Toggle navigation sidebar

Toggle in-page Table of Contents

# 

[**My sample book**](http://docs.google.com/index.html)

* [Welcome to your Jupyter Book](http://docs.google.com/intro.html)
* [Markdown Files](http://docs.google.com/markdown.html)
* [Notebooks with MyST Markdown](http://docs.google.com/markdown-notebooks.html)

Powered by [Jupyter Book](https://jupyterbook.org)

*  [Binder](https://mybinder.org/v2/gh/executablebooks/jupyter-book/master?urlpath=tree/docs/notebooks.ipynb)
* [repository](https://github.com/executablebooks/jupyter-book)
* [open issue](https://github.com/executablebooks/jupyter-book/issues/new?title=Issue+on+page+/notebooks.html&body=Your+issue+content+here.)
* [.ipynb](http://docs.google.com/_sources/notebooks.ipynb)
* .pdf

Contents

## Contents

***Summarization*** adalah suatu cara untuk meringkas suatu dokumen ke dalam bentuk yang lebih sederhana dengan mengambil poin-poin penting dalam dokumen tersebut.

Secara umum, pendekatan news summarization dibagi menjadi dua, yaitu extractive dan abstractive. Extractive summarization berarti mengidentifikasi bagian-bagian penting dari teks dan membuat kalimat dari kata-kata tersebut. Sementara abstractive summarization mereproduksi materi penting dengan cara baru setelah melakukan interpretasi dan memeriksa teks menggunakan NLP untuk membuat kalimat baru yang berasal dari dokumen aslinya.

***TextRank*** adalah algoritma peringkat berbasis grafik untuk memproses teks. TextRank menghasilkan ekstraksi kalimat sebagai ringkasan. Salah satu kelebihan dari algoritma ini, tidak diperlukannya pelatihan menggunakan data training pada algoritma yang digunakan.

**Langkah pertama adalah meng-import library yang diperlukan**

import numpy as np  
import pandas as pd  
import nltk  
import re  
from nltk.tokenize import sent\_tokenize  
from nltk.corpus import stopwords  
from gensim.models import Word2Vec  
from scipy import spatial  
import networkx as nx

**Teks berita yang digunakan**

text='''The Double Asteroid Redirection Test successfully changed the trajectory of the asteroid Dimorphos when the NASA spacecraft intentionally slammed into the space rock on September 26, according to the agency. The DART mission, a full-scale demonstration of deflection technology, was the world’s first conducted on behalf of planetary defense. The mission was also the first time humanity intentionally changed the motion of a celestial object in space. Prior to impact, it took Dimorphos 11 hours and 55 minutes to orbit its larger parent asteroid Didymos. Astronomers used ground-based telescopes to measure how Dimorphos’ orbit changed after impact. Now, it takes Dimorphos 11 hours and 23 minutes to circle Didymos. The DART spacecraft changed the moonlet asteroid’s orbit by 32 minutes. Initially, astronomers expected DART to be a success if it shortened the trajectory by 10 minutes. “All of us have a responsibility to protect our home planet. After all, it’s the only one we have,” said NASA Administrator Bill Nelson. “This mission shows that NASA is trying to be ready for whatever the universe throws at us. NASA has proven we are serious as a defender of the planet. This is a watershed moment for planetary defense and all of humanity, demonstrating commitment from NASA’s exceptional team and partners from around the world.”'''

import nltk  
nltk.download('punkt')  
from nltk import sent\_tokenize

[nltk\_data] Downloading package punkt to /root/nltk\_data...  
[nltk\_data] Unzipping tokenizers/punkt.zip.

**Tokenizing paragraph ke bentuk list sentences**

sentences=sent\_tokenize(text)  
print(sentences)

['The Double Asteroid Redirection Test successfully changed the trajectory of the asteroid Dimorphos when the NASA spacecraft intentionally slammed into the space rock on September 26, according to the agency.', 'The DART mission, a full-scale demonstration of deflection technology, was the world’s first conducted on behalf of planetary defense.', 'The mission was also the first time humanity intentionally changed the motion of a celestial object in space.', 'Prior to impact, it took Dimorphos 11 hours and 55 minutes to orbit its larger parent asteroid Didymos.', 'Astronomers used ground-based telescopes to measure how Dimorphos’ orbit changed after impact.', 'Now, it takes Dimorphos 11 hours and 23 minutes to circle Didymos.', 'The DART spacecraft changed the moonlet asteroid’s orbit by 32 minutes.', 'Initially, astronomers expected DART to be a success if it shortened the trajectory by 10 minutes.', '“All of us have a responsibility to protect our home planet.', 'After all, it’s the only one we have,” said NASA Administrator Bill Nelson.', '“This mission shows that NASA is trying to be ready for whatever the universe throws at us.', 'NASA has proven we are serious as a defender of the planet.', 'This is a watershed moment for planetary defense and all of humanity, demonstrating commitment from NASA’s exceptional team and partners from around the world.”']

**Tahap Preprocessing**

import nltk  
nltk.download('stopwords')

[nltk\_data] Downloading package stopwords to /root/nltk\_data...  
[nltk\_data] Unzipping corpora/stopwords.zip.

True

* re.sub(r’[^\w\s]’) menghapus semua tanda baca dari setiap kalimat menggunakan pemahaman daftar
* Hapus stopwords (a, the, an, dll) untuk fokus pada kata-kata yang lebih signifikan dalam sebuah kalimat.

sentences\_clean=[re.sub(r'[^\w\s]','',sentence.lower()) for sentence in sentences]  
stop\_words = stopwords.words('english')  
sentence\_tokens=[[words for words in sentence.split(' ') if words not in stop\_words] for sentence in sentences\_clean]  
print(sentence\_tokens)

[['double', 'asteroid', 'redirection', 'test', 'successfully', 'changed', 'trajectory', 'asteroid', 'dimorphos', 'nasa', 'spacecraft', 'intentionally', 'slammed', 'space', 'rock', 'september', '26', 'according', 'agency'], ['dart', 'mission', 'fullscale', 'demonstration', 'deflection', 'technology', 'worlds', 'first', 'conducted', 'behalf', 'planetary', 'defense'], ['mission', 'also', 'first', 'time', 'humanity', 'intentionally', 'changed', 'motion', 'celestial', 'object', 'space'], ['prior', 'impact', 'took', 'dimorphos', '11', 'hours', '55', 'minutes', 'orbit', 'larger', 'parent', 'asteroid', 'didymos'], ['astronomers', 'used', 'groundbased', 'telescopes', 'measure', 'dimorphos', 'orbit', 'changed', 'impact'], ['takes', 'dimorphos', '11', 'hours', '23', 'minutes', 'circle', 'didymos'], ['dart', 'spacecraft', 'changed', 'moonlet', 'asteroids', 'orbit', '32', 'minutes'], ['initially', 'astronomers', 'expected', 'dart', 'success', 'shortened', 'trajectory', '10', 'minutes'], ['us', 'responsibility', 'protect', 'home', 'planet'], ['one', 'said', 'nasa', 'administrator', 'bill', 'nelson'], ['mission', 'shows', 'nasa', 'trying', 'ready', 'whatever', 'universe', 'throws', 'us'], ['nasa', 'proven', 'serious', 'defender', 'planet'], ['watershed', 'moment', 'planetary', 'defense', 'humanity', 'demonstrating', 'commitment', 'nasas', 'exceptional', 'team', 'partners', 'around', 'world']]

Seperti yang kita ketahui kita perlu mengkonversi data teks ke dalam bentuk numerik, pertama kita akan menghitung penyisipan kata & menggunakan penyematan ini akan menghitung penyisipan kalimat menggunakan variabel sentence\_tokens yang dihitung di atas.

* Menggunakan Word2Vec() gensim, menghitung embedding untuk setiap kata dari teks.
* Menggunakan pemahaman daftar, kami akan mengganti token kata dengan penyematannya di kalimat\_tokens
* Menghitung max\_length kalimat dalam teks yang ada
* Penyematan kalimat padding menggunakan 0s hingga max\_len. Ini telah dilakukan untuk mempertahankan ukuran embedding yang sama untuk setiap kalimat.

w2v=Word2Vec(sentence\_tokens,size=1,min\_count=1,iter=1000)  
sentence\_embeddings=[[w2v[word][0] for word in words] for words in sentence\_tokens]  
max\_len=max([len(tokens) for tokens in sentence\_tokens])  
sentence\_embeddings=[np.pad(embedding,(0,max\_len-len(embedding)),'constant') for embedding in sentence\_embeddings]  
print(sentence\_embeddings)

WARNING:gensim.models.base\_any2vec:consider setting layer size to a multiple of 4 for greater performance

[array([-1.2670156, -1.5794804, -1.4380718, -1.415217 , -1.4635082,  
 -1.5246675, -1.5068012, -1.5794804, -1.5284365, -1.6130797,  
 -1.4938716, -1.5634602, -1.4080256, -1.6058408, -1.5834286,  
 -1.6295552, -1.6888691, -1.6887738, -1.6421251], dtype=float32), array([-1.586549 , -1.6626729, -1.5217707, -1.6778601, -1.6859446,  
 -1.7265809, -1.672364 , -1.6641412, -1.632014 , -1.7067887,  
 -1.7949789, -1.7757899, 0. , 0. , 0. ,  
 0. , 0. , 0. , 0. ], dtype=float32), array([-1.6626729, -1.4014759, -1.6641412, -1.4809706, -1.6756613,  
 -1.5634602, -1.5246675, -1.4847059, -1.47967 , -1.5145583,  
 -1.6058408, 0. , 0. , 0. , 0. ,  
 0. , 0. , 0. , 0. ], dtype=float32), array([-1.3803481, -1.4736205, -1.4587469, -1.5284365, -1.5364983,  
 -1.5234815, -1.3783212, -1.4999757, -1.4791976, -1.3860564,  
 -1.3682328, -1.5794804, -1.5292237, 0. , 0. ,  
 0. , 0. , 0. , 0. ], dtype=float32), array([-1.5944893, -1.4385271, -1.3992647, -1.3764164, -1.3645425,  
 -1.5284365, -1.4791976, -1.5246675, -1.4736205, 0. ,  
 0. , 0. , 0. , 0. , 0. ,  
 0. , 0. , 0. , 0. ], dtype=float32), array([-1.3771424, -1.5284365, -1.5364983, -1.5234815, -1.3990564,  
 -1.4999757, -1.4171519, -1.5292237, 0. , 0. ,  
 0. , 0. , 0. , 0. , 0. ,  
 0. , 0. , 0. , 0. ], dtype=float32), array([-1.586549 , -1.4938716, -1.5246675, -1.2809511, -1.3748078,  
 -1.4791976, -1.2662787, -1.4999757, 0. , 0. ,  
 0. , 0. , 0. , 0. , 0. ,  
 0. , 0. , 0. , 0. ], dtype=float32), array([-1.3906406, -1.5944893, -1.4494677, -1.586549 , -1.5126129,  
 -1.4927983, -1.5068012, -1.4482985, -1.4999757, 0. ,  
 0. , 0. , 0. , 0. , 0. ,  
 0. , 0. , 0. , 0. ], dtype=float32), array([-1.6559898, -1.5088469, -1.4694964, -1.4669436, -1.4989651,  
 0. , 0. , 0. , 0. , 0. ,  
 0. , 0. , 0. , 0. , 0. ,  
 0. , 0. , 0. , 0. ], dtype=float32), array([-1.4394666, -1.4980348, -1.6130797, -1.4643806, -1.5354937,  
 -1.5324651, 0. , 0. , 0. , 0. ,  
 0. , 0. , 0. , 0. , 0. ,  
 0. , 0. , 0. , 0. ], dtype=float32), array([-1.6626729, -1.5183941, -1.6130797, -1.6187509, -1.5915618,  
 -1.7111207, -1.6111715, -1.6975976, -1.6559898, 0. ,  
 0. , 0. , 0. , 0. , 0. ,  
 0. , 0. , 0. , 0. ], dtype=float32), array([-1.6130797, -1.3530707, -1.380068 , -1.3405163, -1.4989651,  
 0. , 0. , 0. , 0. , 0. ,  
 0. , 0. , 0. , 0. , 0. ,  
 0. , 0. , 0. , 0. ], dtype=float32), array([-1.5448844, -1.6986802, -1.7949789, -1.7757899, -1.6756613,  
 -1.6969739, -1.7396877, -1.7333277, -1.7153453, -1.7759681,  
 -1.9175851, -1.8233082, -1.8100604, 0. , 0. ,  
 0. , 0. , 0. , 0. ], dtype=float32)]

/usr/local/lib/python3.7/dist-packages/ipykernel\_launcher.py:2: DeprecationWarning: Call to deprecated `\_\_getitem\_\_` (Method will be removed in 4.0.0, use self.wv.\_\_getitem\_\_() instead).

**calculate the similarity matrix**

Inisialisasi kesamaan\_matriks dimensi N x N di mana N adalah jumlah total kalimat dalam teks

Menggunakan 1-spatial.distance.cosine(), hitung kesamaan antara setiap dua pasang kalimat.

similarity\_matrix = np.zeros([len(sentence\_tokens), len(sentence\_tokens)])  
for i,row\_embedding in enumerate(sentence\_embeddings):  
 for j,column\_embedding in enumerate(sentence\_embeddings):  
 similarity\_matrix[i][j]=1-spatial.distance.cosine(row\_embedding,column\_embedding)  
print(similarity\_matrix)

[[1. 0.77246314 0.73365694 0.79948056 0.65881008 0.61998105  
 0.61723655 0.66020918 0.475243 0.52809298 0.65944999 0.47378656  
 0.80000263]  
 [0.77246314 1. 0.94938952 0.95596969 0.84978849 0.80238801  
 0.79970944 0.85076177 0.62528783 0.6921255 0.85047531 0.62424618  
 0.95687795]  
 [0.73365694 0.94938952 1. 0.9076128 0.90080667 0.85304326  
 0.85361695 0.89969736 0.68452781 0.74891555 0.901757 0.68517971  
 0.9103114 ]  
 [0.79948056 0.95596969 0.9076128 1. 0.83060819 0.78437376  
 0.7818023 0.83233428 0.61990255 0.68424141 0.83203512 0.61882979  
 0.99783927]  
 [0.65881008 0.84978849 0.90080667 0.83060819 1. 0.93947619  
 0.94019282 0.99663335 0.7304424 0.80634427 0.99923003 0.72961861  
 0.81098199]  
 [0.61998105 0.80238801 0.85304326 0.78437376 0.93947619 1.  
 0.99651885 0.94168627 0.78592151 0.86585635 0.93958169 0.78337973  
 0.76640111]  
 [0.61723655 0.79970944 0.85361695 0.7818023 0.94019282 0.99651885  
 1. 0.93725669 0.79683626 0.8742069 0.93835253 0.79590786  
 0.76271296]  
 [0.66020918 0.85076177 0.89969736 0.83233428 0.99663335 0.94168627  
 0.93725669 1. 0.74712187 0.81862271 0.99773008 0.74493712  
 0.81291312]  
 [0.475243 0.62528783 0.68452781 0.61990255 0.7304424 0.78592151  
 0.79683626 0.74712187 1. 0.90802461 0.73081493 0.99920189  
 0.60028493]  
 [0.52809298 0.6921255 0.74891555 0.68424141 0.80634427 0.86585635  
 0.8742069 0.81862271 0.90802461 1. 0.80946362 0.90666276  
 0.65996909]  
 [0.65944999 0.85047531 0.901757 0.83203512 0.99923003 0.93958169  
 0.93835253 0.99773008 0.73081493 0.80946362 1. 0.73012835  
 0.81236738]  
 [0.47378656 0.62424618 0.68517971 0.61882979 0.72961861 0.78337973  
 0.79590786 0.74493712 0.99920189 0.90666276 0.73012835 1.  
 0.59866977]  
 [0.80000263 0.95687795 0.9103114 0.99783927 0.81098199 0.76640111  
 0.76271296 0.81291312 0.60028493 0.65996909 0.81236738 0.59866977  
 1. ]]

**implement PageRank**

Convert the similarity matrix to a network/graph

Apply pagerank()

nx\_graph = nx.from\_numpy\_array(similarity\_matrix)  
scores = nx.pagerank(nx\_graph)  
print(scores)

{0: 0.06612140757141957, 1: 0.07800888494697931, 2: 0.07985098893138433, 3: 0.07749697268053896, 4: 0.0808450240966187, 5: 0.08014744527405773, 6: 0.08024697459595923, 7: 0.08113949435112383, 8: 0.0717032144577252, 9: 0.075386171846794, 10: 0.08089763404917737, 11: 0.07162234485908987, 12: 0.076533442339132}

Sekarang, yang akan kita lakukan adalah membuat kamus dengan kalimat (kita dapatkan setelah sent\_tokenize()) sebagai kunci & PageRank mereka sebagai nilai & mengurutkan kamus ini berdasarkan PageRanks kalimat. Disini diambil 4 kalimat teratas untuk ringkasan:

top\_sentence={sentence:scores[index] for index,sentence in enumerate(sentences)}  
top=dict(sorted(top\_sentence.items(), key=lambda x: x[1], reverse=True)[:4])  
print(top)

{'Initially, astronomers expected DART to be a success if it shortened the trajectory by 10 minutes.': 0.08113949435112383, '“This mission shows that NASA is trying to be ready for whatever the universe throws at us.': 0.08089763404917737, 'Astronomers used ground-based telescopes to measure how Dimorphos’ orbit changed after impact.': 0.0808450240966187, 'The DART spacecraft changed the moonlet asteroid’s orbit by 32 minutes.': 0.08024697459595923}

for sent in sentences:  
 if sent in top.keys():  
 print(sent)

Astronomers used ground-based telescopes to measure how Dimorphos’ orbit changed after impact.  
The DART spacecraft changed the moonlet asteroid’s orbit by 32 minutes.  
Initially, astronomers expected DART to be a success if it shortened the trajectory by 10 minutes.  
“This mission shows that NASA is trying to be ready for whatever the universe throws at us.

[previous](http://docs.google.com/markdown.html)

[Markdown Files](http://docs.google.com/markdown.html)

[next](http://docs.google.com/markdown-notebooks.html)

[Notebooks with MyST Markdown](http://docs.google.com/markdown-notebooks.html)

By The Jupyter Book Community

© Copyright 2022.