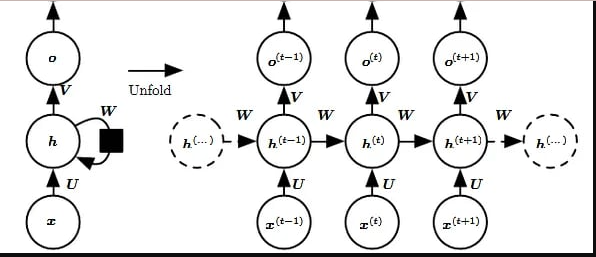
**Experiment 6a**

**Aim: Design and implement a simple RNN model with tensor/keras and check accuracy**

**Description:**

Recurrent Neural Networks (RNNs) are designed to process sequential data by retaining information across time steps. This is achieved through the use of **hidden states**, which act as memory units that carry forward relevant information.



**Input Layer**:

Accepts sequential inputs xt at different time steps t . For example, in text processing, each word in a sentence is an input at a specific time step.

**Hidden Layer**:

The core of the RNN, where the current input (xt)and the hidden state from the previous time step (ht−1) are combined. This layer uses activation functions like tanh or ReLU to update the hidden state:

*ht=f(Uxt+Wht−1+b)*

* + 1. U: Input-to-hidden weight matrix.
    2. W: Hidden-to-hidden weight matrix.
    3. b: Bias term.
    4. f: Non-linear activation function (e.g., tanh).

**Output Layer**:

Produces the output (yt) for each time step based on the current hidden state (ht):

*Yt = g(Vht +c)*

* + 1. V: Hidden-to-output weight matrix.
    2. c: Bias term.
    3. g: Output activation function, often softmax for classification tasks.

**Recurrent Connections**:

The architecture includes feedback loops that enable information to cycle back to earlier layers, creating a dependency between inputs at different time steps.

#### ****Unfolding RNN****

In RNN, unroll represent the sequence processing as a feed forward network, where each time step corresponds to one layer. For example:

A sentence with three words (x1,x2,x3) will result in an unrolled network with three layers processing x1, x2, and x3 sequentially

**Sine Wave as a Dataset**:

* **Periodicity**: The sine wave is periodic, meaning its behavior repeats in a predictable manner. This makes it an ideal dataset for testing models designed to learn sequential dependencies.
* **Easy to Verify Results**: Since the sine wave is mathematically defined, it’s easy to evaluate the accuracy of predictions against the expected values.
* Sine wave data is inherently sequential, where the value at any point depends on its position in the sequence.

#### ****RNN Architecture****

The model is a simple RNN with:

1. **Input Layer**:
   1. The input shape is (seq\_length−1,1)(seq\\_length - 1, 1)(seq\_length−1,1), representing sequences of length seq\_length−1seq\\_length - 1seq\_length−1 with one feature.
2. **Recurrent Layer**:
   1. A SimpleRNN layer with 50 units.
   2. Activation function: tanh, commonly used in RNNs to capture dependencies in sequential data.
3. **Output Layer**:
   1. A Dense layer with a single neuron for predicting the next value in the sequence.

The input data is reshaped to match the expected format for the RNN, which processes 3D tensors:

* Shape: (batch\_size,time\_steps,features)

### ****Training the Model****

* **Epochs**: The model trains for 20 iterations over the training data.
* **Batch Size**: 32 sequences are processed in each batch.
* **Validation Split**: 10% of the training data is used for validation.

### ****Results Analysis****

**Training Process**:

The model iteratively adjusts weights to minimize the MSE between predicted and actual outputs.

Validation data provides feedback on generalization.

**Expected Outcomes**:

**Conclusion:**