One-hot Encoding

Definition : method of representing categorical data (in this case, words) as numerical vectors. Each word is represented as a vector with a length equal to the size of the vocabulary.

Vector Representation: The vector consists of all zeros except for a single element, which is set to 1. The index of the element that is set to 1 corresponds to the integer index assigned to the word during integer encoding.

One-Hot Vector: The resulting vector is called a one-hot vector.

Two step process

Integer Encoding :to perform integer encoding, which involves assigning a unique integer index to each word in the vocabulary.
Vector Creation: to create the one-hot vector for each word. The vector has a length equal to the vocabulary size

Example :Let's say we have the following vocabulary and integer encoding:

- cat: 0
- dog: 1
- bird:2

One-hot vector →
cat: [1, 0, 0]
dog: [0, 1, 0]
bird: [0, 0, 1]

Advantages of One-hot Encoding

Simple and Easy to Implement: One-hot encoding is a relatively simple and straightforward technique.

No Ordinal Relationship: One-hot encoding does not impose any ordinal relationship between the words. Each word is treated as an independent category.

Disadvantages of One-hot Encoding

High Dimensionality: One-hot encoding can lead to high-dimensional vectors, especially for large vocabularies. This can increase the memory requirements and computational complexity of machine learning models
.
Sparsity: One-hot vectors are sparse, meaning that most of the elements are zero. This can be inefficient for some machine learning algorithms.

No Semantic Meaning: One-hot encoding does not capture any semantic meaning of the words. Words that are semantically similar are treated as completely independent.

Alternatives to One-Hot Encoding

Word Embeddings (Word2Vec, GloVe, FastText): Word embeddings are a more advanced technique that learns dense, low-dimensional vector representations of words that capture their semantic meaning.

TF-IDF (Term Frequency-Inverse Document Frequency): TF-IDF is a technique that assigns weights to words based on their frequency in a document and their inverse document frequency across the corpus.

Use cases

- **Using okt(Korean morphological analyzer)**

Text : "나는 자연어 처리를 배운다"

Code :

```
from konlpy.tag import Okt

okt = Okt()
tokens = okt.morphs("나는 자연어 처리를 배운다")
print(tokens)
```

Output :

['나', '는', '자연어', '처리', '를', '배운다']

Assigning a unique integer to each token.

Code :

```
word_to_index = {word : index for index, word in enumerate(tokens)}
print('단어 집합 :',word_to_index)
```

Output :

```
단어 집합 : {'나': 0, '는': 1, '자연어': 2, '처리': 3, '를': 4, '배운다': 5}
```

A function that takes a token as input and generates its corresponding one-hot vector.

Code :

```
def one_hot_encoding(word, word_to_index):
  one_hot_vector = [0]*(len(word_to_index))
  index = word_to_index[word]
  one_hot_vector[index] = 1
  return one_hot_vector
```

Input :

```
one_hot_encoding("자연어", word_to_index)
```

Output :

[0, 0, 1, 0, 0, 0]

Using Keras' texts_to_sequences() method

Text = "나랑 점심 먹으러 갈래 점심 메뉴는 햄버거 갈래 갈래 햄버거 최고야"

```
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.utils import to_categorical

text = "나랑 점심 먹으러 갈래 점심 메뉴는 햄버거 갈래 갈래 햄버거 최고야"

tokenizer = Tokenizer()
tokenizer.fit_on_texts([text])
print('vocab :',tokenizer.word_index)
```

Output

```
vocab : {'갈래': 1, '점심': 2, '햄버거': 3, '나랑': 4, '먹으러': 5, '메뉴는': 6, '최고야': 7}
```

↓

texts_to_sequences() transforms the sub-text into an integer sequence based on the vocabulary.

```
sub_text = "점심 먹으러 갈래 메뉴는 햄버거 최고야"
encoded = tokenizer.texts_to_sequences([sub_text])[0]
print(encoded)
```

Output

```
[2, 5, 1, 6, 3, 7]
```

performing one-hot encoding on an integer-encoded sequence using to_categorical()

```
one_hot = to_categorical(encoded)
print(one_hot)
```

Output

```
[[0. 0. 1. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 1. 0. 0.]
 [0. 1. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 1. 0.]
 [0. 0. 0. 1. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 1.]]
```

commonly found in libraries like Keras (TensorFlow), is used to convert text into sequences of integers based on a pre-defined vocabulary (word index). Let's break down how it works and how it handles words that are not in the vocabulary (Out-of-Vocabulary or OOV words).

One-hot vectors have the following disadvantages:

1. **Inefficient Storage:** As the size of the vocabulary increases, the dimensionality of the vectors increases, making storage very inefficient. The number of dimensions needed is equal to the number of words in the vocabulary.

2. **Inability to Represent Word Similarity:** They fail to capture any semantic similarity between words. For example, they cannot represent that "dog" is more similar to "puppy" than it is to "refrigerator." This can lead to issues in search systems, such as not being able to provide related search terms like "Sapporo guesthouse" when searching for "Sapporo accommodation."
To address these shortcomings, techniques that reflect the latent meaning of words and vectorize them in a multi-dimensional space are used. There are two main approaches:

1. **Count-Based Methods:** LSA (Latent Semantic Analysis), HAL
2. **Prediction-Based Methods:** NNLM, RNNLM, Word2Vec, FastText
3. **Combination of Count and Prediction-Based Methods:** GloVe