Report on Manual Configuration of RNN for Token Counting

Introduction

This report details the design of a simple Recurrent Neural Network (RNN) encoder-decoder model to count occurrences of specific tokens in a sequence. The task involves processing documents composed of tokens {"a", "b", "c", "d", "e", "."} and generating a corresponding sequence of counts. This solution manually sets the weights of the RNN model, negating the need for training with data.

Model Specifications

Input Layer
- Size: 6 (5 for tokens {"a", "b", "c", "d", "e"} and 1 for the end-of-sequence (EOS) token ".")
- Encoding: One-hot encoding is used, where each token is represented by a unique vector in a 6-dimensional space.

Hidden State
- Size: 5, corresponding to the count of each token.
- Role: Accumulates and carries the count of each token through the sequence.

Output Layer
- Size: 6 (5 for the count of each token and 1 binary indicator for sequence end).

Weight Matrices Configuration

Wxh (Input to Hidden State)
- Dimensions: 5x6
- Function: Maps input tokens to their respective count positions.
- Configuration: Set as an identity-like matrix for token counts, with zeros for the EOS token.

Whh (Hidden State to Hidden State)
- Dimensions: 5x5
- Function: Transfers the count from one time step to the next.
- Configuration: Identity matrix, ensuring the transfer of counts is unchanged across time steps.

Why (Hidden State to Output)
- Dimensions: 6x5
- Function: Converts the hidden state (counts) to the output format.
- Configuration: Identity-like matrix for the first five outputs and a mechanism for setting the sixth output (EOS indicator) to 1 upon encountering the "." token.

Example Computation

An input sequence like "badcab." is processed token by token. The RNN updates the hidden state by incrementing counts based on the input and generates the output sequence accordingly. The EOS indicator is set to 1 when the "." token is processed.

Conclusion

The proposed RNN model, with manually configured weights, effectively solves the problem of counting token occurrences without requiring training. This model demonstrates a fundamental understanding of RNN operations and their potential to perform sequential data processing tasks. The simplicity of the model, focusing on direct counting and sequence termination, provides a clear example of how RNNs can be manually tailored to specific computational tasks in NLP.