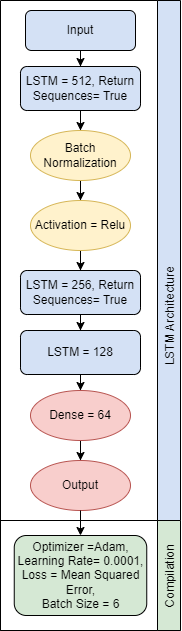
**LSTM Architecture**



**Figure:** LSTM Architecture

The model is implemented using the TensorFlow and Keras libraries. It consists of two LSTM layers, followed by fully connected dense layers. BatchNormalization and ReLU activation functions are applied to enhance the model's training.

1. **Input Layer:** The model starts with an LSTM layer with 512 units. This layer processes the input sequences, where each sequence has a specific length (determined by the `input\_shape`).
2. **BatchNormalization and Activation:** After the first LSTM layer, a BatchNormalization layer is added to normalize the outputs and stabilize the learning process. Then, the ReLU activation function is applied to increase the model's learning capacity.
3. **Second LSTM Layer:** Another LSTM layer follows, with 256 units and `return\_sequences=True` which will return the sequence of hidden states for each time step.
4. **Third LSTM Layer:** A third LSTM layer is added with 128 units. This layer does not return sequences (`return\_sequences=False`), so it only provides the final output for the last time step.
5. **Dense Layers:** Two fully connected dense layers are added after the LSTM layers to further process the extracted features. The first dense layer has 64 units, followed by the final dense layer which is the output layer.
6. **Optimizer and Loss Function:** Adam optimizer with a learning rate of 0.0001 is used, and the mean squared error (MSE) loss function is selected.
7. **Training:** The model is trained using a 5-fold cross-validation approach with a batch size of 6. Early stopping with a patience of 30 is implemented to prevent overfitting and restore the best weights based on validation loss.
8. **Evaluation Metrics:** For each fold of cross-validation, the model's performance is evaluated using three metrics: Mean Squared Error (MSE), Mean Absolute Error (MAE), and Root Mean Squared Error (RMSE).