CAP 6640 – Natural Language Processing **Spring 2025**
Homework #2 Due date: January 31, 2025.

Problem 1:

Artificial Neural Networks, or ANNs in short, are a type of computational learning model that are inspired by how the human brain functions. ANNs are composed of computational unit of nodes named as neurons, organized into layers that process the input data and generate an output for tasks such as predictions and classifications. First component of the ANN is the input layer, which receives the input data related to the each word vector. Each neuron in this layer represents a feature of the input data. The hidden layers, which are located between the input and output layers, contain neurons that apply transformations to relationships by computing a weighted sum of input data and passing the result through an activation function to introduce non-linearity. By incorporating multiple hidden layers, the model can learn complex patterns in the data. As a last layer, the output layer generates the final output of the model, which can be used for making predictions or classifications. How ANNs do the learning process is by adjusting the weights of the connections between the nodes in the network to minimize the error between the predicted output and the actual output.

The incorporation of multiple layers within an ANN contributes to improved accuracy by allowing the model to learn complex representations in the data. Each hidden layer in the network learns different features of the data, and by stacking multiple layers, the model can learn hierarchical representations of the data. Each layer learns increasingly abstract representations. For example, in the context of natural language processing, the first hidden layer might learn simple features such as word frequencies, while the second hidden layer might learn more complex features such as word co-occurrences. By learning these hierarchical representations, the model can capture complex patterns in the data and make more accurate predictions or classifications. Furthermore, adding more layers with non-linear activation functions allows the model to capture highly complex relationships in the data that would be difficult to learn with a simple, and shallow network. In the context of language processing, this can help the model learn the relationships or dependencies between words and phrases. Lastly, instead of memorizing the raw data, the model can learn meaningful high-level features to generalize from the data, which can improve its performance on unseen data.

Problem 2:

1. **Summation-based representation**: For a single neuron in an Artificial Neural Network (ANN), the output of the neuron can be represented mathematically as:

$$z = \sum_{i=1}^{n} w_i x_i + b$$

where z is the weighted sum of the inputs features x_i with corresponding weights w_i, b is the bias term, and n is the number of input features used in this neuron calculation.

The output of the neuron is then passed through an activation function, which introduces non-linearity to the model:

$$a = f(z) = f(\sum_{i=1}^{n} w_i x_i + b)$$

where a is the final output of the neuron after passing through the activation function f.

If we apply this logic for a single layer with m neurons in an ANN, each neuron computes its output as:

$$z_j = \sum_{i=1}^{n} w_{ji} x_i + b_j \text{ for } j = 1, 2, ..., m$$

where z_j is the weighted sum of the inputs features x_i with corresponding weights w_{ji} and bias term b_j for the j-th neuron in the layer.

The output of the j-th neuron is then passed through an activation function:

$$a_j = f(z_j) = f(\sum_{i=1}^{n} w_{ji} x_i + b_j) \text{ for } j = 1, 2, ..., m$$

In summary, the summation-based representation expresses the ANN in terms of individual neurons and connections.

2. **Matrix-based representation**:

The above summation-based representation can be represented in matrix form for a layer of neurons in an ANN. Let's denote the input features as a vector X of size n.

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

The weights of the connections between the input features and the neurons in the layer can be represented as a weight matrix W of size $m \times n$, where m is the number of neurons in the layer.

$$W = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ w_{21} & w_{22} & \cdots & w_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m1} & w_{m2} & \cdots & w_{mn} \end{bmatrix}$$

The bias terms for each neuron can be represented as a bias vector B of size m.

$$B = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

By combining all these, the weighted sum of the inputs for all neurons in the layer can be computed as a matrix multiplication as

$$Z = W \cdot X + B$$

As the last step, by applying the activation function element-wise to the matrix Z, we can compute the output of all activated neurons in the layer as

$$A = f(Z) = f([z_1, z_2, \cdots, z_n]) = [f(z_1), f(z_2), \cdots, f(z_n)]$$

where A is the output of the layer after passing through the activation function f.

In summary, this matrix-based representation makes computations more efficient by leveraging vectorized operations.

Problem 3:

Named Entity Recognition, or NER in short, is the task of identifying and classifying named entities in text into predefined categories such as person names, organization names, locations, etc. However, NER faces several challenges that makes it a difficult task. Some of the challenges associated with named entities are:

1. **Ambiguity in Entities**: Named entities can be ambiguous, meaning that the same word can refer to different entities based on the context. For example, the word "Acibadem" in Turkish can refer to a dessert, a neighbourhood in Istanbul, or the healthcare group organization.
2. **Handling Unknown or Emerging Entities**: NER systems need to be able to handle unknown or emerging entities that are not present in the training data. For instance, probably a system trained on before 2020 data would not recognize the entity "COVID-19" as a disease and classify it as unknown word or a generic word.
3. **Ambiguity**: Named entities can be ambiguous, meaning that it is not obvious to predict whether it is an entity or just a general term used in the text. To give an example, Open AI, can be considered as an entity, but "open" and "AI" can be considered as separate words and can be interpreted as open-source AI.
4. **Complex Named Entities and the Boundries**: Named entities can be complex and composed of multiple words which makes it difficult to identify the boundaries of the entity. To give an instance, "Washington Post Sanctions" can be considered as a news organization entity, but also can be composed of multiple components such as administration in Washington D.C., etc.

To address these challenges, window classification technique can be used in NER. Window classification involves dividing the text into fixed-size windows to capture the context around each word. By considering the context around each word, the model can better understand the meaning of the word and disambiguate named entities.

To illustrate, consider the sentence "Jordan Thompson traveled to Jordan.". In this sentence, the word "Jordan" appears twice, but has different meanings. By using a window classification approach, the model can consider the context around each occurrence of "Jordan" to determine whether it refers to a person's name or a location. However, we might fail to capture the positional information and contextual nuances of the entities in the text if we use a very simple averaging method for the word vectors in the context window. To address this issue, we can use more sophisticated methods such as softmax classifier to calculate the probability of each word in the context window to be the entity of interest appearing together. By using window classification, the model can better handle ambiguity, unknown entities, and complex named entities by considering the context around each word.

Problem 4:

Write Answer

Problem 5:

Write Answer

Problem 6:

Write Answer

Problem 7:

Write Answer

Problem 8:

Write Answer

Problem 9:

Write Answer

Problem 10:

Write Answer