WordPiece encoding is a tokenization method commonly used in natural language processing (NLP) models like BERT. Instead of treating each word as a single unit, WordPiece breaks words down into smaller units, called "subwords," or even individual characters if necessary. This approach is particularly useful for handling rare words, spelling variations, and different forms of the same word.
Here's how WordPiece encoding works:

1. **Vocabulary Building**:
    - WordPiece starts with a basic vocabulary of individual characters and progressively combines characters or subwords into larger units based on how frequently they appear together in the training data.
    - For example, common words like "play" might remain as a single unit, but a rarer word like "playground" might be split into ["play", "##ground"], where "##" indicates that "ground" is a continuation of a previous token.
2. **Tokenization**:
    - During tokenization, each word in the input is split into the longest possible subwords that match entries in the vocabulary.
    - This means that even unfamiliar or complex words can be represented by a combination of subwords from the vocabulary. For instance, "unhappiness" might be split into ["un", "##happy", "##ness"].
3. **Handling Out-of-Vocabulary Words**:
    - By using subwords, WordPiece can handle out-of-vocabulary (OOV) words by breaking them down into smaller known pieces instead of discarding them or representing them with a single "unknown" token.
    - This allows the model to generalize better, as it can understand and process words it has never seen before by using familiar subwords.

Why Use WordPiece Encoding?

WordPiece encoding is effective because it strikes a balance between representing words as whole units (which may lead to a huge vocabulary with many rare words) and representing them as characters (which can lose the semantic information in common subword patterns). By representing words as combinations of common subwords, WordPiece improves model efficiency and helps handle the diversity of language better.

Byte Pair Encoding (BPE) is another subword tokenization method used in NLP, similar to WordPiece, that breaks words into smaller units to handle rare and out-of-vocabulary words efficiently. It was originally developed for data compression and later adapted for NLP applications. Like WordPiece, BPE aims to create a vocabulary that balances common word representations with subword segments.

How Byte Pair Encoding Works

1. **Start with a Character-Level Vocabulary**:
    - Initially, BPE starts with a vocabulary containing only individual characters. Each word is represented as a sequence of characters.
2. **Merge Frequent Character Pairs**:
    - The algorithm iteratively merges the most frequent pairs of characters (or subwords) in the corpus.
    - For example, if "th" and "he" are frequent pairs in the text, BPE will merge them to form "th" and then "the" as a single unit.
    - With each iteration, these merged pairs become new tokens in the vocabulary.
3. **Build Subword Units**:
    - As the merging process continues, the vocabulary grows to include more subwords, eventually reaching common words or frequently occurring word parts.
    - For instance, "playing" might be split into ["play", "ing"], so that "play" and "ing" can be reused in other words like "plays" and "singing."
    - This process stops when a predefined vocabulary size is reached (often based on memory or computational constraints).
4. **Tokenization**:
    - After building the BPE vocabulary, tokenization involves breaking down each word into the longest subwords that match entries in the BPE vocabulary.
    - For an uncommon word like "playground," BPE might tokenize it into ["play", "ground"], reusing known subwords and avoiding out-of-vocabulary issues.

Why Use Byte Pair Encoding?

BPE is popular because it allows NLP models to handle rare or unknown words without assigning a generic "unknown" token, preserving more information about the structure and meaning of the text. It has several advantages:

- **Efficient Vocabulary Size**: BPE keeps the vocabulary relatively compact by balancing whole words with subwords. This helps reduce memory usage and improves model training.

- **Better Handling of Out-of-Vocabulary Words**: By breaking down rare words into known subwords, BPE allows models to process words they haven't explicitly seen in training, improving generalization.
- **Language Agnosticism**: BPE works well for many languages, especially those with complex morphology or compound word structures.