Unit-5

Multilingual Information

Multilingual Information Retrieval - Introduction, Document Pre-processing, Monolingual Information Retrieval, CLIR, MLIR, Evaluation in Information Retrieval, Tools, Software and Resources.

Multilingual Information Retrieval :-

MIR refers to the field of study and technique involved in retrieval relevant information from multilingual sources (or databases.

It addresses the challenges posed by different languages in IR systems.

CLIR: Cross Language IR.
Techniques for retrieving information written in one language using queries written in another languages.

MI: Machine Translation
Integrating translation systems to bridge language gaps b/w queries and documents.

Multilingual Search Engines:
Development and optimization of search engines that can handle queries and documents in multiple languages.

Language Identification:
Techniques to automatically detect the language of a document (or query.

Cross-Lingual Document Similarity:
Methods for measuring similarity b/w documents in different languages.

Multilingual Text classification:

Techniques for classifying text documents that are writte in different languages.

MIR encompasses a range of techniques and technologies aimed at enabling efficient access to information across languages, ensuring that users can explore, analyze, and utilize data regardless of linguistic constraints.

MIR addresses the complexities of searching and retrieving information from multilingual datasets.

By leveraging advances in NLP, ML, and cross-cultural communication, MIR not only facilitates cross-language information access but also enhances the effectiveness of global information systems.

Document Preprocessing:-

Document preprocessing in the content of IR involves several key steps to prepare textual data for effective searching and analysis.

Tokenization:

Breaking down the text into tokens, which are typically words or subwords. This step involves splitting sentences into individual words or meaningful subword units.

Lowercasing:

Converting all text to lowercase to ensure consistency in word matching, as retrieval systems are usually case-insitive.

Stopword Removal:

Removing common words that do not contribute

significantly to the meaning of a document and may hinder retrieval performance.

Normalization:
Standardizing text by reducing words to their base (or root form (lemmatization) (or converting different forms of a word to a common base (stemming).

Removing Punctuation and Special characters:
Eliminating punctuation marks and non-alpha-numeric characters that do not contribute to the meaning of the text.

Encoding and vectorization:
Converting processed text into numerical representations, such as term frequency-inverse document frequency (TF-IDF) vectors or word embeddings, which are suitable for computational analysis and similarity calculations.

Handling Numerical Data and Dates:
Converting numerical data and date formats into standardized representations suitable for retrieval and analysis.

Language Identification:
Determining the language of the document, which is crucial for multilingual IR systems.

Filtering and Preprocessing for Specific Apps:
Tailoring preprocessing steps to specific app needs, such as entity recognition, sentiment analysis (or topic modeling.

Document Syntax and Encoding:
Document syntax and encoding refer to how textual documents are structured and

represented in computer system.

Document Syntax:

Structure: The organization and format of a document, including paragraphs, headings, lists, and other structual elements.

Markup Languages: Formats like HTML, XML, Markdown, etc, define how content is structured using tags and elements.

Syntax Highlighting: Used in text editors and IDEs to visually distinguish different parts of code or markup languages based on syntax rules.

Document Encoding:

Character Encoding: Defines how characters are represented as bytes in computer memory and files. Common encodings include UTF-8, ASCII, and UTF-16.

Unicode: A standard character encoding system that supports multiple languages and characters from various scripts worldwide.

Binary Representation: How text is stored and processed as binary data in CS.

Tokenization:-

Tokenization is a fundamental process in NLP and IR. It involves breaking down a text into smaller units called tokens, which can be words, subwords, or even characters, depending on the level of granularity required.

Tokenization is the process of dividing a text into meaningful units, called tokens. These tokens are the basic building blocks of for subsequent NLP tasks such as parsing, text analysis, and indexing for IR.

The primary goal of tokenization is to segment text into manageable pieces that can be processed effectively by computational systems.

Word Tokenization:

In this, text is split into words based on white space & punctuation.

→ Character Tokenization
→ Subword Tokenization.

Importance:

→ Text Processing
→ Normalization.
→ Feature extraction

Challenges:

→ Ambiguity
→ Language Specificity
→ Domain-Specific Terms.

Normalization:

Normalization in the content of NLP refers to the process of transforming text into a standard form that improves the consistency and comparability of textual data.

Techniques:

Lowercasing: Converting all text to lowercase ensures that words like "Hello", "hello", and "HELLO" are treated as same token.

Removing Accents and Diacritics:

Normalizing characters by removing accents (or diacritical marks ensures consistent representation of words, especially in languages with accented characters.

Expanding contractions: Converting contractions like "don't" to "do not" improves consistency in text analysis.

Removing Special Characters: Eliminating non-alphanumeric characters, punctuation marks, or symbols that do not contribute to the meaning of the text.

Stemming: Reducing words to their base or root form (eg: running → run) using stemming algms. This helps in reducing inflectional forms and improves text normalization for retrieval and analysis tasks.

Lemmatization:
Similar to stemming, but considers the context and meaning of words to produce their lemma (m. dictionary form (eg. converting "am", "are", "is" to "be").

Importance:
→ Enhances text comparability
→ Facilitates ML
→ Improves IR.

Monolingual IR:-

Monolingual IR focuses on retrieving and managing information within a single language.
Unlike multilingual IR, monolingual IR is concerned with optimizing search and retrieval systems within the content of a single language.

Text Indexing:
Creation of indexes that map terms to documents within a single language corpus.

Query Processing:

Techniques for processing user queries and matching them against indexed documents to retrieve relevant information.

Relevance Ranking:

Methods to rank retrieved documents based on their relevance to the user query.

Document Preprocessing:

Steps such as tokenization, stopword removal, stemming or lemmatization and normalization to prepare documents for indexing and retrieval.

Retrieval Models:

Implementation and optimization of retrieval models that determine the relevance of documents to queries based on statistical or probabilistic measures.

Evaluation Metrics:

Metrics like precision, recall, and F1-score are used to evaluate the effectiveness of retrieval systems in retrieving relevant documents.

Domain-specific Apps:

Customization of retrieval systems for specific domains to handle domain-specific terminology and requirements.

Challenges:

→ Ambiguity and Polysemy
→ Document Variability.
→ Scalability
→ Query Understanding.

Document Representation:-

Document representation in the context of IR refers to how textual documents are transformed into structured formats that can be processed, analyzed and indexed by computational systems.

BOW Model: Bag-of-Words.

Represents documents as unordered collections of words, disregarding grammar and word order but retaining multiplicity.

Each unique word in the document becomes a feature in the vector representation.

TF-IDF: Term Freq - Inverse Docu Freq.

Enhances the BOW model by weighting terms based on their frequency in the document (TF) and inversely to their freq across all documents (IDF).

Word Embeddings:

Represent words as dense vectors in a continuous vector space, capturing semantic relationships b/w words.

Document representation using word embeddings involves averaging word vectors to create a representation of the entire document.

Graph-Based Representation:

Represent documents and their relationships using graph structures, where nodes represent documents and edges represent semantic relationships.

Considerations:

→ Dimensionality
→ Sparsity
⇒ Semantic Accuracy
→ Domain Specific Adaptation.

Appls: → IR → Text classification.
→ IE → Recommendation systems

Index Structures:-
Index structures are fundamental to efficient IR systems, providing organized and quick access to documents based on user queries. These structures optimize the speed of retrieval operations by pre-processing docs into indexed formats.

Inverted Index:
An inverted index is a data structure that maps each unique term in a corpus to a list of documents (or postings where the term appears. Widely used in search engines and retrieval systems to quickly identify documents relevant to a user query based on the presence of query terms.

Forward Index:
The forward index maps each document in a collection to the terms it contain, along with their positions within the document.

Full-Text Index:
Combines features of both inverted and forward indices, storing detailed information about terms, their occurences and positions in documents.

Suffix Tree/Array:
A data structure that stores all suffixes of a text string in a way that allows efficient pattern matching and substring retrieval.

Bitmap Index:
A specialized index structure that uses bitmaps,

→ to represent the presence or absence of terms in documents.

Considerations:
→ Space efficiency
→ Update efficiency
→ Query performance
→ Scalability
→ Support for Complex queries.

Applications:
→ Web search engines
→ DB systems
→ Text Mining and IR

Retrieval Models:-

Retrieval models are algms and frameworks used in IR systems to rank and retrieve documents that are relevant to users queries.

Boolean Model:

Based on set theory, where documents are either relevant (1) or non-relevant (0) to a query.
Queries are represented as Boolean expressions (AND, OR, NOT) of terms.

VSM: Vector Space Model.

Represents documents and queries as vectors in a high-dimensional space, where each dimension corresponds to a term or feature.
Scores documents based on cosine similarity b/w query and document vectors.
Widely used in search engines and text retrieval systems due to its flexibility.

Probabilistic Models: BM25

Computes the probability that a document is relevant to a query based on probabilistic assumptions.
Known for robust performance in handling relevance and retrieval effectiveness.

Language Models :-

Treats documents and queries as distributions of terms in language models.

Ranks documents based on the likelihood score or divergence from the query language model.

Considerations :

- → Scalability
- → Adaptability
- → Revevance and Ranking
- → Interpretability.

Applications :

- → Web search Engine
- → Enterprise search
- → Question Answering Systems
- → Text Mining and IE.

Document a Priori Models :-

The term "a priori models" in the content of IR typically refers to probabilistic models that make assumptions about the distribution of terms in document lengths in a collection.

Poisson Model :

The Poisson model is probabilistic model used in IR to estimate the likelihood of a document being relevant to a query.

'λ' represents the expected number of occurences of a term in a document.

Characteristics : Term occurences
Document length
Relevance Estimation.

Geometric Model :

The geometric model is another probabilistic approach used in IR, assuming that term occurences in a document follow a geometric distribution.

Characteristics: Term occurrences
Probability Estimation
Document Ranking.

Applications:
- IR systems
- Relevance Feedback.

Challenges:
- → Assumption validity
- → Parameters Estimation
- → Integration with Retrieval models.

CLIR:-

CLIR stands for Cross-Lingual IR, which is a specialized area of IR focused on retrieving relevant information from documents written in different languages than the language of the user query.

CLIR addresses the challenge of accessing and understanding multilingual information resources, facilitating effective information retrieval across language barriers.

Language Translation:

Automatic Translation: Utilizes MT techniques to translate user queries from one language into the language of the docs being searched.

Indexing Strategies: Develops methods to index documents in multiple languages, enabling efficient retrieval based on translated queries.

Language Identification: Determines the language of docs to ensure accurate indexing and retrieval.

Term Mapping: Establishes mappings b/w terms in different languages to bridge vocabulary gaps and improve retrieval accuracy.

Evaluation Metrics:

CLIR Evaluation: Adapts traditional IR evaluation metrics to assess the effectiveness of CLIR systems in retrieving relevant information across languages.

Challenges:
- → Translation quality
- → Multilingual Indexing
- → Resource availability.

Techniques:
- → MT integration
- → Cross-lingual embedding
- → Transfer learning.

Applications:
- → Multilingual search engines
- → Global Information Access
- → Multinational Organizations.

Machine Translation:

MT refers to the automated process of translating text or speech from one NL to another using computational algms and models.

MT systems aim to replicate human translation capabilities, enabling communication and understanding across different languages.

Rule-Based MT:

Uses linguistic rules and dictionaries to translate text from a source lang to target language.

Requires explicit grammar rules, syntactic analysis, and lexical databases.

SMT: Statistical MT

Translates based on statistical models trained on large bilingual corpora. Relies on statistical relationships btw words and phrases in source and target language.

Neural MT:
uses NN, typically seq2seq models with attention mechanisms, to learn mappings btw source to target languages.

Applications:
→ Global Communication
→ Content Localization.
→ Cross-lingual IR.
→ Multilingual Customer support.

Challenges:
→ Translation quality
→ Domain Adaptation
→ Low-Resource langs
→ Ambiguity & Context.

Interlingual Document Representation:
It refers to a method of representing docs in a way that facilitates understanding and processing across multiple languages.
Interlingual representation aims to capture the underlying meaning (or content of docs in a language-independent manner.

Semantic Representation:
Represent documents at a semantic-level that transcends language specific details.

Cross-lingual Embeddings:
Embedding words (or docs into a common vector space where similar meanings are close together.

Ontology-Based Representation:
Use structured ontologies or knowledge graphs to represent document content in a language-independent format.

Aligned Corpora and Parallel Data:
Align and integrate information from parallel corpora to create Interlingual representations.

Benefits and Apps:

→ Cross lingual IR
→ Multilingual Text Mining
→ Cross-lingual Knowledge Discovery.

MLIR:-

MLIR stands for Multilingual IR, which is a specialized area within IR focused on retrieving relevant information from docs that are written in multiple languages.

Cross-lingual Indexing:

Index documents written in multiple languages to facilitate efficient retrieval based on user queries. &

Query Translation and Reformulation:

Translate user queries from one language into multiple languages to retrieve relevant docs.

Multilingual Relevance Ranking:

Rank documents in multiple languages based on their relevance to a user query expressed in any of those languages.

Evaluation Metrics and Benchmarking:

Develop evaluation methodologies and metrics to assess the effectiveness of MLIR systems.

Applications:

→ Global Information Access
→ Cross-cultural Collaboration
→ Multinational Organizations.

Language Identification:-

Language Identification, is also known as language

detection or language guessing, is the process of automatically determining the NL. of a given text or document.

Statistical Methods:

Character N-grams: Analyze the freq. of character sequences (n-grams) in the text, comparing them against lang specific profiles.

Word-N-grams: Use the freq. of word sequences to infer the lang of the text.

ML Approaches:

SL: Train classifiers on labeled datasets where each text sample is annotated with the language.

DL: Use NN. to automatically learn language features from text representations.

Dictionary-Based Methods:

Dictionary Look-up: Compare words or phrases in the text against dictionaries or lexicons of known words in different languages.

Language Specific Patterns:

Identify language-specific patterns or linguistic features to distinguish blw languages.

Index Construction:

Document Preprocessing:

→ Tokenization
→ Normalization
→ Stopword Removal

Indexing Structures:

→ Inverted Index
→ Language Specific Indexes

Term weighting and scoring:

→ TF-IDF
→ Language Aware Scoring.

Cross-lingual linking:

→ Term Translation
→ Conceptual Mapping

Index Optimization:

→ Compression
→ Efficiency

Query Translation:

Query translation refers to the process of translating a user query from one language into one or more target languages to retrieve relevant documents or information.

Aggregation Models:

Aggregation models are used to combine predictions or outputs from multiple sources or models into a unified decision or result.
These models are essential for improving accuracy, robustness and reliability across various tasks.

Ensemble Learning:

Ensemble learning involve combining multiple individual models to produce a stronger model that typically performs better than any individual model alone.

Voting: Combining predictions by majority voting or weighted voting from multiple classifiers.

Bagging (Bootstrap Aggregation): Training multiple instances of the same model on different subsets of the data and aggregating their predictions.

Boosting: sequentially building models where each subsequent model focuses on correcting errors made by the previous ones.

Stacking:

Stacking is a meta-learning technique where multiple base models are trained on the same data, and a meta-model learns how to combine their predictions effectively.

Federated Learning:

Federated learning aggregates model updates from multiple decentralized devices or servers without centralizing data.

Applications:

→ classification and Regression
→ Anomaly Detection
→ Crowd Sourcing.

Challenges:

→ Model Diversity
→ Computational Complexity
→ Interpretability.

Tools, Software and Resources:-

ML and DS Tools:

Python Libraries: Numpy, Pandas → Numerical computation and data manipulation

scikit-learn → data mining and data analysis

TensorFlow, PyTorch → DL frameworks for building and training NN.

Keras → High level NN API.

R-libraries: caret → training & evaluating ML models.

tidyverse → designed for DS.

mlr3 → ML in R.

Visualization and Analysis:

Matplotlib, seaborn: plotting libraries for creating visualizations in Python.

ggplot2: Data visualization in R.

Tableau, Power BI: Business Intelligence tools.

NLP:

NLP libraries:

NLTK (NL Toolkit): Programs for NLP tasks in Py

Scapy: efficient text processing.

DL for NLP:

→ Hugging Face Transformers

→ Allen NLP

→ BERT, GPT-3.

Text Annotation and Labeling:

Label Studio: open-source data labeling tool for ML projects.

Prodigy: Active learning annotation tool for NLP and computer vision tasks.

IR and search engines:

ElasticSearch: Distributed, search and analytics based on Lucene.

Apache Solr: Open source platform built on Apache Lucene

Lucene: Java based search library used in search engine

Terrier: Modular platform for the rapid dev of large scale IR app'ns.

Model Deployment:

Docker, Kubernetes: Containerization and orchestration platforms for deploying and scaling apps.

Torch Serve: Framework for serving ML models in production.

Cloud Platforms:

AWS: EC services including SageMaker for ML

Google Cloud AI Platform: Managed services for building and deploying ML models.

Open Data Portals:

→ Kaggle: Platform for DS competitions and data sets.

→ UCI ML Repository

Miscellaneous Tools and Resources:

→ Version Control

→ Jupyter Notebooks

→ Community Forums.

→ Online Courses.

Multilingual Automatic Summarization - Introduction, Approaches to summarization, Evaluation, How to Build a Summarizer, Competitions and Datasets.

Introduction :-

Multilingual automatic summarization is a field within NLP that aims to automatically condense and extract essential information from texts written in multiple languages.

The goal is to create concise summaries that preserve the key points of the original content across different languages, allowing users to quickly grasp the main ideas without needing to read the entire text.

Language Diversity:

Dealing with the nuances and syntactic variations across different languages.

Cross-lingual Alignment:

Ensuring that summaries accurately reflect the original content's meaning across languages.

Resource Availability:

Availability of parallel corpora and bilingual dictionaries for training and evaluation.

Techniques used in multilingual summarization often involve leveraging ML models, such as neural n/wor and transformer-based architectur like BERT and GPT, which have shown effectiveness in understanding and generating summaries in multiple languages.

These models are trained on multilingual datasets and can handle diverse linguistic patterns to produce coherent and information summaries.

Approaches to Summarization:-

There are several approaches to automatic summarization.

Extractive Summarization:

Extractive summarization involves selecting and extracting important sentences or phrases directly from the original text to form a summary.

Process: It typically involves ranking sentences based on features like relevance, importance, or similarity to the overall document.

Abstractive Summarization:

Abstractive summarization involves generating new sentences that convey the most important information from the original text in a more concise manner.

Process: It uses advanced NLP techniques, often involving DL models like transformers, to interpret and generate new text based on the content of the original doc.

Hybrid Approach:

It combines elements of both extractive and abstractive techniques to leverage their respective strengths.

The Classics:-

Frequency-Based Methods:

These methods identify the most frequent words or phrases in a text as indicators of importance.

Techniques: Like TF-IDF measures how important a word is to a document relative to a collection of docs.

Graph-Based Methods:

These methods represent the text as a graph where sentences are nodes and relationships blw sentences are edges.

Techniques: Algms such as PageRank are adapted to rank sentences based on their centrality and importance in the graph.

Latent Semantic Analysis: LSA

LSA is a mathematical technique that analyzes relationships blw terms and concepts in a collection of texts.

Techniques: It creates a semantic space where words and documents that are semantically similar are located close to each other.

Naive Bayes Classifier:

This classifier calculates the probability of a sentence being relevant to the summary based on the occurrence of words in the sentence.

Graph-Based Approaches:-

Graph-based approaches in automatic summarization leverage the representation of text as a graph, where nodes represent units of text and edges denote relationships blw these units.

Sentence Extraction Using Graph Algms:

Represent sentences as nodes in a graph and create edges blw them based on similarity measures.

eg: cosine similarity, semantic similarity.

Techniques: Algms like Text-Rank and LexRank adapt the pageRank algm from Google to rank sentences based on their

centrality and importance within the graph.

Word Bigraphs and Keyword Extractions:
Construct a graph where nodes represent words or phrases, and edges represent co-occurrences or semantic relationships b/w them.

Document Clustering and Summarization:
Represent documents as nodes in a graph and connect them based on similarity measures. Clustering algms. like k-means or hierarchical clustering can be applied to group related documents together within the graph.

Multilingual Summarization:
Multilingual summarization involves the process of automatically generating summaries from texts that are written in multiple languages.

Challenges:
→ Language Diversity
→ Cross-lingual Alignment
→ Resource Availability.

Applications:
→ News Aggregation
→ Cross-cultural communication
→ Multilingual Document Management.

Evaluation:-
Evaluation of summarization systems, whether monolingual or multilingual is essential to assess their effectiveness and quality.

Precision and Recall:
Adapted from IR, these metrics how well the summary covers important info (recall), while avoiding irrelevant details (precision.

F-measure: Harmonic mean of precision and recall, providing a balanced measure of summarization quality.

ROUGE: Recall Oriented Understudy for Gisting Evaluation measures overlap b/w system generated summaries and reference (human-generated) summaries.

BLEU: Bilingual Evaluation understudy.
Originally designed for MT. It evaluates the overlap of n-grams b/w system generated summaries and reference summaries.

Semantic similarity:
Assess the semantic similarity b/w system generated summaries and reference summaries using embeddings or semantic models.

Entity-Based Evaluation:
Focuses on how well named entities are identified and preserved in the summary.

Cross-lingual Metrics:
Adapt existing monolingual metrics like ROUGE for cross-lingual evaluation, considering language specific characteristics and translation quality.

Domain Relevance:
Evaluate how well summaries capture domain-specific terminology, concepts, and nuances relevant to the source text.

Choosing the appropriate evaluation metrics depends on factors such as the type of summarization task, available resources, and the desired level of detail and coverage in evaluating system performance.

Manual Evaluation Methodologies:

Manual evaluation methodologies for summarization involve human judges assessing the quality of summaries based on various criteria.

Quality Criteria:

Informativeness: How well does the summary convey the main points and essential information of the original text.

- → Conciseness
- → Relevance
- → Coherence
- → Fluency

Scoring Scales:

Likert Scale: Judges rate summaries on a scale for each criterion, providing a quantitative assessment of summary quality.

Binary Judgements: Judges decide whether a summary meets predefined criteria (eg. Yes/No).

Guidelines and Instructions:

Annotation Guidelines: Provide detailed instructions to judges on how to evaluate summaries, including examples and explanations of scoring criteria.

IAA: Inter Annotator Agreement.

Consistency check: Measure agreement among multiple judges (annotators) evaluating the same summaries.

Sample selection:

Random sampling: Select a representative sample of summaries from the evaluation corpus to minimize bias and ensure comprehensive coverage.

Balanced sampling: Ensure an equal distribution of summaries across different conditions, for fair evaluation.

Automated Evaluation Methods:-

Automated evaluation methods for summarization are used to assess the quality of summaries using computational metrics rather than relying solely on human judges.

These methods are generally faster and more scalable than manual evaluation, making them useful for large-scale experiments and real-time evaluation in automated systems.

Semantic Based Metrics:

METEOR: Metric for Evaluation of Translation with Explicit ORdering.

It evaluates the quality of summaries by considering not only n-gram overlap but also semantic similarity based on synonymy, stemming and word order.

BERTScore: It uses contextual embeddings from BERT (Bidirectional Encoder Representations from Transformers) to compute similarity scores b/w system generated and reference summaries.

Overlap-Based Metrics:

→ ROUGE

→ BLEU

Evaluation Frameworks:

PyRouge: A python library for computing ROUGE scores, widely used in summarization research and development.

NLTK: Provides tools and libraries for implementing and evaluating summarization systems using various automated metrics.

Cross-lingual Embeddings Based metrics:

LASER: Language Agnostic Sentence Representation
Utilizes multilingual embeddings to evaluate semantic similarity b/w system-generated summaries and reference summaries across different languages.

MUSE: Multilingual Universal Sentence Encoder.
Another framework that provides embeddings for cross-lingual evaluation, measuring how well summaries capture the semantic content regardless of content.

How to Build a Summarizer:-
Building a summarizer involves several key steps and considerations, depending on whether you are aiming for extractive or abstractive summarization.

Define the scope and Requirements:
clarify the goal of your summarizer.
eg: news articles, scientific papers.

Data collection and Preprocessing:
Data source: Gather a dataset of documentation suitable for your summarization task.

Text Preprocessing: clean and preprocess the text data.
- → Tokenization
- → Sentence segmentation
- → Removing stopwords, punctuation, and non-informative elements.

Feature Extraction:
Compute TF-IDF scores for words in each doc to identify important words and phrases.
Represent each sentences as a vector using TF-IDF or word embeddings. to capture semantic meaning.

Summarization Algm:

For extractive summarization
→ Graph based methods.
TextRank Algm, LexRank Algm
→ ML Approaches.
SVM, Random Forests.

For abstractive summarization
→ Seq2seq models.
BERT & GPT

Deployment and Iteration:

Integration: Integrate your summarizer into an appn or platform where it can process new documents or texts.

Feedback loop: Gather feedback from users and evaluate performance continuously to refine and improve your summarization model.

Considerations:

→ Scalability
→ Quality vs Speed.
→ Domain Adaptation.

Ingredients (Components):

Data Corpus: Gather a dataset of documents or texts suitable for your summarization task.

Text Preprocessing Tools:

Use libraries such as NLTK, Spacy or TF for tasks like tokenization, sentence segmentation, and removing stopwords and punctuation.

Devices:

Python Libraries: NLTK, Spacy, TF, PyTorch for NLP tasks and ML

Pre-trained Models: BERT, GPT & TF

Instructions:

→ Data Preparation
→ Feature Extraction
→ Summarization model Development
  • Extractive summarization
  • Abstractive summarization

Competitions:-

TAC - Text Analysis Conference
SemEval - Semantic Evaluation
DUC - Document Understanding Conference
SCAI - Summarization and Creative AI

→ Kaggle
→ Codelab

ACL → Association for Computational Linguistics.
NAACL - North American Chapter of the Association for Computational Linguistics.
EMNLP - Conference on Empirical Methods in NLP
TREC - Text Retrieval Conference.

Datasets:-

S2ORC - Semantic Scholar Open Research Corpus
XSum - Extreme Summarization.
Covid-19 Open Research Data Set (CORD-19).