**What is Sanitization?**

**Jo 250 words ka article araha usko 30-50 words mei transform karna**

Data Compression for e.g. 250 words to 30 to 50 words

**What is Filtration of data?**

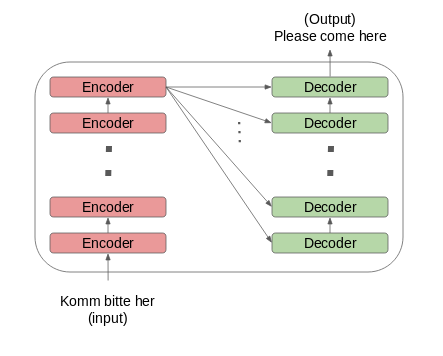
* + Convert Text to lower case
  + Remove Special Characters/Unwanted Characters
  + Correction of any typos 🡪 (TextBlob/PyspellChecker/autocorrect Library)
  + Normalization (such as running, ran into run)
    - Stemming
    - Lemmatization

Stemming uses the stem of the word while lemmatization uses the context in which the word is being used.

* + Tagging (Parts of Speech such as noun, pronoun, verb etc.)
  + Chunking/Chinking: chunking breaks those word. While chinking further break those words (Chinking is not always necessary)

**What is Transformer?**

Works as an encoder and decoder



**NLP working:**

1. Perform Cleansing of Data
   1. Convert Text to lower case
   2. Remove Special Characters/Unwanted Characters
   3. Correction of any typos 🡪 Normalization
      1. Stemming
      2. Lemmatization
   4. Chunking/Chinking
2. The cleanse data would be sent to machine learning/deep learning model -----🡪 (Sending data to Transformer)
3. Tune transformer according to our requirement
4. Consist of 2 phases training and testing. In the training phase we will train the model to generate a summary from a given news article using a dataset of 7000-8000 articles. The testing phase would then consist of our own system producing summarized weather news reports.

**Why are you using Libraries?**

We are using libraries to generate our own model which can perform those tasks required for our system.

1. Typos ---🡪 (TextBlob/PyspellChecker/autocorrect Library)
2. Machine Learning/ Deep Learning ------ > Transformer -----🡪 GPT3, BERT libraries
3. TTS ---🡪 GTTS
4. Web scrapping -----🡪 Urllib, BS4, Selenium
5. Filtration of data------🡪 NLTK, NLG, SpaCy

**Comparison chart (Text to Speech Algorithm)**

3 speech recognition systems were compared which included the Google API, Microsoft API and sphinx 4. The experimental details are as follows:

The audio files were selected from various sources to evaluate the Microsoft API, Google API, and Sphinx-4. According to CMUSphin, Sphinx-4's decoder supports only one of the two specific audio formats (16000 Hz / 8000 Hz). WAV files not supported by all three so they design tool which recognition all audio files in the same format (16000 Hz / 8000 Hz).

**Results:**

According to the results the Sphinx-4 library gave a 37% WER), Google Speech API gave 9% WER and Microsoft Speech API gave 18% WER.

WER here represents Word Error Rate which tells how many words are not transcribed correctly with the reference words

Thus Google API provided a better result compared to the other 2.

# Parser/Scrapper function:

->three domains i.e. entertainment, sports, technology

->we give different links w.r.t above mentioned domains

->hence we have one table for URLs

->channelname/sports or channelname/technology (we have to give links in this format)

->parse all the articles from the links

->body k ander jitny tags hain jitny urls hain un tamam ko uthana hy and store them into an array.

->and iterate them one by one

->behtar yeh hay k hum code ko sath mai rakhain agr article code sath mai nhi rakh sakty tou

koi unique id hou because after 4 hours agr koi bhi updated news na ai hou tou wo match

kar sakay that whether he already parse this news or not ...because of the uniqueness of the

article id or whatever the unique id we assign them will be match with the privious one and

helps to identify the uniqueness of article or news.

->Runs this method for every page and del all the data from that except the (data present in

the <p> tag of the article,label ,date, and image)we have to stored these things

->ab yahan sai config table start hojayega

->which tells that sama.tv/technology sai jo data uth k ayrga usmai yeh yeh mere kam k positions

hain ya yeh yeh meri classes hain ya yeh yeh meri ids hain

->jahan sai tumhy is particular type ka data mily usey utha k DB mai daldo

->yahi same silsila har hamari har url k sath individual chal raha hoga

->In text sammarization we use rogue (is a statical word or statistical calculation)jis k

through hum apna syntax define krty hain k wo kitna acha work kr raha or phir usko rouge k

sath compare karty hain.

->2 sai 3 algos hum nai choose kiyea hain text summerization k liyea jis ka rouge score acha

ayega hum wo use karlain gy

--> Our system right now caters for three domains (i.e. sports, technology and entertainment) so the admin can select any of that domain

--> The admin can give different url links, to scrape data, w.r.t above mentioned domains (the format of the link should be "channel\_name/domain\_name" for example dawn.com/sports).

--> These URLs would be saved in the URLs table in database

--> The parser would then parse all the articles from the given link.

--> The parser would work in such a way that it would save all the article urls in the database. Then parse each of these articles one by one.

--> Each article/article url would be assigned a unique id so when the data is once again scrapped from the links after a set time period (such as 4 hours) we can compare the new articles with the previously scrapped articles present and see if any of the new articles scrapped are the repeated ones.

--> The above mentioned points would be repeated for all of the url links. Data would be gatheed this way and only the data mentioned in the <p> tag would be saved in database while the rest of it would be discarded.

--> Next we now need to start building config table

--> For this the admin the particular url that he wants to configure

--> And then select the desired tag and the related class or id method in that.

--> These values and the related data would then be saved in database.