PRACTICAL – 5

AIM: - Use the Keras deep learning library and split words with (text_to_word_sequence).

KERAS: -

Keras is an open-source deep learning library written in Python. It provides a user-friendly and modular interface for designing, building, training, and deploying various types of artificial neural networks, particularly deep neural networks. Keras was developed with a focus on enabling rapid experimentation and prototyping of deep learning models.

The features of the keras are User-Friendly, Modularity, Compatibility, Flexibility, Extensibility, Visualization.

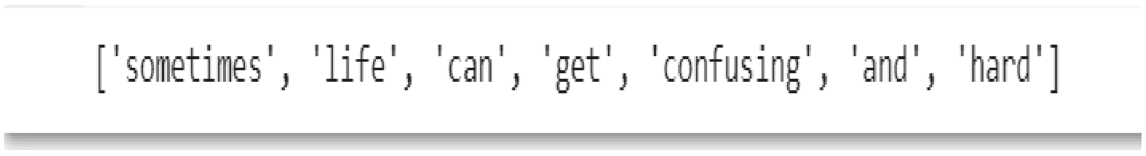
For installation of the keras in colab we have to use the command

!pip install -q keras

CODE: -

```
from keras.preprocessing.text import text_to_word_sequence
txt = "Sometimes life can get confusing and hard"
result = text_to_word_sequence(txt)
print(result)
```

OUTPUT: -

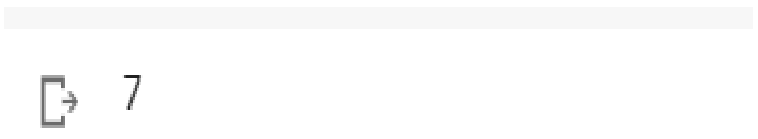


```
['sometimes', 'life', 'can', 'get', 'confusing', 'and', 'hard']
```

CODE: -

```
from keras.preprocessing.text import text_to_word_sequence

txt = "Sometimes life can get confusing and hard"

words = set(text_to_word_sequence(txt))

vocab_size  =  len(words)

print(vocab_size)
```

OUTPUT: -



```
7
```

PRACTICAL – 6

AIM: - Use the Keras deep learning library and write a code for encoding with(one_hot).

ONE_HOT: -

One-hot encoding is a technique used in deep learning and natural language processing to represent categorical data, such as words or labels, as binary vectors. In one-hot encoding, each category is represented by a vector where all elements are set to zero except for the element corresponding to the category's index, which is set to one. This creates a unique binary representation for each category, allowing them to be easily fed into machine learning algorithms.

For example, consider a simple vocabulary of three words: "apple", and "banana”.

To one-hot encode these words:

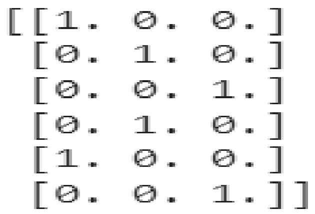
"apple" could be represented as [1, 0, 0]

"banana" could be represented as [0, 1, 0]

CODE: -

```
from keras.utils import to_categorical

color_labels = [0, 1, 2, 1, 0, 2]

one_hot_encoded = to_categorical(color_labels)

print(one_hot_encoded)
```

OUTPUT: -



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

PRACTICAL – 7

AIM: - Use the Keras deep learning library and write a code for Hash Encoding with (hashing_trick).

HASH ENCODING: -

Hashing Encoding, often implemented using the **"hashing_trick"** function in Keras, is a technique used to convert categorical data into numerical representations. This is particularly useful when dealing with a large number of categories and you want to reduce the dimensionality of the representation. Let's go through an example using movie genres.

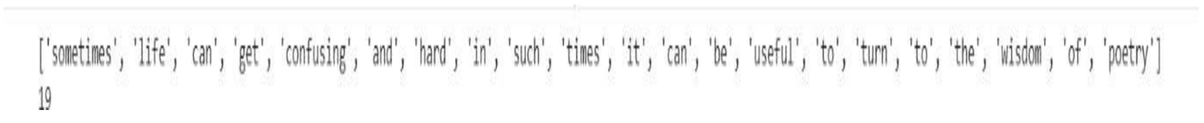
Imagine you have a dataset of movies, and each movie is associated with one or more genres. The genres include "Action", "Comedy", "Drama", "Horror", "Romance", and so on. You want to represent these genres numerically using Hashing Encoding.

BEFORE HASHING: -

CODE: -

```
from keras.preprocessing.text import text_to_word_sequence
text = "Sometimes life can get confusing and hard, In such times it can be useful to turn to the  wisdom of poetry."
result = text_to_word_sequence(text)
print(result)
words = set(text_to_word_sequence(text))
vocab_size = len(words)
print(vocab_size)
```

CODE: -



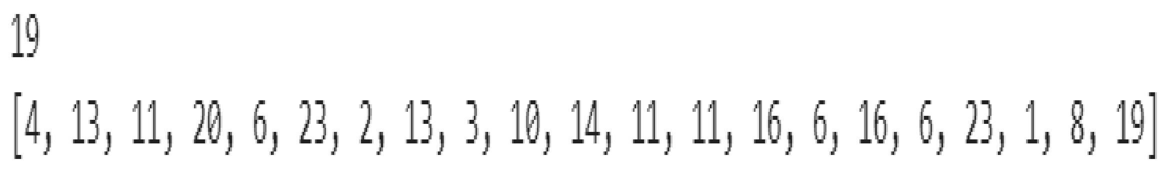
```
['sometimes', 'life', 'can', 'get', 'confusing', 'and', 'hard', 'in', 'such', 'times', 'it', 'can', 'be', 'useful', 'to', 'turn', 'to', 'the', 'wisdom', 'of', 'poetry']
19
```

BY USING ONE_HOT: -

CODE: -

```
from keras.preprocessing.text import one_hot

from keras.preprocessing.text import text_to_word_sequence

text = "Sometimes life can get confusing and hard, In such times it can be useful to turn to the  wisdom of poetry."

words = set(text_to_word_sequence(text))

vocab_size = len(words)

print(vocab_size)

result = one_hot(text, round(vocab_size*1.3))

print(result)
```

OUTPUT: -



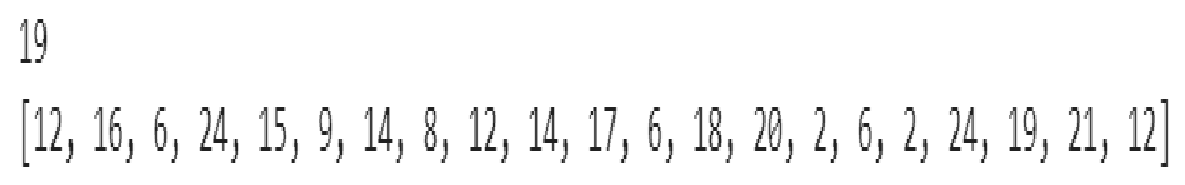
```
19
[4, 13, 11, 20, 6, 23, 2, 13, 3, 10, 14, 11, 11, 16, 6, 16, 6, 23, 1, 8, 19]
```

AFTER HASHING: -

CODE: -

```
from keras.preprocessing.text import hashing_trick

from keras.preprocessing.text import text_to_word_sequence

text = "Sometimes life can get confusing and hard, In such times it can be useful
to turn to the  wisdom of poetry."

words = set(text_to_word_sequence(text))

vocab_size = len(words)

print(vocab_size)

result = hashing_trick(text, round(vocab_size*1.3), hash_function='md5')

print(result)
```

OUTPUT: -



```
19
[12, 16, 6, 24, 15, 9, 14, 8, 12, 14, 17, 6, 18, 20, 2, 6, 2, 24, 19, 21, 12]
```

PRACTICAL – 8

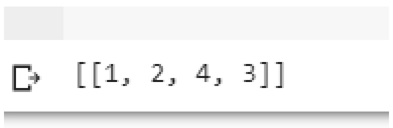
AIM : Use the Keras deep learning library give a demo of Tokenizer API.

1. Tokenization: The tokenizer first breaks down the text into individual words or tokens. This is done using the fit_on_texts method.
2. Indexing: Each unique word is then assigned a unique integer index. This is done by the word_index property of the tokenizer object.
3. Text to sequence conversion: The texts_to_sequences method can be used to convert a list of text samples to a list of sequences of integers.

CODE:

```
import tensorflow as tf
from tensorflow.keras.preprocessing.text
import Tokenizer # Create a tokenizer
tokenizer =
Tokenizer(num_words=1
000) # Fit the tokenizer on
a list of texts
texts = ["This is a text.", "This is
another text."]
tokenizer.fit_on_texts(texts)
# Convert a text to a
sequence of tokens text
= "This is a new text."
tokens =
tokenizer.texts_to_sequences(
[text]) # Print the tokens
print(tokens)
```

OUTPUT:



```
[[1, 2, 4, 3]]
```