are the same. Each predictor makes mistakes in some specific time interval and always predicts wrong. These error intervals are defined as following:

- Predictor M_1 makes errors in interval $[I_1, I_2]$ of length $0.30 = I_2 - I_1$.
- Predictor M_2 makes errors in interval $[I_3, I_4]$ of length $0.35 = I_4 - I_3$.
- Predictor M_3 makes errors in interval $[I_5, I_6]$ of length $0.40 = I_6 - I_5$.

1. **Nightmare Scenario - Maximum Error Rate:** Choose the positions of I_1 to I_6 on the timeline so that the final majority vote produces the maximum possible total error. Find such an arrangement and prove this leads to the highest error rate.

2. **The Golden Scenario - Minimum Error Rate:** Choose the positions of I_1 to I_6 so that the final majority vote produces the minimum possible total error. Find such an arrangement and prove this leads to the lowest error rate.

Solution to Part 1

[Your solution to part 1 here]

Solution to Part 2

A naive strategy is to place the intervals as far apart as possible, thereby minimizing the length of the overall overlap interval. The key observation is that the sum of the lengths of the three intervals exceeds 1 (the length of the master interval $[0, 1]$, as stated in the problem).

For the sum of the intervals we have:

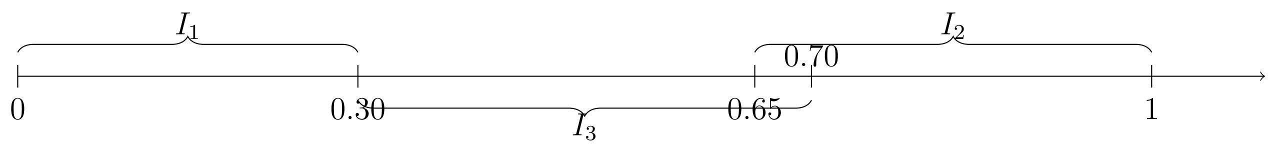
$$0.30 + 0.35 + 0.40 = 1.05$$

So, the **lower bound** for the overlap length is 0.05—there must be at least an overlap of size 0.05. But can we achieve this lower bound? Fortunately, the answer is yes! By providing a concrete example, we can show that this lower bound for the overlap of intervals is attainable, thereby proving that our solution is optimal.

Assume the intervals are placed as follows:

- $I_1 = 0, \quad I_2 = 0.30$
- $I_5 = 0.30, \quad I_6 = 0.70$
- $I_3 = 0.65, \quad I_4 = 1$

The overlap interval is only from point I_3 to I_5 with an exact length of 0.05. Thus, we reach the lower bound and solve the problem optimally!



Additional Notes

Well, to be honest, I don't have any clue what's the connection between this question and AI! Anyways, Here are some notes that come to my mind:

- Ensemble methods are useful methods! I believe one of the best resorces
- An example is mixture of experts