Date :  20-10-2023

text similarity checker (1.0.0)

# Overview

Comparison of similarity between two texts using popular algorithms, as well as comparison of similarity with text lemmatization and without, along with removal of stop words and without. Display common words and bigrams.

* Cosine Similarity
* Jaccard Similarity
* Fuzzy Ratio

By using the Text Similarity Checker API in this way, academic institutions can maintain academic integrity, uphold their standards, and encourage original research and learning. The API automates the plagiarism detection process, making it faster and more objective than manual checks.

# Getting Started

## Prerequisites

* Volt Foundry
* [**Rapid API Account**](https://rapidapi.com/altanalys/api/text-similarity-checker)
* X-RapidAPI-Key, X-RapidAPI-Host

## Importing the adapter

**To import the Data Adapter to Volt Foundry, do the following:**

1. Sign in to the  [HCL Foundry](https://manage.hclvoltmx.com/).
2. From the left navigation menu, select **API Management**.
3. In **API Management**, select **Custom Data Adapters**.  
   
4. Click **IMPORT** to import a custom data adapter.  
   
5. On the Import Data Adapter dialog box, click browser to import.  
   
6. Select **Text Similarity Checker zip** file and click **IMPORT**.

After you import the data adapter, Volt Foundry opens a window that shows the metadata of the data adapter.

A screenshot of a computer

Description automatically generated

After you import the data adapter, you can view it on the Custom Data Adapters page and use it to create services on Volt Foundry.

A screenshot of a computer

Description automatically generated

## [Creating an Integration service](javascript:void(0);)

After you import the data adapter into Volt Foundry, you can use it to create an Integration Service.

Follow the given steps to create an Integration service using the **Text Similarity Checker** Adapter.

1. Sign in to the [HCL Foundry](https://manage.hclvoltmx.com/).
2. From the left navigation menu, select **API Management**.
3. In **API Management**, select **Integration**.  
   
4. To create a new service, click the **+** button or the **CONFIGURE NEW** button.  
   
5. On the Service Definition tab, select the service type as**Text Similarity Checker,** and click **SAVE**.

A screenshot of a computer

Description automatically generated

Alternatively, you can also create a Foundry app and create an Integration service inside it.

E. [Creating and Executing operations](javascript:void(0);)



After you create an integration service, you can create and execute operations using the service.

#### Creating an Operation

* In **API Management/Foundry app you created**, in the **Integration** section, select the service that you created.
* After you select the service, navigate to the **Operation List** tab.  
  
* From the drop down list, select an operation that you want to execute, and click **ADD OPERATION**.

A screenshot of a computer

Description automatically generated

#### Executing an Operation

* From the **Operations List** tab, in the **Configured Operations** section, select the operation you want to execute.

A screenshot of a computer

Description automatically generated

* On the Operation Page, in the **Request Input** tab, in the **Body** section, for the following parameters, specify the values in the respective boxes under the **Test Value** column.

**Mandatory Parameters**

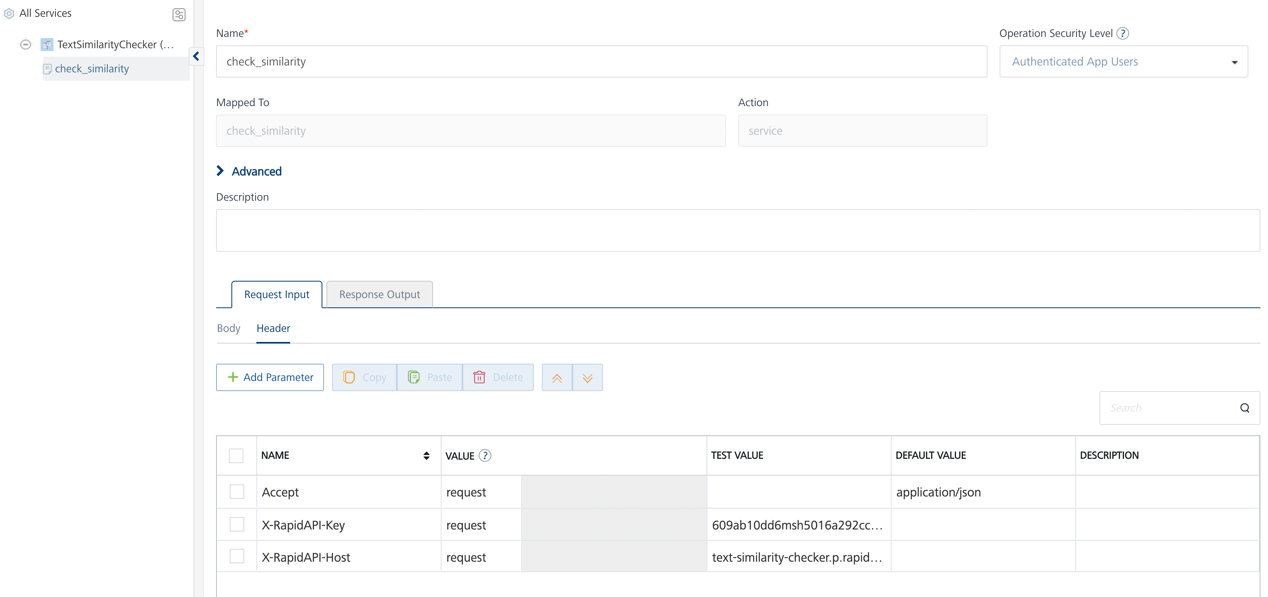
* + **X-RapidAPI-Key, X-RapidAPI-Host [in header]:**  API Key associated with your RapidAPI account, required for authentication.

To get keys – Login in to [**Rapid API Account**](https://rapidapi.com/altanalys/api/text-similarity-checker)-> You can view your keys.

* + **text1:** The first text input for comparison.
  + **text2:** The second text input for comparison.

A screenshot of a computer

Description automatically generated



* Select a run-time environment and click **Save and Fetch Response** to get a response based on your inputs.



## [Publishing your application](javascript:void(0);)

If you want to use the services in client applications, you need to publish an app to a run-time environment. You can create the service (as described above) in an application or import the service into an application and publish the application.

# References

## Endpoint Documentation

### **Requests Input Parameters**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| X-RapidAPI-Key(string) | **[Required]** **[Header]**API Key associated with your account. |
| X-RapidAPI-Host (string) | **[Required]** **[Header]**The host parameter specifies the domain name or IP address of the RapidAPI endpoint. In the context of the Text Similarity Checker API, the default host is set to 'text-similarity-checker.p.rapidapi.com'. This parameter is necessary for routing the request to the appropriate API endpoint. |
| text1 (string) | **[Required]** The first text input for comparison. |
| text2 (string) | **[Required]** The second text input for comparison. |

### **Response and Definitions**

Your API response strongly depends on your query and the parameters you choose to make use of. Find below a representative example API response, returned for the comparisons of two texts number. Each available response object will be explained further below in this section.

Example:

Text1: The Sun is a dazzling beacon of life in the heavens. Its brilliant corona unfurls during eclipses, captivating Earth's inhabitants.

Text2: The Sun stands as a radiant symbol of vitality in the celestial realm. Its resplendent corona unveils itself amidst eclipses, captivating the denizens of Earth.

#### Response Object:

|  |  |
| --- | --- |
| **Object** | **Description** |
| lemmatized\_text\_without\_stopwords -> total\_common\_lemmas (string) | Total count of common lemmas between the two texts. |
| lemmatized\_text\_without\_stopwords -> levenshtein\_distance (string) | Levenshtein distance metric between the two texts. |
| lemmatized\_text\_without\_stopwords -> total\_common\_lemmatized\_bigrams (string): | Total count of common lemmatized bigrams between the two texts. |
| lemmatized\_text\_without\_stopwords -> jaccard\_similarity (string): | Jaccard similarity coefficient between the two texts. |
| lemmatized\_text\_without\_stopwords -> cosine\_similarity (string) | Cosine similarity between the two texts. |
| lemmatized\_text\_without\_stopwords -> fuzzy\_ratio (string) | Fuzzy matching ratio between the two texts. |
| lemmatized\_text\_without\_stopwords -> tf\_idf\_cosine\_similarity (string) | TF-IDF weighted cosine similarity between the two texts. |
| lemmatized\_text\_without\_stopwords -> common\_lemmatized\_bigrams (array) | Array of common lemmatized bigrams between the two texts. |
| lemmatized\_text\_without\_stopwords -> common\_lemmas (array) | Array of common lemmas between the two texts. |
| text\_as\_is -> levenshtein\_distance (string): | Levenshtein distance metric for the texts without preprocessing. |
| text\_as\_is -> jaccard\_similarity (string) | Jaccard similarity coefficient for the texts without preprocessing. |
| text\_as\_is -> cosine\_similarity (string) | Cosine similarity for the texts without preprocessing. |
| text\_as\_is -> fuzzy\_ratio (string) | Fuzzy matching ratio for the texts without preprocessing. |
| without\_stopwords -> levenshtein\_distance (string) | Levenshtein distance metric for the texts without stopwords. |
| without\_stopwords -> jaccard\_similarity (string) | Jaccard similarity coefficient for the texts without stopwords. |
| without\_stopwords -> cosine\_similarity (string) | Cosine similarity for the texts without stopwords. |
| without\_stopwords -> fuzzy\_ratio (string) | Fuzzy matching ratio for the texts without stopwords. |

**Reference Document:**  [[Text Similarity Checker API Documentation (altanalys) | RapidAPI](https://rapidapi.com/altanalys/api/text-similarity-checker)](https://rapidapi.com/rahilkhan224/api/vehicle-rc-verification-india)

# Revision History

Adapter version 1.0.0:

## Known Issues

* No Known Issues

## Limitations

* No limitations