1. What advantages an RNN model of LM provides over an NN model? (2-4 sentences) [2 points]
2. Compare CBOW and Skip-gram models of word representation. (no more than half a page) [2 points]
3. What specific problem the FastText method solves and how? (2-4 sentences) [2 points]
4. Create a word embedding of the Wikipedia News Corpus dataset available from [here](). The Wikipedia News Corpus contains text from Wikipedia's English-language current events page, with dates. It contains ~25,000 instances in text file format. Give a couple of examples of your input text, context, and prediction. Show at least one example where the result is quite bad and provide your thoughts on why. [4 points]

   Submit a single PDF file with no more than 2 pages in length. Any other format or late submission will not be graded.

 # Answer 1

An RNN (Recurrent Neural Network) model of Language Modeling (LM) offers several advantages over a traditional feedforward Neural Network (NN) model:

- Handling Sequential Data: RNNs are designed to handle sequential data by maintaining internal state or memory, allowing them to capture dependencies and relationships across time steps in the input sequence.
- Variable-Length Inputs: RNNs can process inputs of variable lengths, which is crucial for tasks like language modeling where the length of sentences or sequences may vary.
- Contextual Information: RNNs excel at incorporating contextual information from previous inputs into the current prediction, making them effective for tasks requiring understanding of context and temporal dependencies, such as generating text or predicting the next word in a sequence.

 # Answer 2

CBOW (Continuous Bag of Words) and Skip-gram are two popular architectures used in word embedding models like Word2Vec, both developed by Mikolov et al. Each model has its own characteristics and advantages:

CBOW (Continuous Bag of Words):

- In CBOW, the model predicts the target word based on the context words (surrounding words) within a window. The input to the model is a set of context words, and the goal is to predict the target word.
- CBOW is faster to train compared to Skip-gram because it aggregates context information from multiple words to predict one target word.

- CBOW is effective for frequently occurring words and performs well when the context words provide strong clues about the target word.

Skip-gram:

- In Skip-gram, the model predicts the context words (surrounding words) given a target word. The input to the model is a single target word, and the model is trained to predict the context words.
- Skip-gram is slower to train compared to CBOW because it considers each target-context pair separately.
- Skip-gram is particularly useful for infrequent words or capturing semantic relationships between words, as it tries to maximize the likelihood of predicting context words given a target word.

Comparison

- CBOW is efficient and straightforward, making it suitable for tasks where context words provide clear information about the target word (e.g., part-of-speech tagging).
- Skip-gram is more versatile and can capture more nuanced semantic relationships between words, making it better suited for tasks like word analogy and understanding semantic similarities.
- Skip-gram tends to perform better with large datasets and is preferred when the goal is to learn embeddings for rare words or capture fine-grained semantic information.

In summary, CBOW is faster and simpler, while Skip-gram is more flexible and powerful for learning word representations in diverse contexts, especially with larger and more complex datasets. The choice between CBOW and Skip-gram depends on the specific task and the characteristics of the dataset being used.

 # Answer 3

The FastText method addresses the problem of efficiently learning word representations, particularly for handling out-of-vocabulary words and capturing subword information. It achieves this by representing each word as a bag of character n-grams (subword units), allowing the model to learn embeddings not only for known words but also for unseen or rare words by leveraging their character-level similarities to known words. This approach enables FastText to handle morphologically rich languages and improve performance on tasks where robust handling of unseen vocabulary is essential.