SMS Spam Detection

Approach-1 (Bi-LSTM)

1. Loading the dataset.
2. Detect the language of text with langdetect library.
3. Removing stopwords with using NLTK corpus, punctuation, etc.
4. Convert text to lowercase.
5. Tokenization and lemmatization.
6. Text to numeric convertion of data.
7. Splitting the dataset into train set and test set.
8. Creating Bi-LSTM model.
9. Training the model with optimizing hyperparameters.
10. Evaluating the model.

Approach-2 (Llama-2)

1. Loading the dataset.
2. Tokenization
3. Splitting the dataset into train set and test set
4. Loading Llama 2 model.
5. Optimizing hyperparameters.
6. Training model.

Getting Access to Model and Token

This model is gated. That's why, to do this project we need to have a HuggingFace site account and a token for accessing the Llama-2 model.

After creating those requirements, users needs to apply for an access pass to load the model.

Tokenizer

Tokenizer parameters:

- padding="max_length" → Ensures all sequences have the same length (max_length=128).
- truncation=True → Cuts off messages longer than 128 tokens.
- max_length=128 → Limits tokenized sequences to 128 tokens.

batched=true in tokenizer process takes inputs as a batch, which is more efficient.

Tokenizer Padding

In NLP models (like LLaMA 2), input sentences have different lengths. However, deep learning models require fixed-length inputs for efficient processing.

Padding ensures that shorter sentences match the longest one in the batch.

After Tokenization

Deletes raw data from dataset. Only tokenized data remains. Y values of dataset must be named as “labels”.

Split dataset into train and test set.

Loading The Model

Load the AutoModelForSequenceClassification and give number of label as a parameter.

While running the code, program is going to download the model. The model is approximately 12.5GB.

Parameters/Arguments of Training

- “output_dir” specifies where to save the model logs and outputs.
- “evaluation_strategy” evaluates the model with selected method. (In this case epochs)
- “save_strategy” defining checkpoint frequency. (In this case it saves checkpoint in every epoch).
- “save_total_limit” limits the number of checkpoints. Older ones gets deleted if count exceeds the limit.
- “per_device_train_batch_size” batch size used in training.
- “per_device_eval_batch_size” batch size used in evaluation process.
- “num_train_epochs” number of epochs.
- “logging_dir” specifies the log path.
- “logging_steps” printing frequency of logs. It creates logs by every X steps.
- “push_to_hub” determines if we wanted to push the model into Hugging Face Hub.

Defining Trainer

While creating trainer we need: model, training arguments (parameters above), training set and evaluation (test) set.