# **Machine Translation Project: English to Russian Seq2Seq Model**

## **Overview**

In this project, you will implement a sequence-to-sequence (Seq2Seq) neural network to translate English sentences to Russian. You will build an encoder-decoder architecture using LSTMs and train it on a dataset of English-Russian sentence pairs.

## **Learning Objectives**

* Understand and implement sequence-to-sequence models
* Work with natural language processing techniques for machine translation
* Handle text preprocessing and vocabulary creation
* Implement teacher forcing and understand its importance
* Evaluate translation quality

## **Dataset**

You will use a dataset of English-Russian sentence pairs provided in a text file (rus.txt). The file contains tab-separated pairs with English sentences in the first column and corresponding Russian translations in the second column.

## **Project Requirements**

### **1. Data Preprocessing**

* Load the text file containing English-Russian sentence pairs
* Clean the text by removing punctuation, numbers, and converting to lowercase
* Create vocabularies for both English and Russian words
* Add special tokens (<sos> and <eos>) to the Russian sentences
* Split the data into training and validation sets

### **2. Create Data Loaders**

* Implement a custom Dataset class for translation data
* Handle converting words to indices using vocabulary lookups
* Implement a collate function to pad sequences of different lengths
* Create DataLoader objects for both training and validation sets

### **3. Model Architecture**

* Implement an Encoder class with:
  + Word embedding layer
  + LSTM layer that outputs final hidden and cell states
* Implement a Decoder class with:
  + Word embedding layer
  + LSTM layer
  + Fully connected output layer
* Implement a Seq2Seq class that combines the encoder and decoder
  + Handle teacher forcing with a configurable ratio
  + Generate output sequences one token at a time

### **4. Training Loop**

* Implement a training function that:
  + Sets the model to training mode
  + Computes loss and updates parameters
  + Uses gradient clipping to prevent exploding gradients
* Implement an evaluation function that:
  + Sets the model to evaluation mode
  + Computes validation loss without teacher forcing
* Create a main training loop that:
  + Trains for a set number of epochs
  + Saves the best model based on validation loss
  + Reports progress

### **5. Translation Function**

* Implement a function to translate English sentences to Russian
* Handle both string input and tensor input
* Generate translations token by token until reaching the end token or max length

## **Implementation Guidelines**

* Use PyTorch for implementing the neural network
* Use embedding dimension of 256 and hidden dimension of 1024
* Use the Adam optimizer with a learning rate of 0.001
* Use CrossEntropyLoss and ignore padding tokens
* Use gradient clipping with a threshold of 1.0
* Train for at least 10 epochs