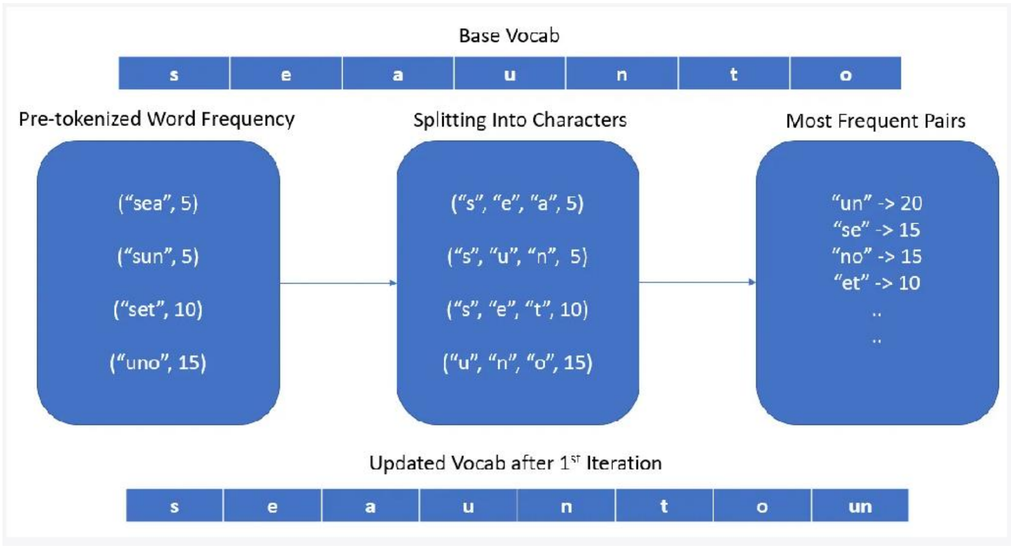
Tokenization

Tokenization in information retrieval is the process of breaking down a piece of text or document into smaller units, called tokens. These tokens are typically words or groups of words that represent the basic unit of meaning in a language.
Tokenization is an important step in information retrieval, as it allows the retrieval system to process the text more efficiently and accurately. When a document is tokenized, each token is indexed by the retrieval system, making it easier to search forspecific words or phrases.

Byte-Pair Encoding (BPE)
Byte-Pair Encoding first requires the input text to be pre-tokenized — which can be assimple as whitespace tokenization or you could employ a rule-based tokenizer such as SpaCy for performing the pre-tokenization step.

Now we form a base vocabulary which is nothing but a collection of all unique characters present in the corpus. We also calculate the frequency of each token andrepresent each token as a list of individual characters from base vocabulary.

Now merging begins. We keep adding tokens to our base vocab as long as the maximum size is not breached on the basis of following criteria — the pair of tokens occurring most number of times is merged and introduced as a new token. This step isrepeated until we reach the configured maximum vocab size.



Objective behind the algorithm

BPE was originally a data compression algorithm that you use to find the best way to represent data by identifying the common byte pairs. We now use it in NLP to find thebest representation of text using the smallest number of tokens.

BPE ensures that the most common words are represented in the vocabulary as a single token while the rare words are broken down into two or more subword tokens and this is in agreement with what a subword-based tokenization algorithm does.

Functionality and ImplementationFunctionality:

Byte-Pair Encoding (BPE) is a compression algorithm used in Natural Language Processing (NLP) to represent large vocabulary with a small set of subword units.

It was introduced by Sennrich et al. in 2016 and has been widely used in various NLP tasks such as machine translation, text classification, and text generation.

The basic idea of BPE is to iteratively merge the most frequent pair of consecutive bytes or characters in a text corpus until a predefined vocabulary size is reached. The resulting subword units can be used to represent the original text in a more compact and efficient way.

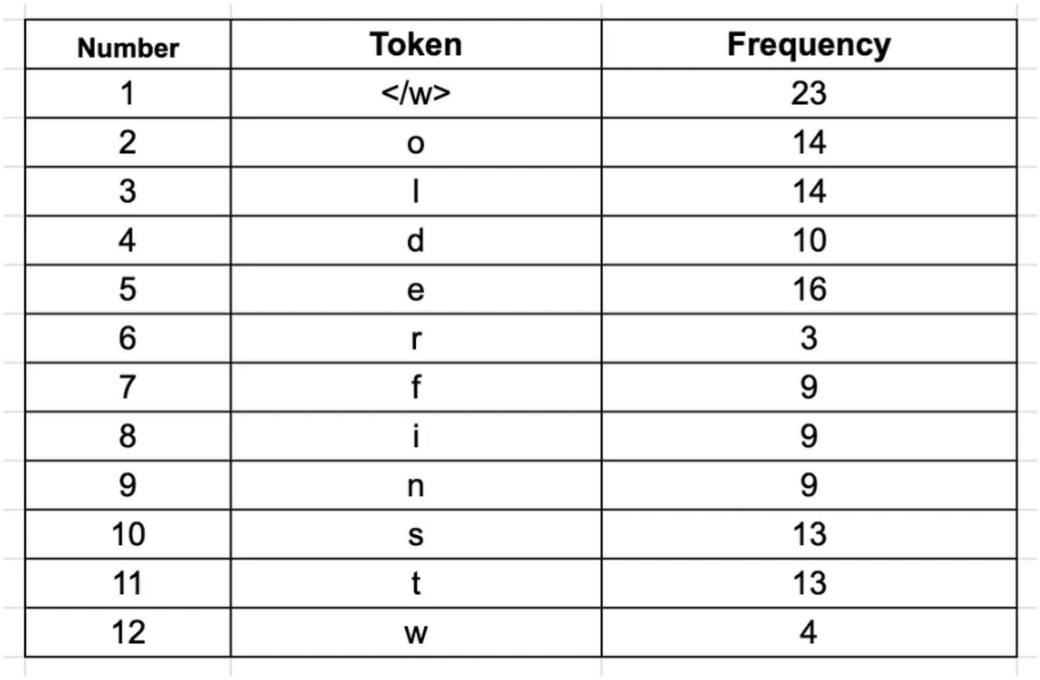
Steps involved in BPE:

Initialize the vocabulary with all the bytes or characters in the text corpus

1. Calculate the frequency of each byte or character in the text corpus.
2. Repeat the following steps until the desired vocabulary size is reached:
3. Find the most frequent pair of consecutive bytes or charactersin the text corpus
4. Merge the pair to create a new subword unit.
5. Update the frequency counts of all the bytes or characters thatcontain the merged pair.
6. Add the new subword unit to the vocabulary.
7. Represent the text corpus using the subword units in the vocabulary.

Illustration with Example:

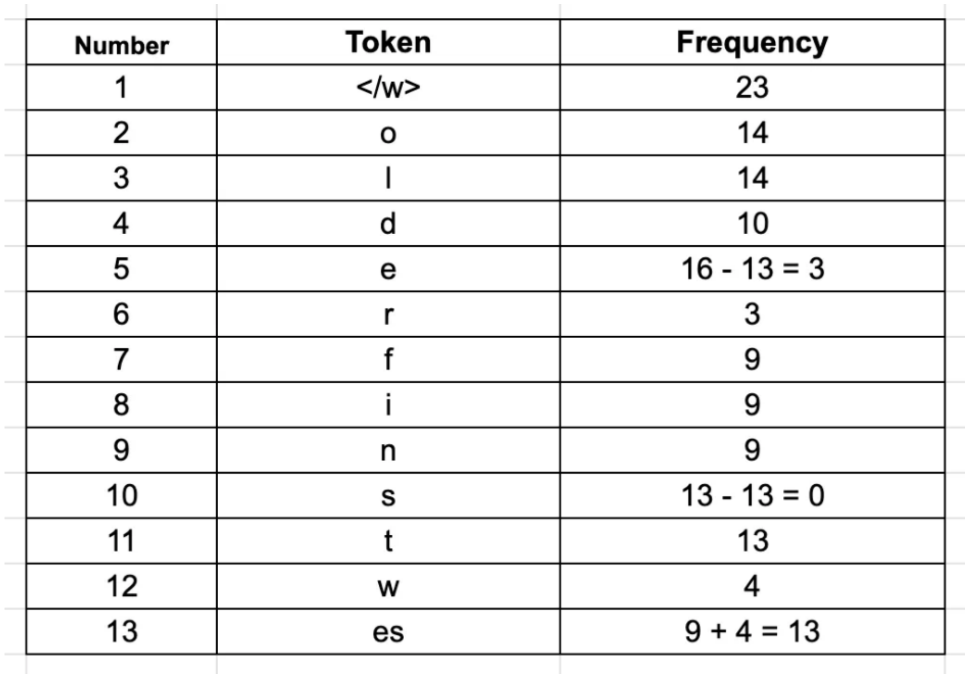
Suppose we have a corpus that has the words (after pre-tokenization based on space) — old,older, highest, and lowest and we count the frequency of occurrence of these words in the corpus. Suppose the frequency of these words is as follows:
{“old”: 7, “older”: 3, “finest”: 9, “lowest”: 4}
Let us add a special end token “</w>” at the end of each word.
{“old</w>”: 7, “older</w>”: 3, “finest</w>”: 9, “lowest</w>”: 4}
The “</w>” token at the end of each word is added to identify a word boundary so that thealgorithm knows where each word ends. This helps the algorithm to look through each character and find the highest frequency character pairing.
Moving on next, we will split each word into characters and count their occurrence. Theinitial tokens will be all the characters and the “</w>” token.



Number	Token	Frequency
1	</w>	23
2	o	14
3	l	14
4	d	10
5	e	16
6	r	3
7	f	9
8	i	9
9	n	9
10	s	13
11	t	13
12	w	4

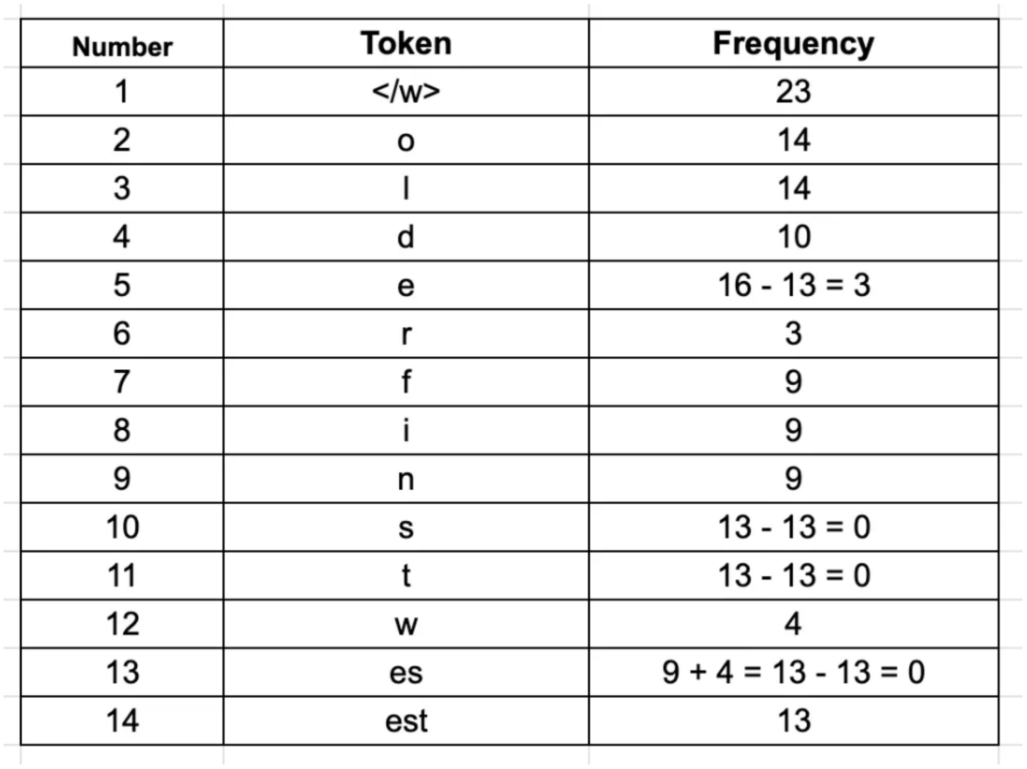
Iteration 1: We will start with the second most common token which is “e”. The most common byte pair in our corpus with “e” is “e” and “s” (in the words finest and lowest) which occurred 9 + 4 = 13 times. We merge them to form a new token “es” and note down itsfrequency as 13. We will also reduce the count 13 from the individual tokens (“e” and “s”).

This will let us know about the leftover “e” or “s” tokens. We can see that “s” doesn’t occuralone at all and “e” occurs 3 times. Here is the updated table:



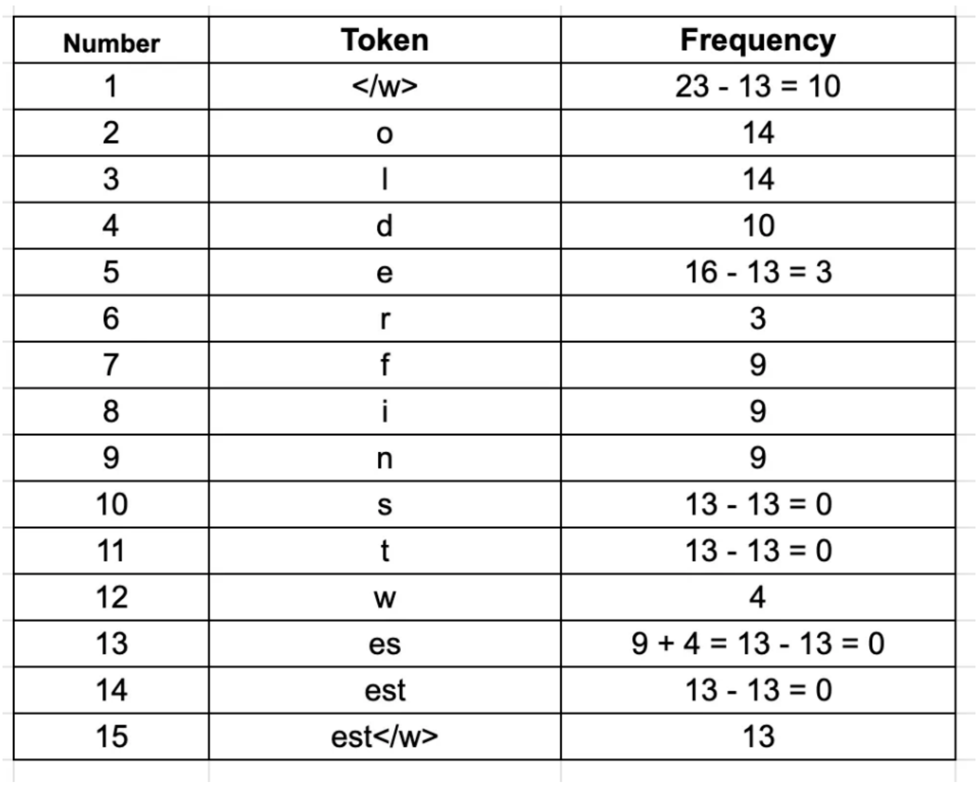
Number	Token	Frequency
1	</w>	23
2	o	14
3	l	14
4	d	10
5	e	16 - 13 = 3
6	r	3
7	f	9
8	i	9
9	n	9
10	s	13 - 13 = 0
11	t	13
12	w	4
13	es	9 + 4 = 13

Iteration 2: We will now merge the tokens “es” and “t” as they have appeared 13 times inour corpus. So, we have a new token “est” with frequency 13 and we will reduce the frequency of “es” and “t” by 13.



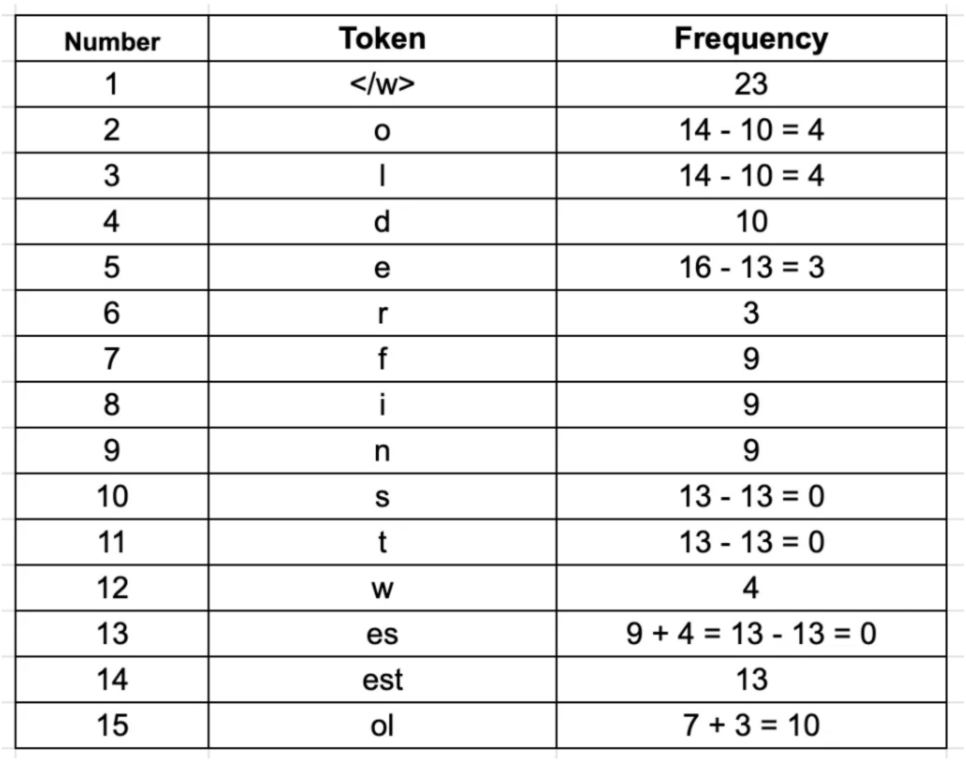
Number	Token	Frequency
1	</w>	23
2	o	14
3	l	14
4	d	10
5	e	16 - 13 = 3
6	r	3
7	f	9
8	i	9
9	n	9
10	s	13 - 13 = 0
11	t	13 - 13 = 0
12	w	4
13	es	9 + 4 = 13 - 13 = 0
14	est	13

Iteration 3: Let’s work now with the “</w>” token. We see that byte pair “est” and “</w>”occurred 13 times in our corpus.



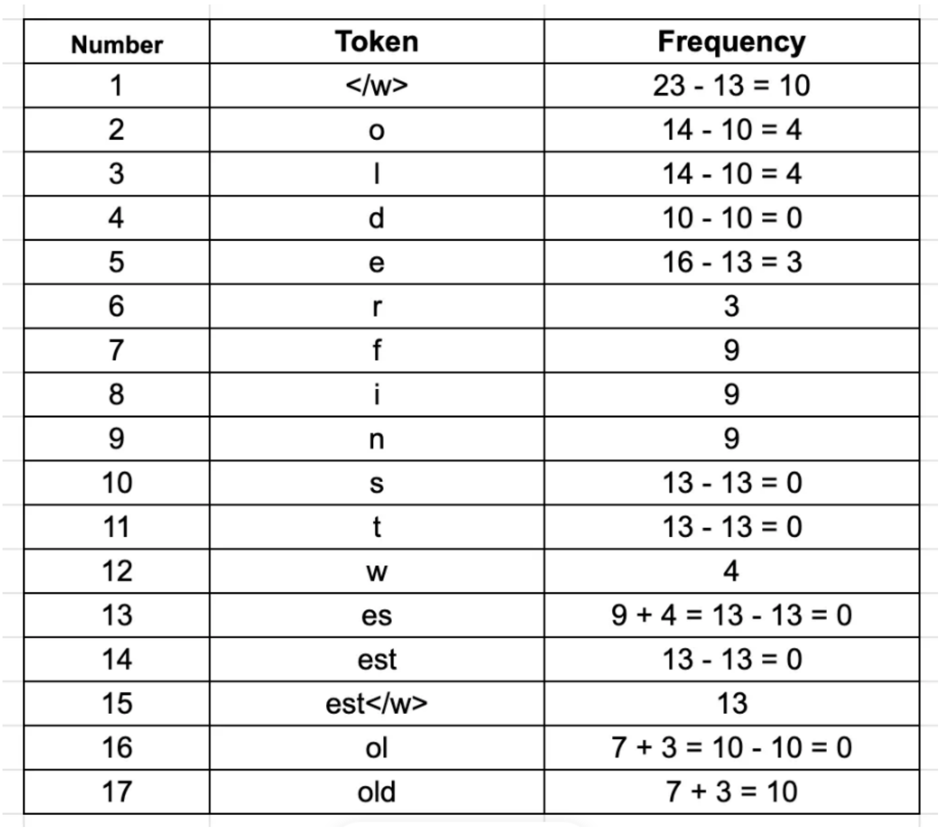
Number	Token	Frequency
1	</w>	23 - 13 = 10
2	o	14
3	l	14
4	d	10
5	e	16 - 13 = 3
6	r	3
7	f	9
8	i	9
9	n	9
10	s	13 - 13 = 0
11	t	13 - 13 = 0
12	w	4
13	es	9 + 4 = 13 - 13 = 0
14	est	13 - 13 = 0
15	est</w>	13

Iteration 4: Looking at the other tokens, we see that byte pairs “o” and “l” occurred 7 + 3 =10 times in our corpus.



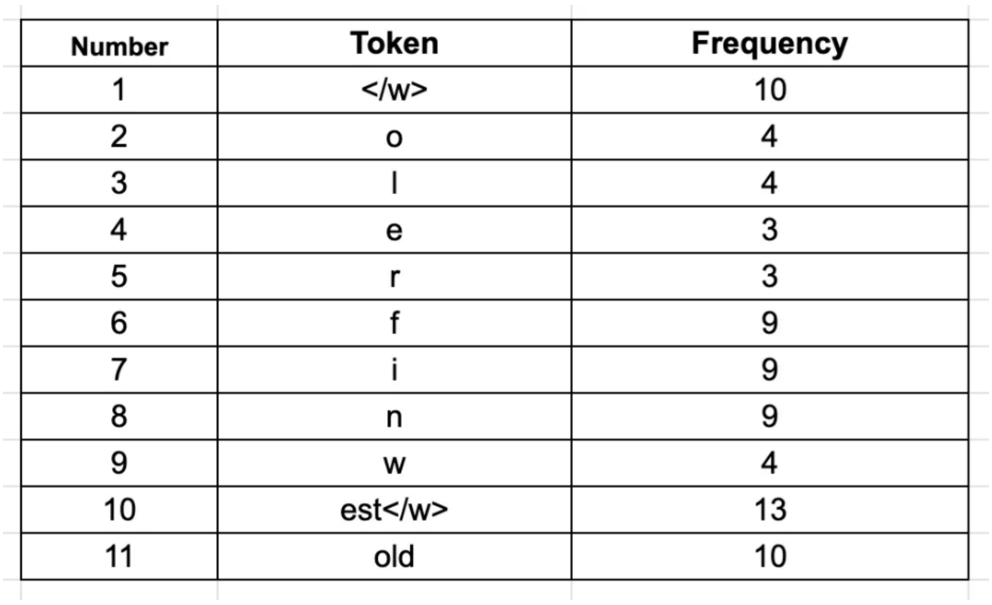
Number	Token	Frequency
1	</w>	23
2	o	14 - 10 = 4
3	l	14 - 10 = 4
4	d	10
5	e	16 - 13 = 3
6	r	3
7	f	9
8	i	9
9	n	9
10	s	13 - 13 = 0
11	t	13 - 13 = 0
12	w	4
13	es	9 + 4 = 13 - 13 = 0
14	est	13
15	ol	7 + 3 = 10

Iteration 5: We now see that byte pairs “ol” and “d” occurred 10 times in our corpus.



Number	Token	Frequency
1	</w>	23 - 13 = 10
2	o	14 - 10 = 4
3	l	14 - 10 = 4
4	d	10 - 10 = 0
5	e	16 - 13 = 3
6	r	3
7	f	9
8	i	9
9	n	9
10	s	13 - 13 = 0
11	t	13 - 13 = 0
12	w	4
13	es	9 + 4 = 13 - 13 = 0
14	est	13 - 13 = 0
15	est</w>	13
16	ol	7 + 3 = 10 - 10 = 0
17	old	7 + 3 = 10

If we now look at our table, we see that the frequency of “f”, “i”, and “n” is 9 but we havejust one word with these characters, so we are not merging them. For the sake of the simplicity of this article, let us now stop our iterations and closely look at our tokens.



Number	Token	Frequency
1	</w>	10
2	o	4
3	l	4
4	e	3
5	r	3
6	f	9
7	i	9
8	n	9
9	w	4
10	est</w>	13
11	old	10

The tokens with 0 frequency count have been removed from the table. We can now see thatthe total token count is 11, which is less than our initial count of 12. This is a small corpus but in practice, the size reduces a lot. This list of 11 tokens will serve as our vocabulary.

When we add a token, either our count increases or decreases or remains the same. In practice, the token count first increases and then decreases. The stopping criteria can be either the count of the tokens or the number of iterations.

Advantages BPE:

Byte-pair encoding offers several benefits for natural language processing andmachine learning applications:

Reduced vocabulary size: BPE helps reduce the number of unique tokens required to represent a given text, which can lead to more efficient processing and memory usage.

Handling out-of-vocabulary words: By breaking words into subword units, BPE can help algorithms handle rare or unknown words more effectively.

Improved language modeling: BPE has been shown to improve the performance of various NLP tasks, such as machine translation and text classification.

Byte pair encoding lends itself to NLP tasks due to its simplicity and speed; BPE is suitably effective for the tokenization of terms, does not require large computational overheads, and remains consistent, making it reliable.

Disadvantages BPE:

However, byte-pair encoding also has some limitations:

Fixed vocabulary: Once the BPE vocabulary is generated, it remains static, which means it may not adapt well to changes in language use or new data.

Suboptimal tokenization: BPE may sometimes produce suboptimal tokenizations that do not accurately represent the semantic structure of alanguage.

Applications :

Byte-pair encoding has found wide-ranging applications in various NLP and machinelearning tasks, including:

1. Machine translation: BPE has been used to improve translation quality by reducing the vocabulary size and handling rare words more effectively.

2. Text summarization: By using BPE tokenization, algorithms can generatemore accurate and coherent summaries of long documents.
3. Sentiment analysis: BPE can improve the performance of sentiment analysis algorithms by providing more fine-grained representations of words and phrases.