

# Fuzzy Pattern Matching in NLP

In Natural Language Processing, we often deal with messy or imperfect text typos, misspellings, and inconsistencies.

Fuzzy pattern matching helps us find approximate matches between strings, making NLP systems more flexible and error-tolerant.

**R** por Rafa Carpio Muñoz

# What is Fuzzy Pattern Matching?

## Definition

A technique used to find similarities between strings that are not exactly the same.

## Key Feature

Handles small errors or differences in text, such as typos or rearranged words.

## Examples

- "hello" vs "helo" → 80% similarity
- "apple" vs "appel" → close enough to match



# Why is it useful in NLP?



## Spell correction

Automatically corrects typing errors and misspellings in text.



## Entity matching

Identifies customer names, locations, and other entities despite variations and inconsistencies.



## Chatbots

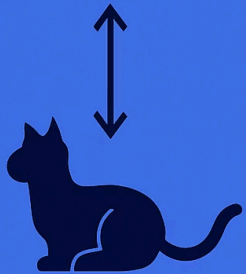
Helps chatbots understand and respond to imperfect or noisy user input.



## Search systems

Provides useful fuzzy suggestions when exact matches are not found.

KITTEN  
kitten



String Distance:  
0

Case Difference:  
Upper/Lower



# Levenshtein Distance

1

**Original**

"kitten"

2

**Step 1**

"sitten" (replace 'k' with 's')

3

**Step 2**

"sittin" (replace 'e' with 'i')

4

**Final**

"sitting" (add 'g')

The Levenshtein distance counts the minimum number of single-character edits (insertions, deletions, or substitutions) required to change one word into another. Fewer edits indicate greater similarity between the strings.

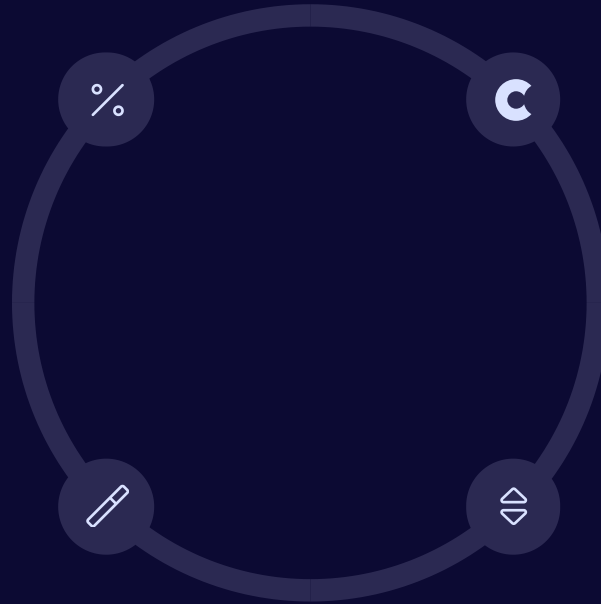
# TheFuzz Library in Python

## `fuzz.ratio()`

Calculates a simple similarity score between two strings.

## `fuzz.token_set_ratio()`

Handles partial matches even when the words are in different orders.



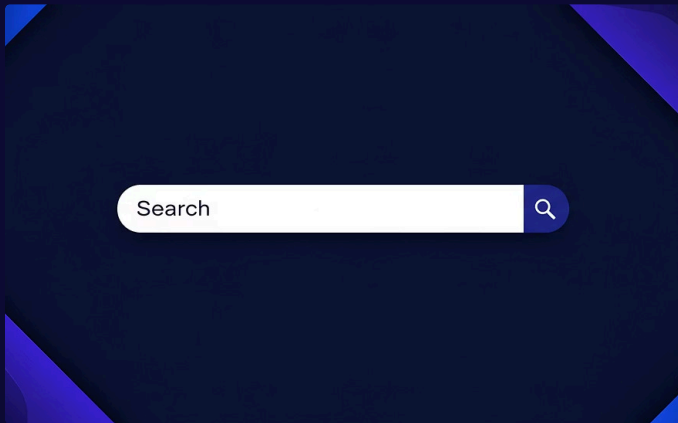
## `fuzz.partial_ratio()`

Finds the best matching substring within longer strings.

## `fuzz.token_sort_ratio()`

Compares two strings after sorting the words, ignoring word order.

# Common Use Cases

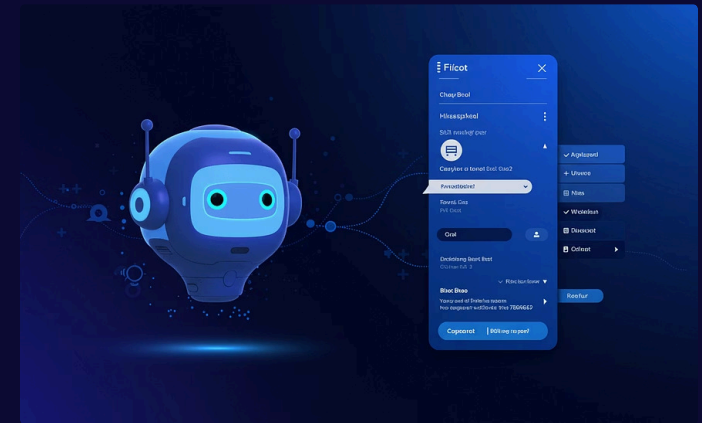


## Form Correction

## Fixing user typos in search boxes and online forms.

# Database Matching

Finding records with inconsistent name spellings.



## Smart Chatbots

Understanding user intent despite  
typing errors.

# Limitations



## Text Length

Not suitable for comparing entire documents or very long texts.

---



## False Positives

May incorrectly match unrelated strings if similarity thresholds are set too low.

---



## Performance

Can be computationally expensive when applied to large datasets.

# Conclusion

**1**

## **String Comparison**

Allows comparison of text that isn't exactly identical.

**2**

## **Levenshtein Distance Basis**

Based on the concept of edit distance between strings.

**3**

## **Simple Implementation**

Easy to use with Python libraries like TheFuzz.

**4**

## **Practical Applications**

Widely used in spell checkers, chatbots, and other NLP systems.