**Extractive Summarization**

1. Frequency Based

|  |
| --- |
| Method |
| The method first tokenizes the text body into individual sentences and then tokenizes each sentence into individual words. Stopwords are removed from the words, and then the frequency of each word is computed.  Then, the score for each sentence is computed based on the frequency of its words. The sentence score is simply the sum of frequencies of its words, representing how important or informative the sentence is. And the top n sentences with the highest scores are selected to form the summary. Finally, the summary is generated by joining the selected sentences together. |
| Advantages |
| * Simple and efficient to implement, making it easy to use in various applications. * Fast processing, as it doesn’t require deep understanding, just pattern recognition based on frequency. * Retains important information by focusing on high-frequency words, so it’s likely to capture the main points. |
| Disadvantages |
| * Lacks context and can sometimes miss meaning in the text because it’s based purely on frequency rather than meaning. * Extracted summaries may feel choppy or disjointed, as they’re literal sentences from the text without any rewriting. * Doesn’t capture the full meaning or nuances, which may lead to missed or misunderstood information if the text is complex. |

1. TF-IDF Approach

|  |
| --- |
| Method |
| We start by tokenizing the text into individual sentences and constructing a TF-IDF matrix, where each row represents a sentence and each column represents the importance of terms in the sentences. The function tokenizes the text into individual sentences, creates a TF-IDF matrix using the TfidfVectorizer class, and computes the cosine similarity between each sentence and the document using the cosine similarity function. By selecting the top n sentences with the highest similarity scores, the function forms a coherent summary, which is returned as a single string. For this approach we used N-grams of range 3-4. |
| Advantages |
| * This method is effective in capturing the main idea of the text by focusing on the most relevant sentences based on their contextual importance. * The use of TF-IDF helps in weighing the significance of terms, which can lead to more meaningful summaries compared to simpler methods. * Cosine similarity allows for a comparison of sentence relevance, making the summarization process efficient. |
| Disadvantages |
| * Lacks context and can sometimes miss meaning in the text because it’s based purely on frequency rather than meaning. * Extracted summaries may feel choppy or disjointed, as they’re literal sentences from the text without any rewriting. * Doesn’t capture the full meaning or nuances, which may lead to missed or misunderstood information if the text is complex. |

1. Count Vectorizer

|  |
| --- |
| Method |
| Start by tokenizing the input text into individual sentences. It constructs a count matrix that reflects the frequency of each word in the sentences, excluding common stopwords. Next, we calculate the cosine similarity between the sentences to determine their relevance. By selecting the top N sentences with the highest similarity scores, it assembles these into a coherent summary, which is returned as a single string. |
| Advantages |
| * Simple to implement and user-friendly. * Effective for identifying key sentences based on word frequency. * Cosine similarity efficiently evaluates sentence relevance. |
| Disadvantages |
| * May produce less meaningful summaries due to a lack of word significance weighting. * Can overlook important terms that are infrequent. * Summaries may lack coherence, as sentence selection is based solely on frequency. |

1. Luhn Summarizer

|  |
| --- |
| Method |
| Applies the concept of frequency analysis, where the importance of a sentence is determined by the frequency of significant words within it. The algorithm identifies words that are most relevant to the topic of the text by filterin gout some common stop words and then ranks sentences. The Luhn Summarizer is effective for extracting key sentences from a document. |
| Advantages |
| * Focuses on significant terms, enhancing the relevance of selected sentences. * Simple and efficient, making it suitable for various text types. * Effective at maintaining the overall meaning of the original text. |
| Disadvantages |
| * May miss context or nuances, as it relies heavily on word frequency. * Can produce summaries that feel disjointed if selected sentences are not well connected. * Limited in handling complex sentence structures or advanced topics. |

1. Edmundson Summarizer

|  |
| --- |
| Method |
| Uses bonus words, stigma words, and null words. These enable the algorithm to emphasize or de-emphasize those words in the summarized text. |
| Advantages |
| * Focuses on significant terms, enhancing the relevance of selected sentences. * Simple and efficient, making it suitable for various text types. * Effective at maintaining the overall meaning of the original text. |
| Disadvantages |
| * May miss context or nuances, as it relies heavily on word frequency. * Can produce summaries that feel disjointed if selected sentences are not well connected. * Limited in handling complex sentence structures or advanced topics. |

1. LSA Summarizer

|  |
| --- |
| Method |
| Works by identifying patterns and relationships between texts, rather than soley rely on frequency analysis.  Latent Semantic Analysis (LSA) is a type of natural language processing that looks at how documents and the terms they contain are related. It searches unstructured data for hidden relationships between terms and concepts using singular value decomposition, a mathematical technique. LSA summarizer generates more contextually accurate summaries by understanding the meaning and context of the input text. |
| Advantages |
| * Produces contextually accurate summaries by understanding meanings and relationships. * Capable of identifying underlying themes, leading to more coherent results. * Reduces the risk of losing important context. |
| Disadvantages |
| * More complex and computationally intensive than frequency-based methods, which may result in slower performance. * Requires more resources to perform effectively, making it less accessible for quick implementations. * May still miss specific details if the relationships between words are not clearly defined in the text. |

1. TextRank

|  |
| --- |
| Method |
| TextRank builds a graph where sentences are nodes, and edges are created based on sentence similarity. By ranking sentences according to their connections and similarity to other sentences, the TextRank summarizer identifies the most important ones to include in the summary. This approach allows TextRank to capture the structure and key ideas of the text more effectively, often resulting in summaries that are both coherent and informative. |
| Advantages |
| * Generates coherent summaries by connecting related sentences through similarity. * Captures important ideas without heavily relying on frequency, making it effective for complex texts. * Simple to implement, making it versatile for many types of text. |
| Disadvantages |
| * Can be computationally demanding, especially for large texts. * May struggle with very short or highly specialized texts where sentence similarity is harder to establish. |

1. KL Sum

|  |
| --- |
| Method |
| It is a sentence selection algorithm where a target length for the summary is fixed. It is a method that greedily adds sentences to a summary so long as it decreases the KL Divergence. It selects sentences based on the similarity of word distribution to the original text. It aims to lower the KL-divergence criteria. It uses a greedy optimization approach and keeps adding sentences till the KL-divergence decreases. |
| Advantages |
| * By selecting sentences that closely match the original text's word distribution, KL-Sum ensures that the summary retains the primary information and overall topic coverage. * It’s particularly useful when capturing the key points without deviating from the original text’s vocabulary. |
| Disadvantages |
| * Can miss context, as it doesn’t account for the relationship between sentences. * May struggle with highly varied texts where consistent word distribution is harder to achieve. |

**Abstractive Summarization**

1. Google T5

|  |
| --- |
| Method |
| T5 is a transformer-based language model that has achieved state-of-the-art results on various NLP tasks, including text summarization. T5 is an encoder-decoder model that is pre-trained on a mixture of unsupervised and supervised tasks in a multi-task setting. |
| Advantages |
| * The ability of T5 to perform multiple NLP tasks by simply changing the prefix of the input is a significant advantage. Additionally, its performance on various tasks has made it one of the most promising approaches for NLP applications. * The model is pre-trained on a mixture of unsupervised and supervised tasks, it has the potential to generalize well to new tasks. |
| Disadvantages |
| * T5 requires a substantial amount of computational resources for training and inference. * It might not achieve state-of-the-art performance on large-scale NLP benchmarks, and LLM chatbots might do better than it. |

1. ChatGPT

|  |
| --- |
| Method |
| T5 is a large-scale transformer-based language model that has achieved state-of-the-art results on various NLP tasks, including text summarization. T5 is an encoder-decoder model that is pre-trained on a mixture of unsupervised and supervised tasks in a multi-task setting. |
| Advantages |
| * The ability of T5 to perform multiple NLP tasks by simply changing the prefix of the input is a significant advantage. Additionally, its performance on various tasks has made it one of the most promising approaches for NLP applications. * The model is pre-trained on a mixture of unsupervised and supervised tasks, it has the potential to generalize well to new tasks. |
| Disadvantages |
| * T5 requires a substantial amount of computational resources for training and inference. * It might not achieve state-of-the-art performance on large-scale NLP benchmarks, and LLM chatbots might do better than it. |

1. DistilBERT

|  |
| --- |
| Method |
| The DistilBERT summarization method uses a smaller, distilled version of the BERT model, specifically optimized for efficiency and speed without sacrificing much performance. It uses a transformer-based architecture to process input text, capturing contextual relationships and meanings, and generating concise summaries while retaining important information. DistilBERT is designed to be lightweight and faster than BERT, making it more accessible for real-time applications. |
| Advantages |
| * Fast and efficient, suitable for real-time summarization on limited resources. * Retains much of BERT's performance in understanding context and nuance, delivering coherent summaries. |
| Disadvantages |
| * Some reduction in accuracy compared to full BERT, especially with highly complex texts. * May still require significant computational resources compared to simpler summarization models. |

1. Pegasus

|  |
| --- |
| Method |
| PEGASUS is trained to generate summaries by masking out entire sentences during pre-training, forcing it to "reconstruct" the masked parts, which simulates real-world summarization. By training this way, PEGASUS learns to generate coherent, concise summaries that capture the essence of the original text in a human-like manner. |
| Advantages |
| * Excellent for abstractive summarization, generating summaries that are closer to human-written ones. * Efficient in capturing context, tone, and overall meaning of the original text. |
| Disadvantages |
| * Computationally intensive, often requiring specialized hardware, making it less accessible for real-time summarization. * May struggle with very short texts or content with a limited structure. |

1. LED (Longformer Encoder-Decoder)

|  |
| --- |
| Method |
| The Longformer Encoder-Decoder (LED) model uses a sparse attention mechanism that enables it to process long documents efficiently. It combines sliding window attention for local context with global attention on key tokens, allowing it to handle large inputs while maintaining coherence in summaries. |
| Advantages |
| * Highly effective for long-document summarization, processing much larger text inputs than typical models. * Efficient in memory usage, thanks to sparse attention, allowing for scalable summarization on long texts. |
| Disadvantages |
| * Computationally intensive, often requiring more hardware requirements, making it less accessible for real-time summarization. * May struggle with very short texts or content with a limited structure. |