A REPORT

ON

PRODUCT ATTRIBUTE AND SPECIFICATION EXTRACTION FROM WEB

BY

ARNAV YAYAVARAM	2021A7PS3117H
AYUSH BHAUWALA	2021A7PS0180H

AT

Amazon Applied Science, Hyderabad (Online)

A Practice School-I Station of

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

JULY 2023

A REPORT

ON

PRODUCT ATTRIBUTE AND SPECIFICATION EXTRACTION FROM WEB

BY

ARNAV YAYAVARAM 2021A7PS3117H B.E. Computer Science

AYUSH BHAUWALA 2021A7PS0180H B.E. Computer Science

Prepared in partial fulfilment of the

Practice School-I Course Nos.

BITS C221/ BITS C231/ BITS C241

AT

Amazon Applied Science, Hyderabad (Online)

A Practice School-I Station of

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

JULY 2023

ACKNOWLEDGEMENTS

We are immensely thankful to all the individuals who have guided us in completing this report. Our gratitude extends to our mentors, **Mr. Jiten K Rana** and **Mr. Anish N**, whose invaluable mentorship has been instrumental throughout our internship journey. Their constant guidance and extensive knowledge have significantly contributed to our learning and growth in our respective roles. We genuinely appreciate their willingness to share their expertise and the encouragement and support they provided us.

Additionally, we would like to express our profound appreciation to **Amazon India** for granting us the opportunity to intern with their esteemed company. This internship has provided us with valuable experiences and learnings, and we are grateful for the chance to work with such a talented and dedicated team.

We must acknowledge and thank **Professor Morapakala Srinivas**, our PS faculty-in-charge, for his exceptional guidance and unwavering support, which have been invaluable in shaping our internship experience. His insights and feedback have greatly enhanced our understanding of the subject matter, and their mentorship has played a vital role in our professional development.

Lastly, we sincerely thank our families and peers for their unwavering support throughout our internship. Their encouragement and understanding have been crucial to our success, and we are truly grateful for their continuous presence.

In conclusion, we extend our heartfelt thanks to all the individuals mentioned above for their contributions to our growth and learning during this internship.

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

(RAJASTHAN)

PRACTICE SCHOOL DIVISION

Station: Amazon, Applied Science **Centre:** Hyderabad (Online)

Duration: 2 months **Date of Start:** 30th May 2023

Date of Submission: 20th July 2023

Title of the Project: Product Attribute and Specification Extraction from Web

ID No.: 2021A7PS3117H 2021A7PS0180H

Name: Arnav Yayavaram Ayush Bhauwala

Discipline: B.E. Computer Science B.E. Computer Science

Name and designation of the experts

Mr. Anish N – Applied Scientist

Mr. Jiten K Rana – Applied Scientist

Name of the PS Faculty: Prof. Morapakala Srinivas

Keywords: Machine Learning, Natural Language Processing, Named Entity Recognition

Project Area: Natural Language Processing

Abstract: The goal of this project is to develop a knowledge base (KB) of product names and their respective specification information by extracting details from product, manufacturer, and review webpages. These knowledge bases will be used to create product profiles, enrich the Amazon catalogue, and group products.

TABLE OF CONTENTS

INTRODUCTION

This report presents my internship experience and learning at Amazon – Applied Sciences. We worked with the Retail Business Services (RBS) team. Our project is to build a knowledge base of product names and their attributes by extracting details from online websites. The specific product assigned to me is digital cameras.

This report provides a brief overview of the scope of the work completed and the background information necessary to understand the work. It also states the purpose of the report, which is to share the learnings from the internship and to demonstrate the skills that were developed.

In this project, we build a knowledge base for products by extracting structured and unstructured text from online websites using web scraping.

We use a machine learning model to identify the brand and model name of a product from unstructured text. This model is trained on a dataset of labelled text, and it uses Named Entity Recognition (NER) to identify specific entities in the text. Once the model is trained, we can use it to identify the brand and model name of any product from unstructured text.

OBJECTIVE

The project's primary objective is to develop a comprehensive knowledge base of product attributes for Amazon. This knowledge base will encompass vital information such as features, specifications, and other attributes for each product available on the platform. Its purpose is to facilitate the creation of informative product profiles, enrich the Amazon catalogue, and enable more meaningful product grouping.

Further Use of Knowledge Base

By leveraging this knowledge base, Amazon aims to enhance its search engine results pages (SERPs) and recommendations. The wealth of information the knowledge base provides will empower Amazon to gain deeper insights into its product offerings, enabling a better understanding of customer needs and preferences.

Moreover, the knowledge base will significantly improve the overall customer experience on Amazon. With easily accessible and detailed product information, customers will enjoy a seamless browsing experience, effortlessly locating the desired products and gaining comprehensive insights into their features and benefits.

Ultimately, the development of this knowledge base serves as a fundamental building block for Amazon's continual growth and success. By harnessing the power of product attributes, Amazon can better serve its customers, deliver personalized recommendations, and further solidify its position as a trusted and customer-centric e-commerce platform.

WEB SCRAPING

What is it?

Web scraping refers to the extraction of data from a website. This information is collected and then exported into a format that is more useful for the user. Be it a spreadsheet or an API.

Web scraping is the process of extracting data from a specific web page. It involves making an HTTP request to a website's server, downloading the page's Html, and parsing it to extract the desired data.

The first part of the project was to scrape relevant data from websites for different product types, the first being digital cameras. The websites I used to scrape relevant data were:

https://www.digicamdb.com/cameras/

https://www.dpreview.com/products/cameras/all?page=1

We required unstructured text or a title containing a short description from which brand and model name may be extracted to obtain more normalized data.

Methodology:

To accomplish this, I utilized the Python package Beautiful Soup which is extensively used for web scraping. The scraped data was then stored in CSV (Comma-Separated Values) files.

For the first website, I scraped a short description of the sensor info, which would act as a title or a short description, along with the specifications displayed in a tabular format.

The second website included the Manufacturer's description, which contained relevant data which would act as the title for this website.

Along with this information, I extracted the brand and model name as separate attributes in the subsequently created CSV files, which would be used to train and test the spacy and BERT models.

NAMED ENTITY RECOGNITION

Named Entity Recognition (NER) is an essential part of Natural Language Processing (NLP) that seeks to extract information from text.

It does this by detecting and categorizing important information known as named entities.

These entities mentioned in the unstructured text are classified into pre-defined classes, which could be a person's name, an organization, locations, medical codes, time expressions, quantities, monetary values, percentages, and other numerical data.

In this project, NER is utilized to extract information regarding 2 pre-defined categories, namely the Brand and Model of digital cameras.

I have implemented these concepts of NER using two methods:

i. Using spaCy
ii. Using BERT Model

NER uses algorithms that function based on grammar, statistical NLP models, and predictive models.

spaCy

spaCy is an open-source software library for advanced natural language processing, written in Python and Cython.

Unlike NLTK (Natural Language Toolkit), which is widely used for teaching and research, spaCy focuses on providing software for production usage. spaCy also supports deep learning workflows that connect statistical models trained by popular machine learning libraries like TensorFlow, and PyTorch.

spaCy features convolutional neural network models for part-of-speech tagging, dependency parsing, text categorization, and named entity recognition (NER).

Prebuilt statistical neural network models to perform these tasks are available for 23 languages, including English, Portuguese, Spanish, Russian, and Chinese, and there is also a multi-language NER model.

Additional support for tokenization for more than 65 languages allows users to train custom models on their own datasets.

Difference between NLTK and spaCy:

1. NLTK provides a wide range of algorithms, making it useful for researchers but overwhelming for developers. SpaCy focuses on the best algorithm and keeps it up to date.
2. NLTK supports various languages, while spaCy has statistical models for seven languages and supports multi-language named entities.
3. NLTK is a string-processing library, while spaCy uses an object-oriented approach with document, word, and sentence objects.
4. SpaCy supports word vectors, unlike NLTK.
5. SpaCy's performance is generally better due to its use of state-of-the-art algorithms, but NLTK outperforms it in sentence tokenization.

IMPLEMENTATION

To train a model in spaCy for NER we must create a dictionary with the custom entities and annotations.

The custom entities being {classes: [BRAND, MODEL], 'annotations'}

Annotations is a list of dictionaries that contain the text, the entities it contains and the start and end index of the entity in the text

BERT

BERT (Bidirectional Encoder Representations from Transformers) is a Natural Language Processing Model proposed by researchers at Google Research in 2018. When it was proposed, it achieved state-of-the-art accuracy on many NLP tasks.

BERT is a transformer-based model, which uses an attention mechanism to learn contextual relations between words in a text.

TRANSFORMERS

Transformers are a type of deep learning architecture that has been widely used in natural language processing (NLP). Transformers are designed to handle sequential data, such as text, by capturing contextual relationships between words.

At the core of a Transformer model are self-attention mechanisms. which allows each word in a sequence to attend to all other words, considering their relative importance for understanding the context. This enables the model to give more weight to relevant words and capture long-range dependencies effectively.

Training

In the case of BERT, the model consists of an encoder stack of Transformer layers. During pre-training, BERT is exposed to many unlabelled text and learns to predict missing words by considering the surrounding context bidirectionally. This process helps BERT to capture rich, contextualized representations of words.

Tuning

After pre-training, BERT is fine-tuned on specific NLP tasks. For fine-tuning, task-specific labelled data is used to train BERT on specific objectives, such as text classification or named entity recognition. By adjusting the model's parameters during fine-tuning, BERT adapts its learned representations to the nuances and requirements of the target task.

IMPLEMENTATION

[Using Pytorch and Hugging Face API]

- PyTorch: It is an open-source deep-learning Python framework primarily used for building and training neural networks, which I used to train the BERT model.
- Hugging Face: Hugging Face, Inc. is an American company that develops tools for building applications using machine learning. It has a platform for its transformers library built for natural language processing applications and its platform that allows users to share machine learning models and datasets.

We used the IOB format (short for inside, outside, beginning), also commonly referred to as the BIO format, a common format for tagging tokens.

To train a model using BERT in PyTorch, we must tag each word from the text with labels.

In this format, the letter “B” is used to denote that a word is the beginning of an entity, “I” denotes the word is inside an entity, and “O” denotes it is not part of any entity.

Hence, we have 5 labels in total - B-br, B-mo, I-br, I-mo, O.

- br-refers to the custom tag for brand name
- mo-refers to the custom tag for model number or model name

To implement BERT, we used stochastic gradient descent.

Stochastic gradient descent (often abbreviated SGD) is an iterative method for optimizing an objective function with suitable smoothness properties (e.g., differentiable or subdifferentiable).

Parameters:

Learning rate ------ $2e^{-3}$

Batch size---------- 2

Epochs ---------- 5

Result

Accuracy:

-spaCy: 100%

-BERT: >99.5%

Overfitting

Problem: Using spaCy and BERT, custom models with accuracy with nearly 100 percent accuracy were created.

This is indicative of overfitting, which is undesirable. From the data scraped from both websites, the format of the text remained quite structured, with the sentence beginning with the brand name followed by the model. As all the data follows this pattern, the model makes errors if data is passed in a different format.

Correction: One way to prevent overfitting would be to make the training data more complex artificially. We can try to find more complicated information that is less structured, or we can obtain such data by adding noise to the data. We can jumble these sentences and remove brand and model names to create more variability.

This will decrease overall efficiency but will improve the model and make it more sturdy and less susceptible to the problems of overfitting.

ADDING NOISE

1. Using nlpaug

   nlpaug is a Python library designed to facilitate data augmentation in natural language processing tasks. Data augmentation involves applying various techniques to modify and expand a dataset, which can help improve the performance and robustness of machine learning models.

   The nlpaug library provides a range of augmentation methods specifically tailored for textual data, such as:

   - Textual Transformation
   - Textual Generation
   - Textual Embedding
   - Textual OCR (Optical Character Recognition)

   These augmentations can help to prevent overfitting, enhance generalization, and increase the diversity of training data for NLP tasks such as sentiment analysis, text classification, machine translation, and more.

   When we used nlpaug on our data, it was not able to augment the text correctly as it was not able to understand the context of the text. Because of this, we did not use nlpaug to add noise to our data.

2. Jumbling words

   We jumbled the words randomly so that the relative order of brand and model name in the text changes. We also removed the brand and model names from certain training examples to diversify the training data.

3. Using Falcon LLM and Langchain

   Falcon LLM is a foundational large language model (LLM) with 40 billion parameters trained on one trillion tokens, developed by Technology Innovation Institute (TII).

   LangChain is a purpose-built framework aimed at streamlining the development of applications that leverage large language models. Serving as an integration framework for language models, its applications closely align with those of language models in general. These applications encompass document analysis and summarization, chatbot implementation, and code analysis, among others.

We used Langchain to prompt Falcon LLM to generate a meaningful sentence out of the product's title.

This is the prompt that was used:

Question - Convert the given text of a refrigerator into a meaningful sentence. Make sure to not use any special characters like , | - or any other character.
Text: {question}
In this text, the brand is {brand} and model number is {model}. Make sure that the brand and model number are unchanged in the final output.

Answer - """

Example:

Input

Kelvinator 170 L Direct Cool Single Door 3 Star Refrigerator (KRD A190MRP)

Output

This is a Kelvinator 170 L Direct Cool Single Door 3 Star Refrigerator. It has an energy rating of A+ which means it uses less energy and is more efficient than the standard.

Preparing unseen data

We scrape data from different websites than what was used for training so that we have a better estimate of the performance of our model on unseen data.

For refrigerators, the data was scraped from www.gadgetsnow.com

For cameras, the data was scraped from www.dpreview.com

We compared results on unseen data after training the model on data with and without noise.

For Refrigerators

	Accuracy	Precision - Brand	Recall - Brand	Precision - Model	Recall - Model
Without noise	73.9%	58%	48%	37%	49%
With noise	87%	66%	77%	71%	71%

For Refrigerators & Cameras combined

	Accuracy	Precision - Brand	Recall - Brand	Precision - Model	Recall - Model
Without noise	70.8%	45%	56%	19%	30%
With noise	85.8%	63%	81%	43%	64%

PERFORMANCE METRICS

1. Precision:

Precision is the ratio between the True Positives and all the Positives. In simple words, of the samples we predicted to be True how many of those were actually True.

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

2. Recall:

The recall is the measure of our model correctly identifying True Positives.
Recall also gives a measure of how accurately our model is able to identify the relevant data. We refer to it as Sensitivity or True Positive Rate.
Out of all the True samples, how many did we classify correctly as True.

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

3. Accuracy:

Accuracy is the ratio of the total number of correct predictions and the total number of predictions.

$$Accuracy = \frac{TrueNegatives + TruePositive}{TruePositive + FalsePositive + TrueNegative + FalseNegative}$$

Using accuracy as a defining metric for our model makes sense intuitively, but more often than not, it is advisable to use Precision and Recall too. There might be other situations where our accuracy is very high, but our precision or recall is low.

For example, if there is a particular sentence with 10 words.

A particular NLP model may tag all 10 words with the ‘O’ tag.

However it has tagged the ‘Brand’ and ‘Model’ incorrectly and has not identified the entities correctly.

In this case the model would show an accuracy of 80% which is misleading.

This result leads us to believe that the performance of the model is quite good when it is not able to correctly carry out the main objective of the model.

ANALYSIS OF RESULTS

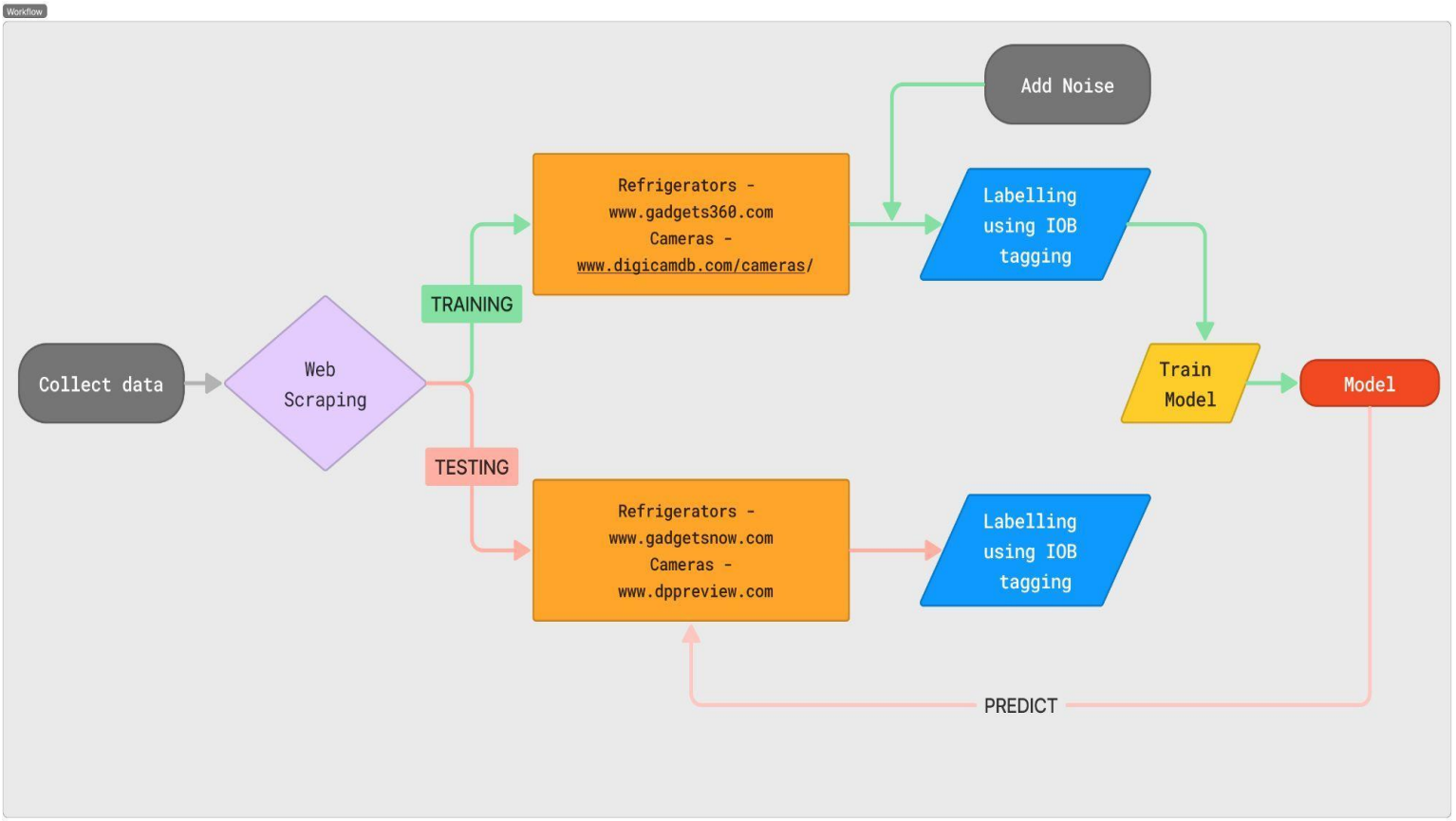
By incorporating various noise-inducing techniques, such as jumbling, sentence reorganization, and textual augmentation, a notable enhancement in the performance of all measured parameters is observed. This augmentation strategy effectively mitigates the issue of overfitting that arises from an abundance of closely related data points, resulting in a more robust and generalized model.

In the domain of refrigerator testing, an interesting observation has been made regarding certain colors that lack obvious indications and are sometimes capitalized. Consequently, the model's accuracy and precision are adversely affected, leading to suboptimal results.

Moreover, the presence of capitalized words poses a challenge for the model's ability to accurately identify brand names, contributing to potential inaccuracies in its predictions.

Overall, the implementation of noise-based techniques within the LangChain framework has proven to be a valuable approach, significantly boosting performance metrics and rectifying overfitting concerns.

FLOWCHART



This flowchart details the process of creating an NLP model and the sequence of steps we carried out and the overlaying logic.

CONCLUSION

In conclusion, the spacy and BERT models developed in this project have proven effective in identifying the brand and model name of digital cameras from unstructured text. These models provide a solid foundation for building more sophisticated models that can handle edge cases and expand to other product types. By refining and extending these models, they can contribute significantly to Amazon's product catalogue and specifications.

The information extracted from text can be leveraged to enhance the accuracy of product recommendations. By understanding the specific brand and model of a product, recommendations can be tailored more precisely to customers' preferences, improving their overall shopping experience. Additionally, this data can provide customers with detailed information about the products they are considering purchasing, enriching their understanding, and enabling them to make more informed decisions.

During this internship, we not only made technical contributions to the development of NLP models but also immersed ourselves in the fascinating field of natural language processing. This experience deepened our understanding of the inherent challenges in NLP model development and exposed me to state-of-the-art technologies used to overcome these challenges. We are confident that the skills and knowledge gained during this internship will be invaluable in shaping my future career endeavours. We are grateful for the opportunity to have been a part of this project.

REFERENCES

Ng, A. (n.d.). Named Entity Recognition Tagging. *CS230 Stanford.* https://cs230.stanford.edu/blog/namedentity/

Kafritsas, N. (2022, August 12). *Named Entity Recognition with Deep Learning (BERT) — The Essential Guide.* Towards Data Science. https://towardsdatascience.com/named-entity-recognition-with-deep-learning-bert-the-essential-guide-274c6965e2d

Turing. (n.d.) A Comprehensive Guide to Named Entity Recognition (NER). https://www.turing.com/kb/a-comprehensive-guide-to-named-entity-recognition

Alammar, J. (n.d.) The Illustrated Transformer. http://jalammar.github.io/illustrated-transformer/

Hofer, M. (2018, May 4). *Deep Learning for Named Entity Recognition #1: Public Datasets and Annotation Methods.* Towards Data Science. https://towardsdatascience.com/deep-learning-for-ner-1-public-datasets-and-annotation-methods-8b1ad5e98caf

Mattingly, W.J.B. (2021 January). How to Train spaCy NER Model. *Introduction to Named Entity Recognition.* https://ner.pythonhumanities.com/03_02_train_spacy_ner_model.html

GLOSSARY

Web Scraping: The process of extracting data from websites by making HTTP requests, downloading HTML content, and parsing it to extract relevant information.

Beautiful Soup: A Python package commonly used for web scraping tasks to parse HTML and XML documents.

CSV (Comma-Separated Values): A file format that stores tabular data as plain text, with each record separated by commas.

Named Entity Recognition (NER): A natural language processing task that involves identifying and categorizing named entities, such as names of persons, organizations, locations, dates, etc., in unstructured text.

spaCy: An open-source Python library for advanced natural language processing, focusing on production usage and supporting various NLP tasks, including NER.

BERT (Bidirectional Encoder Representations from Transformers): A transformer-based NLP model proposed by Google Research, known for achieving state-of-the-art accuracy on multiple NLP tasks.

Transformers: Deep learning architecture widely used in NLP to capture contextual relationships between words in sequential data, like text.

Pre-training: The initial training phase where BERT learns contextual representations by predicting missing words from unlabelled text data.

Fine-tuning: The process of training BERT on specific NLP tasks using labelled data to adapt its learned representations to the target task.

SGD (Stochastic Gradient Descent): An iterative optimization algorithm used to train neural networks and find the optimal set of parameters by minimizing the loss function.

Overfitting: A condition where a machine learning model performs well on the training data but poorly on unseen data due to memorizing patterns specific to the training set.

Data Augmentation: Techniques used to expand and diversify the training data by applying modifications to the original dataset, enhancing the model's generalization and robustness.

IOB Format (Inside, Outside, Beginning): A common format for tagging tokens used in NER, where "B" denotes the beginning of an entity, "I" indicates a token inside an entity, and "O" means the token is not part of any entity.

Hugging Face: An organization that develops tools and platforms for building machine learning applications, known for its transformers library in NLP.

Precision: A metric that measures the ratio of correctly predicted positive instances to all predicted positive instances.

Recall: A metric that measures the ratio of correctly predicted positive instances to all actual positive instances.

Accuracy: A metric that measures the ratio of correct predictions to the total number of predictions.

Langchain: A framework developed for the integration of large language models (LLM) like Falcon LLM into various NLP applications.

Falcon LLM: A large language model developed by Technology Innovation Institute (TII) with 40 billion parameters, used for natural language processing tasks.

Textual Augmentation: Techniques applied to textual data to modify and expand the dataset, enhancing the performance and robustness of machine learning models.

Sentence Reorganization: A noise-inducing technique that shuffles the words in a sentence to create variations in the training data.

Jumbling: A noise-inducing technique that randomly rearranges the order of words in a sentence to increase data diversity.

Textual OCR (Optical Character Recognition): An augmentation method for textual data that involves transforming the text using OCR techniques.

Generalization: The ability of a machine learning model to perform well on unseen data, indicating its capacity to learn patterns and make accurate predictions beyond the training data.