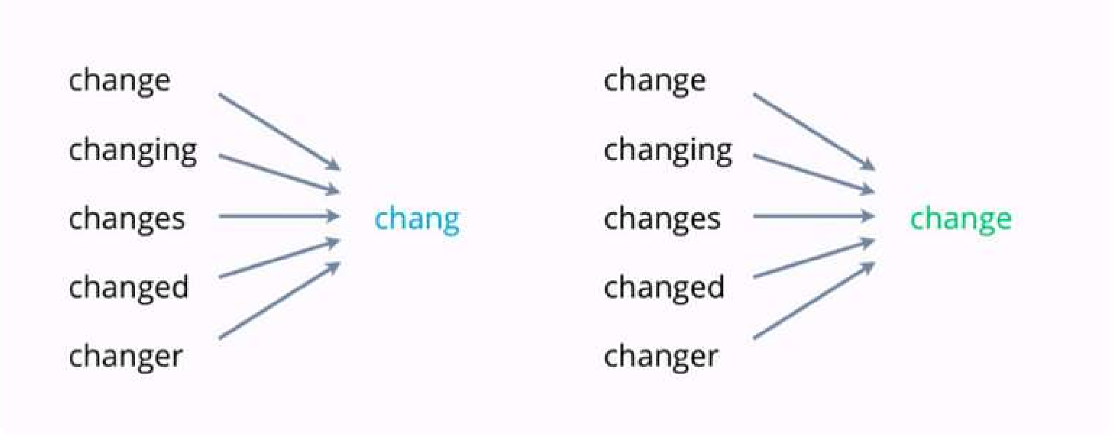
Aysel Aydin

Summary

The website content discusses stemming and lemmatization as text

2— Stemming & Lemmatization in NLP: Text Preprocessing Techniques



and explained some of the text preprocessing techniques. **Click to read**

In this article, we will cover the **Stemming & Lemmatization** topics.

Stemming and lemmatization are two text preprocessing techniques used to reduce words to their base or root form. The primary goal of these techniques is to reduce the number of unique words in a text document, making it easier to analyze and understand.

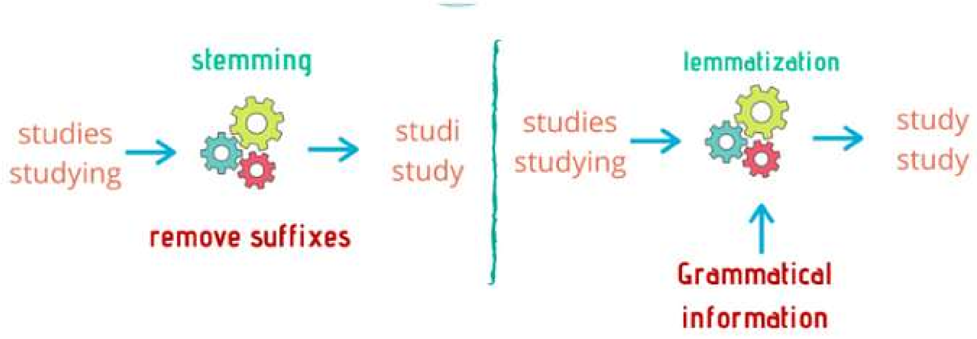
They are widely used for Search engines and tagging. Search engines use stemming for indexing the words. Therefore, instead of storing all forms of a word, a search engine may only store its roots. In this way, stemming reduces the size of the index and increases retrieval accuracy.

Let’s learn them deeply!

Stemming involves removing suffixes from words to obtain their base form while lemmatization involves converting words to their morphological base form.

Stemming is a **simpler** and **faster** technique compared to lemmatization. It uses a set of rules or algorithms to remove suffixes and obtain the base form of a word. However, stemming can sometimes produce a base form that is not valid, in which case it can also lead to ambiguity.

vocabulary and morphological analysis to determine the base form of a word. Lemmatization is **slower** and **more complex** than stemming. It produces a valid base form that can be found in a dictionary, making it more accurate than stemming.



Stemming is preferred when the meaning of the word is **not important** for analysis. for example: **Spam Detection**

Lemmatization would be recommended when the meaning of the word is **important** for analysis. for example: **Question Answer**

Porter & Zemberek Porter stemming algorithm is one of the most common stemming algorithms which is basically designed to remove and replace well-known suffixes of English words.

find word roots in Turkish is known as “Zemberek”. Zemberek is a natural language processing library that can separate word roots and suffixes in accordance with the language structure and morphology of Turkish.

Although the Porter Stemming Algorithm was developed for English texts, it can be adapted to different languages. However, it is more effective to use natural language processing tools and algorithms specifically designed for different languages such as Turkish, as they are not fully adapted to the characteristics of the language.

Zemberek is more successful in understanding and processing the rich morphological structure of Turkish and therefore gives better results on Turkish texts. Therefore, it is more common to choose language-specific tools such as Zemberek for language processing and root-finding tasks for Turkish.

> ***I will cover the subject of “Zemberek” in more detail in another article.***

Let’s see how it works Porter stemming algorithm:

```
from nltk.stem.porter import PorterStemmer

stemmer = PorterStemmer()

def stem_words(text):
    word_tokens = text.split()
```

```
text = 'text preprocessing techniques for natural language processing by Ayse
stem_words(text)
```

Output:

```
['text',
 'preprocess',
 'techniqu',
 'for',
 'natur',
 'languag',
 'process',
 'by',
 'aysel',
 'aydin']
```

Now let’s consider the topic of “Lemmatization”

In our lemmatization example, we will be using a popular lemmatizer called **WordNet** lemmatizer.

WordNet is a word association database for English and a useful resource for English lemmatization. However, there is no direct equivalent of this source for the lemmatization of Turkish texts.

As I mentioned above, I will discuss the subject of “Zemberek” in more detail in another article.

Let’s code and apply Lemmatization.

```
from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()

def lemmatize_word(text):
    word_tokens = text.split()
    lemmas = [lemmatizer.lemmatize(word, pos ='v') for word in word_tokens]
    return lemmas

text = 'text preprocessing techniques for natural language processing by Ayse
lemmatize_word(text)
```

Output:

```
['text',
 'preprocessing',
 'techniques',
 'for',
 'natural',
     'by',
     'Aysel',
     'Aydin']
```

Conclusion

To summarize, stemming and lemmatization are methods that help us in text preprocessing for Natural Language Processing. They both aim to reduce inflections down to common base root words, but each takes a different approach to doing so.

In some cases, stemming may produce better results than lemmatization, while in other cases, lemmatization may be more accurate. Therefore, it is essential to weigh the trade-offs between simplicity, speed, and accuracy when selecting a text normalization technique.

I hope it will be a useful article for you. Happy coding 🤞

Contact Accounts: Twitter, LinkedIn

NLP Lemmatization Stemming Text Preprocesing

Recommended from ReadMedium

Aysel Aydin

10—Understanding Word2Vec 1: Word Embedding in NLP

In this article, we will talk about Word2vec one of the word embedding techniques. Before we start, I recommend you read the article I…

3 min read

Mdabdullahalhasib

A Simple Introduction to LangChain Framework

Today we will explore LangChain, a framework of Generative AI to develop end-to-end LLMs-based applications.

3 min read

Vipra Singh

LLM Architectures Explained: NLP Fundamentals (Part 1)

Deep Dive into the architecture & building of real-world applications leveraging NLP Models starting from RNN to the Transformers.

39 min read

Nivedita Bhadra

15 min read

Dr. Walid Soula

Bag-of-Words

Explore the fundamentals of the Bag-of-Words model in natural language processing

4 min read

Vipra Singh

LLM Architectures Explained: Word Embeddings (Part 2)

Deep Dive into the architecture & building real-world applications leveraging NLP Models starting from RNN to Transformer.

53 min read

Jo Wang

Deep Learning Part 5 -How to prevent overfitting

Techniques used to prevent overfitting in deep learning models:

4 min read

Harsh Vardhan

referred to as word embeddings — is…

10 min read

Rahul Kumar

NLP Hands-On with Text Classification

This post is a part of the NLP Hands-on series and consists of the following tasks: 1. Text Classification 2. Token Classification 3…

3 min read

Mdabdullahalhasib

A Complete Guide to Embedding For NLP & Generative AI/LLM

Understand the concept of vector embedding, why it is needed, and implementation with LangChain.

11 min read

Ajay Halthor

Word2Vec, GloVe, and FastText, Explained

How computers understand words

10 min read