CS-E4650 Methods of Data Mining

Exercise 5.4 Topics of text clusters

:

Group members

Nguyen Xuan Binh (887799)

Erald Shahinas (906845)

Alexander Pavlyuk (906829)

Table of Contents

Learning goal: Clustering text data and techniques for describing topics of clusters

In this task, you should cluster a collection of short scientific texts and identify the main topics of each cluster. Ideally, you will indentify 3–10 unique topics (areas or techniques of computer science) that describe majority of documents excluding possible outliers. In MC, you can find data set scopusabstracts.csv, which consists of abstracts of scientific papers from Scopus https://www.elsevier.com/products/ scopus. Each line describes one document: its id, title, and abtract, separated by #.

1. Preprocessing

(a) In the baseline solution, combine the title and abstract. Preprocess the data like in the previous task, but this time, create also **bigrams** (in addition to unigrams) as possible features. Since the number of features would otherwise be too high, it is suggested to use frequency-based filtering to prune out very frequent or extremely rare words/collocations (see parameters of sklearn TfidfVectorizer). Consider also adding new stopwords, if any frequent but uninformative words complicate later steps. When features are fine, present the data in the tf-idf form so that each document vector is normalized to unit L2 norm.

Describe briefly the preprocessing methods: tools (like nltk), in which order the steps were performed, stemmer, stopword list (including own additions), tf-idf version (equation), minimum or maximum frequencies (if any), and other possible steps or options that could affect the results.

All the calculations have been perfomed on JypyterHub (https://jupyter.cs.aalto.fi) in the Python notebook. Additionally, numpy (https://numpy.org/), matplotlib (https://matplotlib.org/), and pandas (https://pandas.pydata.org/) libraries have been imported to handle specific functions.

In [ ]:
```
!pip install nltk
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages
(3.8.1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages
(from nltk) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages
(from nltk) (1.3.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-
packages (from nltk) (2023.6.3)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (f
rom nltk) (4.66.1)
```

In [ ]:
```
import pandas as pd

import re
import nltk
from nltk.tokenize import word_tokenize
from nltk.stem import PorterStemmer
from nltk.stem.snowball import SnowballStemmer
from nltk.corpus import stopwords
```

```
from nltk.probability import FreqDist
from sklearn.feature_extraction.text import TfidfVectorizer
from string import punctuation
import numpy as np

from sklearn.metrics import davies_bouldin_score
from sklearn.cluster import KMeans
from sklearn.decomposition import TruncatedSVD

from copy import deepcopy
nltk.download('stopwords')
nltk.download('punkt')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

Out[ ]: True

In [ ]:

```
# Load data
data_path = 'scopusabstracts.csv'

# Read the data
scopusdata = pd.read_csv('scopusabstracts.csv', sep='#')

# Extract the text from each line. In the baseline solution, combine the title a
corpus = [str1 + " " + str2 for str1, str2 in zip(scopusdata['TITLE'].to_list(),

# some examples
print('10 first titles + abstracts:')
for i in corpus[:10]:
    print(i)
print()
```

10 first titles + abstracts:
Anomaly detection in wide area imagery [Geniş alan görüntülerinde anomali tespit
i] This study is about detecting anomalies in wide area imagery collected from an
aircraft. The set of anomalies have been identified as anything out of the normal
course of action. For this purpose, two different data sets were used and the exp
eriments were carried out on these data sets. For anomaly detection, a convolutio
nal neural network model that tries to generate the next image using past images
is designed. The images were pre-processed before being given to the model. Anoma
ly detection is performed by comparing the estimated image and the true image.
Person re-identification with deep kronecker-product matching and group-shuffling
random walk Person re-identification (re-ID) aims to robustly measure visual affi
nities between person images. It has wide applications in intelligent surveillanc
e by associating same persons' images across multiple cameras. It is generally tr
eated as an image retrieval problem: Given a probe person image, the affinities b
etween the probe image and gallery images (P2G affinities) are used to rank the r
etrieved gallery images. There exist two main challenges for effectively solving
this problem. 1) Person images usually show significant variations because of dif
ferent person poses and viewing angles. The spatial layouts and correspondences b
etween person images are therefore vital information for tackling this problem. S
tate-of-the-art methods either ignore such spatial variation or utilize extra pos
e information for handling the challenge. 2) Most existing person re-ID methods r
ank gallery images considering only P2G affinities but ignore the affinities betw
een the gallery images (G2G affinity). Such affinities could provide important cl
ues for accurate gallery image ranking but were only utilized in post-processing
stages by current methods. In this article, we propose a unified end-to-end deep
learning framework to tackle the two challenges. For handling viewpoint and pose
variations between compared person images, we propose a novel Kronecker Product M
atching operation to match and warp feature maps of different persons. Comparing
warped feature maps results in more accurate P2G affinities. To fully utilize all
available P2G and G2G affinities for accurately ranking gallery person images, a
novel group-shuffling random walk operation is proposed. Both Kronecker Product M
atching and Group-shuffling Random Walk operations are end-to-end trainable and a
re shown to improve the learned visual features if integrated in the deep learnin
g framework. The proposed approach outperforms state-of-the-art methods on Market
-1501, CUHK03 and DukeMTMC datasets, which demonstrates the effectiveness and gen
eralization ability of our proposed approach. Code is available at https://githu
b.com/YantaoShen/kpm_rw_person_reid.
Crack detection in images of masonry using cnns While there is a significant body
of research on crack detection by computer vision methods in concrete and asphal
t, less attention has been given to masonry. We train a convolutional neural netw
ork (CNN) on images of brick walls built in a laboratory environment and test its
ability to detect cracks in images of brick-and-mortar structures both in the lab
oratory and on real-world images taken from the internet. We also compare the per
formance of the CNN to a variety of simpler classifiers operating on handcrafted
features. We find that the CNN performed better on the domain adaptation from lab
oratory to real-world images than these simple models. However, we also find that
performance is significantly better in performing the reverse domain adaptation t
ask, where the simple classifiers are trained on real-world images and tested on
the laboratory images. This work demonstrates the ability to detect cracks in ima
ges of masonry using a variety of machine learning methods and provides guidance
for improving the reliability of such models when performing domain adaptation fo
r crack detection in masonry.
Towards an energy efficient code generator for mobile phones Using a smartphone b
ecome the part of our everyday life in the last few years. These devices can help
us in many areas of life (sport, job, weather etc.), but sometimes can be also ve
ry annoying because of the battery life time. That is why it is very important to
find solutions, which can reduce the energy consumption of the smartphones. One p
ossible method is the 'computation offloading' where a part of the processes are
executed on a remote device (e.g. in the cloud). A lot of example has already sho
wn, that computation offloading can reduce the energy usage of the mobile device

s. However, the amount of energy saving may differ, as the decision making of the offloading process can be controlled with several techniques. Our decision making theory is based on the scheduling theory. In this paper, we are going to introduce a new system called ECGM (Energy efficient Code Generator for Mobile phones). ECGM decides automatically at compile time, which task should run on the smartphone and which task can be offloaded. The benefit of our system will be demonstrated through measurements based on our energy-efficiency scheduling technique.
Sub-polyhedral scheduling using (Unit-)two-variable-per-inequality polyhedra Polyhedral compilation has been successful in the design and implementation of complex loop nest optimizers and parallelizing compilers. The algorithmic complexity and scalability limitations remain one important weakness. We address it using sub-polyhedral under-aproximations of the systems of constraints resulting from affine scheduling problems. We propose a sub-polyhedral scheduling technique using (Unit-)Two-Variable-Per-Inequality or (U)TVPI Polyhedra. This technique relies on simple polynomial time algorithms to under-approximate a general polyhedron into (U)TVPI polyhedra. We modify the state-of-the-art PLuTo compiler using our scheduling technique, and show that for a majority of the Polybench (2.0) kernels, the above under-approximations yield polyhedra that are non-empty. Solving the under-approximated system leads to asymptotic gains in complexity, and shows practically significant improvements when compared to a traditional LP solver. We also verify that code generated by our sub-polyhedral parallelization prototype matches the performance of PLuTo-optimized code when the under-approximation preserves feasibility. Copyright
Extracting multiple viewpoint models from relational databases Much time in process mining projects is spent on finding and understanding data sources and extracting the event data needed. As a result, only a fraction of time is spent actually applying techniques to discover, control and predict the business process. Moreover, current process mining techniques assume a single case notion. However, in real-life processes often different case notions are intertwined. For example, events of the same order handling process may refer to customers, orders, order lines, deliveries, and payments. Therefore, we propose to use Multiple Viewpoint (MVP) models that relate events through objects and that relate activities through classes. The required event data are much closer to existing relational databases. MVP models provide a holistic view on the process, but also allow for the extraction of classical event logs using different viewpoints. This way existing process mining techniques can be used for each viewpoint without the need for new data extractions and transformations. We provide a toolchain allowing for the discovery of MVP models (annotated with performance and frequency information) from relational databases. Moreover, we demonstrate that classical process mining techniques can be applied to any selected viewpoint.
A Program Result Checker for the Lexical Analysis of the GNU C Compiler In theory, program result checking has been established as a well-suited method to construct formally correct compiler frontends but it has never proved its practicality for real-life compilers. Such a proof is necessary to establish result checking as the method of choice to implement compilers correctly. We show that the lexical analysis of the GNU C compiler can be formally specified and checked within the theorem prover Isabelle/HOL utilizing program checking. Thereby we demonstrate that formal specification and verification techniques are able to handle real-life compilers.
Advances in visual object tracking algorithm based on correlation filter [基于相关滤波的视觉目标跟踪算法新进展] With excellent comprehensive performance, correlation filter-based tracking algorithms have become a hotspot of the theoretical research and practical application in the field of visual object tracking. Despite many studies, there is still a lack of systematic analyses on the existing correlation filter-based tracking algorithms from the level of tracking framework. Therefore, starting from the basic framework of object tracking algorithms, the characteristics of correlation filter-based tracking algorithms are deeply analyzed, and their basic problems in each working stage are presented in this paper. On this basis, the main technological progress of correlation filter-based tracking algorithms and characteristics of corresponding algorithms in recent ten years are summari

```
zed, and 20 typical correlation filter-based tracking algorithms are evaluated an
d analyzed. Finally, the outstanding issues to be urgently solved and the future
research directions of correlation filter-based tracking algorithms are discusse
d.
A Study on Various Database Models: Relational, Graph, and Hybrid Databases Relat
ional database is a popular database for storing various types of information. Bu
t due to the ever-increasing growth of data, it becomes hard to maintain and proc
ess the database. So, the graph model is becoming more and more popular since it
can store and handle big data more efficiently compared to relational database. B
ut both relational database and graph database have their own advantages and disa
dvantages. To overcome their limitations, they are combined to make a hybrid mode
l. This paper discusses relational database, graph database, their advantages, th
eir applications and also talks about hybrid model.
Better Reporting of Studies on Artificial Intelligence: CONSORT-AI and Beyond An
increasing number of studies on artificial intelligence (AI) are published in the
dental and oral sciences. The reporting, but also further aspects of these studie
s, suffer from a range of limitations. Standards towards reporting, like the rece
ntly published Consolidated Standards of Reporting Trials (CONSORT)-AI extension
can help to improve studies in this emerging field, and the Journal of Dental Res
earch (JDR) encourages authors, reviewers, and readers to adhere to these standar
ds. Notably, though, a wide range of aspects beyond reporting, located along vari
ous steps of the AI lifecycle, should be considered when conceiving, conducting,
reporting, or evaluating studies on AI in dentistry.
```

In [ ]:

```
# Preprocessing

# Step 1: tokenization and lowercasing
tokens_list = [word_tokenize(document) for document in corpus]

lc_tokens_list = []

for token_document in tokens_list:
    lc_tokens_list.append([token.lower() for token in token_document])

print('After tokenization and lowercasing:\n')
for i in lc_tokens_list[:10]:
    print(i)
print()

# original number of tokens
uniques = np.unique([token for token_document in lc_tokens_list for token in tok
print("Original number of tokens: {}\n".format(len(uniques)))
```

```
After tokenization and lowercasing:

['anomaly', 'detection', 'in', 'wide', 'area', 'imagery', '[', 'geniş', 'alan',
'görüntülerinde', 'anomali', 'tespiti', ']', 'this', 'study', 'is', 'about', 'det
ecting', 'anomalies', 'in', 'wide', 'area', 'imagery', 'collected', 'from', 'an',
'aircraft', '.', 'the', 'set', 'of', 'anomalies', 'have', 'been', 'identified',
'as', 'anything', 'out', 'of', 'the', 'normal', 'course', 'of', 'action', '.', 'f
or', 'this', 'purpose', ',', 'two', 'different', 'data', 'sets', 'were', 'used',
'and', 'the', 'experiments', 'were', 'carried', 'out', 'on', 'these', 'data', 'se
ts', '.', 'for', 'anomaly', 'detection', ',', 'a', 'convolutional', 'neural', 'ne
twork', 'model', 'that', 'tries', 'to', 'generate', 'the', 'next', 'image', 'usin
g', 'past', 'images', 'is', 'designed', '.', 'the', 'images', 'were', 'pre-proces
sed', 'before', 'being', 'given', 'to', 'the', 'model', '.', 'anomaly', 'detectio
n', 'is', 'performed', 'by', 'comparing', 'the', 'estimated', 'image', 'and', 'th
e', 'true', 'image', '.']
['person', 're-identification', 'with', 'deep', 'kronecker-product', 'matching',
'and', 'group-shuffling', 'random', 'walk', 'person', 're-identification', '(',
're-id', ')', 'aims', 'to', 'robustly', 'measure', 'visual', 'affinities', 'betwe
en', 'person', 'images', '.', 'it', 'has', 'wide', 'applications', 'in', 'intelli
gent', 'surveillance', 'by', 'associating', 'same', 'persons', "'", 'images', 'ac
ross', 'multiple', 'cameras', '.', 'it', 'is', 'generally', 'treated', 'as', 'a
n', 'image', 'retrieval', 'problem', ':', 'given', 'a', 'probe', 'person', 'imag
e', ',', 'the', 'affinities', 'between', 'the', 'probe', 'image', 'and', 'galler
y', 'images', '(', 'p2g', 'affinities', ')', 'are', 'used', 'to', 'rank', 'the',
'retrieved', 'gallery', 'images', '.', 'there', 'exist', 'two', 'main', 'challeng
es', 'for', 'effectively', 'solving', 'this', 'problem', '.', '1', ')', 'person',
'images', 'usually', 'show', 'significant', 'variations', 'because', 'of', 'diffe
rent', 'person', 'poses', 'and', 'viewing', 'angles', '.', 'the', 'spatial', 'lay
outs', 'and', 'correspondences', 'between', 'person', 'images', 'are', 'therefor
e', 'vital', 'information', 'for', 'tackling', 'this', 'problem', '.', 'state-of-
the-art', 'methods', 'either', 'ignore', 'such', 'spatial', 'variation', 'or', 'u
tilize', 'extra', 'pose', 'information', 'for', 'handling', 'the', 'challenge',
'.', '2', ')', 'most', 'existing', 'person', 're-id', 'methods', 'rank', 'galler
y', 'images', 'considering', 'only', 'p2g', 'affinities', 'but', 'ignore', 'the',
'affinities', 'between', 'the', 'gallery', 'images', '(', 'g2g', 'affinity', ')',
'.', 'such', 'affinities', 'could', 'provide', 'important', 'clues', 'for', 'accu
rate', 'gallery', 'image', 'ranking', 'but', 'were', 'only', 'utilized', 'in', 'p
ost-processing', 'stages', 'by', 'current', 'methods', '.', 'in', 'this', 'articl
e', ',', 'we', 'propose', 'a', 'unified', 'end-to-end', 'deep', 'learning', 'fram
ework', 'to', 'tackle', 'the', 'two', 'challenges', '.', 'for', 'handling', 'view
point', 'and', 'pose', 'variations', 'between', 'compared', 'person', 'images',
',', 'we', 'propose', 'a', 'novel', 'kronecker', 'product', 'matching', 'operatio
n', 'to', 'match', 'and', 'warp', 'feature', 'maps', 'of', 'different', 'person
s', '.', 'comparing', 'warped', 'feature', 'maps', 'results', 'in', 'more', 'accu
rate', 'p2g', 'affinities', '.', 'to', 'fully', 'utilize', 'all', 'available', 'p
2g', 'and', 'g2g', 'affinities', 'for', 'accurately', 'ranking', 'gallery', 'pers
on', 'images', ',', 'a', 'novel', 'group-shuffling', 'random', 'walk', 'operatio
n', 'is', 'proposed', '.', 'both', 'kronecker', 'product', 'matching', 'and', 'gr
oup-shuffling', 'random', 'walk', 'operations', 'are', 'end-to-end', 'trainable',
'and', 'are', 'shown', 'to', 'improve', 'the', 'learned', 'visual', 'features',
'if', 'integrated', 'in', 'the', 'deep', 'learning', 'framework', '.', 'the', 'pr
oposed', 'approach', 'outperforms', 'state-of-the-art', 'methods', 'on', 'market-
1501', ',', 'cuhk03', 'and', 'dukemtmc', 'datasets', ',', 'which', 'demonstrate
s', 'the', 'effectiveness', 'and', 'generalization', 'ability', 'of', 'our', 'pro
posed', 'approach', '.', 'code', 'is', 'available', 'at', 'https', ':', '//githu
b.com/yantaoshen/kpm_rw_person_reid', '.']
['crack', 'detection', 'in', 'images', 'of', 'masonry', 'using', 'cnns', 'while',
'there', 'is', 'a', 'significant', 'body', 'of', 'research', 'on', 'crack', 'dete
ction', 'by', 'computer', 'vision', 'methods', 'in', 'concrete', 'and', 'asphal
t', ',', 'less', 'attention', 'has', 'been', 'given', 'to', 'masonry', '.', 'we',
```

'train', 'a', 'convolutional', 'neural', 'network', '(', 'cnn', ')', 'on', 'image
s', 'of', 'brick', 'walls', 'built', 'in', 'a', 'laboratory', 'environment', 'an
d', 'test', 'its', 'ability', 'to', 'detect', 'cracks', 'in', 'images', 'of', 'br
ick-and-mortar', 'structures', 'both', 'in', 'the', 'laboratory', 'and', 'on', 'r
eal-world', 'images', 'taken', 'from', 'the', 'internet', '.', 'we', 'also', 'com
pare', 'the', 'performance', 'of', 'the', 'cnn', 'to', 'a', 'variety', 'of', 'sim
pler', 'classifiers', 'operating', 'on', 'handcrafted', 'features', '.', 'we', 'f
ind', 'that', 'the', 'cnn', 'performed', 'better', 'on', 'the', 'domain', 'adapta
tion', 'from', 'laboratory', 'to', 'real-world', 'images', 'than', 'these', 'simp
le', 'models', '.', 'however', ',', 'we', 'also', 'find', 'that', 'performance',
'is', 'significantly', 'better', 'in', 'performing', 'the', 'reverse', 'domain',
'adaptation', 'task', ',', 'where', 'the', 'simple', 'classifiers', 'are', 'train
ed', 'on', 'real-world', 'images', 'and', 'tested', 'on', 'the', 'laboratory', 'i
mages', '.', 'this', 'work', 'demonstrates', 'the', 'ability', 'to', 'detect', 'c
racks', 'in', 'images', 'of', 'masonry', 'using', 'a', 'variety', 'of', 'machin
e', 'learning', 'methods', 'and', 'provides', 'guidance', 'for', 'improving', 'th
e', 'reliability', 'of', 'such', 'models', 'when', 'performing', 'domain', 'adapt
ation', 'for', 'crack', 'detection', 'in', 'masonry', '.']
['towards', 'an', 'energy', 'efficient', 'code', 'generator', 'for', 'mobile', 'p
hones', 'using', 'a', 'smartphone', 'become', 'the', 'part', 'of', 'our', 'everyd
ay', 'life', 'in', 'the', 'last', 'few', 'years', '.', 'these', 'devices', 'can',
'help', 'us', 'in', 'many', 'areas', 'of', 'life', '(', 'sport', ',', 'job', ',',
'weather', 'etc', '.', ')', ',', 'but', 'sometimes', 'can', 'be', 'also', 'very',
'annoying', 'because', 'of', 'the', 'battery', 'life', 'time', '.', 'that', 'is',
'why', 'it', 'is', 'very', 'important', 'to', 'find', 'solutions', ',', 'which',
'can', 'reduce', 'the', 'energy', 'consumption', 'of', 'the', 'smartphones', '.',
'one', 'possible', 'method', 'is', 'the', "'computation", 'offloading', "'", 'whe
re', 'a', 'part', 'of', 'the', 'processes', 'are', 'executed', 'on', 'a', 'remot
e', 'device', '(', 'e.g', '.', 'in', 'the', 'cloud', ')', '.', 'a', 'lot', 'of',
'example', 'has', 'already', 'shown', ',', 'that', 'computation', 'offloading',
'can', 'reduce', 'the', 'energy', 'usage', 'of', 'the', 'mobile', 'devices', '.',
'however', ',', 'the', 'amount', 'of', 'energy', 'saving', 'may', 'differ', ',',
'as', 'the', 'decision', 'making', 'of', 'the', 'offloading', 'process', 'can',
'be', 'controlled', 'with', 'several', 'techniques', '.', 'our', 'decision', 'mak
ing', 'theory', 'is', 'based', 'on', 'the', 'scheduling', 'theory', '.', 'in', 't
his', 'paper', ',', 'we', 'are', 'going', 'to', 'introduce', 'a', 'new', 'syste
m', 'called', 'ecgm', '(', 'energy', 'efficient', 'code', 'generator', 'for', 'mo
bile', 'phones', ')', '.', 'ecgm', 'decides', 'automatically', 'at', 'compile',
'time', ',', 'which', 'task', 'should', 'run', 'on', 'the', 'smartphone', 'and',
'which', 'task', 'can', 'be', 'offloaded', '.', 'the', 'benefit', 'of', 'our', 's
ystem', 'will', 'be', 'demonstrated', 'through', 'measurements', 'based', 'on',
'our', 'energy-efficiency', 'scheduling', 'technique', '.']
['sub-polyhedral', 'scheduling', 'using', '(', 'unit-', ')', 'two-variable-per-in
equality', 'polyhedra', 'polyhedral', 'compilation', 'has', 'been', 'successful',
'in', 'the', 'design', 'and', 'implementation', 'of', 'complex', 'loop', 'nest',
'optimizers', 'and', 'parallelizing', 'compilers', '.', 'the', 'algorithmic', 'co
mplexity', 'and', 'scalability', 'limitations', 'remain', 'one', 'important', 'we
akness', '.', 'we', 'address', 'it', 'using', 'sub-polyhedral', 'under-aproximati
ons', 'of', 'the', 'systems', 'of', 'constraints', 'resulting', 'from', 'affine',
'scheduling', 'problems', '.', 'we', 'propose', 'a', 'sub-polyhedral', 'schedulin
g', 'technique', 'using', '(', 'unit-', ')', 'two-variable-per-inequality', 'or',
'(', 'u', ')', 'tvpi', 'polyhedra', '.', 'this', 'technique', 'relies', 'on', 'si
mple', 'polynomial', 'time', 'algorithms', 'to', 'under-approximate', 'a', 'gener
al', 'polyhedron', 'into', '(', 'u', ')', 'tvpi', 'polyhedra', '.', 'we', 'modif
y', 'the', 'state-of-the-art', 'pluto', 'compiler', 'using', 'our', 'scheduling',
'technique', ',', 'and', 'show', 'that', 'for', 'a', 'majority', 'of', 'the', 'po
lybench', '(', '2.0', ')', 'kernels', ',', 'the', 'above', 'under-approximation
s', 'yield', 'polyhedra', 'that', 'are', 'non-empty', '.', 'solving', 'the', 'und
er-approximated', 'system', 'leads', 'to', 'asymptotic', 'gains', 'in', 'complexi
ty', ',', 'and', 'shows', 'practically', 'significant', 'improvements', 'when',

'compared', 'to', 'a', 'traditional', 'lp', 'solver', '.', 'we', 'also', 'verif
y', 'that', 'code', 'generated', 'by', 'our', 'sub-polyhedral', 'parallelizatio
n', 'prototype', 'matches', 'the', 'performance', 'of', 'pluto-optimized', 'cod
e', 'when', 'the', 'under-approximation', 'preserves', 'feasibility', '.', 'copyr
ight']
['extracting', 'multiple', 'viewpoint', 'models', 'from', 'relational', 'database
s', 'much', 'time', 'in', 'process', 'mining', 'projects', 'is', 'spent', 'on',
'finding', 'and', 'understanding', 'data', 'sources', 'and', 'extracting', 'the',
'event', 'data', 'needed', '.', 'as', 'a', 'result', ',', 'only', 'a', 'fractio
n', 'of', 'time', 'is', 'spent', 'actually', 'applying', 'techniques', 'to', 'dis
cover', ',', 'control', 'and', 'predict', 'the', 'business', 'process', '.', 'mor
eover', ',', 'current', 'process', 'mining', 'techniques', 'assume', 'a', 'singl
e', 'case', 'notion', '.', 'however', ',', 'in', 'real-life', 'processes', 'ofte
n', 'different', 'case', 'notions', 'are', 'intertwined', '.', 'for', 'example',
',', 'events', 'of', 'the', 'same', 'order', 'handling', 'process', 'may', 'refe
r', 'to', 'customers', ',', 'orders', ',', 'order', 'lines', ',', 'deliveries',
',', 'and', 'payments', '.', 'therefore', ',', 'we', 'propose', 'to', 'use', 'mul
tiple', 'viewpoint', '(', 'mvp', ')', 'models', 'that', 'relate', 'events', 'thro
ugh', 'objects', 'and', 'that', 'relate', 'activities', 'through', 'classes',
'.', 'the', 'required', 'event', 'data', 'are', 'much', 'closer', 'to', 'existin
g', 'relational', 'databases', '.', 'mvp', 'models', 'provide', 'a', 'holistic',
'view', 'on', 'the', 'process', ',', 'but', 'also', 'allow', 'for', 'the', 'extra
ction', 'of', 'classical', 'event', 'logs', 'using', 'different', 'viewpoints',
'.', 'this', 'way', 'existing', 'process', 'mining', 'techniques', 'can', 'be',
'used', 'for', 'each', 'viewpoint', 'without', 'the', 'need', 'for', 'new', 'dat
a', 'extractions', 'and', 'transformations', '.', 'we', 'provide', 'a', 'toolchai
n', 'allowing', 'for', 'the', 'discovery', 'of', 'mvp', 'models', '(', 'annotate
d', 'with', 'performance', 'and', 'frequency', 'information', ')', 'from', 'relat
ional', 'databases', '.', 'moreover', ',', 'we', 'demonstrate', 'that', 'classica
l', 'process', 'mining', 'techniques', 'can', 'be', 'applied', 'to', 'any', 'sele
cted', 'viewpoint', '.']
['a', 'program', 'result', 'checker', 'for', 'the', 'lexical', 'analysis', 'of',
'the', 'gnu', 'c', 'compiler', 'in', 'theory', ',', 'program', 'result', 'checkin
g', 'has', 'been', 'established', 'as', 'a', 'well-suited', 'method', 'to', 'cons
truct', 'formally', 'correct', 'compiler', 'frontends', 'but', 'it', 'has', 'neve
r', 'proved', 'its', 'practicality', 'for', 'real-life', 'compilers', '.', 'suc
h', 'a', 'proof', 'is', 'necessary', 'to', 'establish', 'result', 'checking', 'a
s', 'the', 'method', 'of', 'choice', 'to', 'implement', 'compilers', 'correctly',
'.', 'we', 'show', 'that', 'the', 'lexical', 'analysis', 'of', 'the', 'gnu', 'c',
'compiler', 'can', 'be', 'formally', 'specified', 'and', 'checked', 'within', 'th
e', 'theorem', 'prover', 'isabelle/hol', 'utilizing', 'program', 'checking', '.',
'thereby', 'we', 'demonstrate', 'that', 'formal', 'specification', 'and', 'verifi
cation', 'techniques', 'are', 'able', 'to', 'handle', 'real-life', 'compilers',
'.']
['advances', 'in', 'visual', 'object', 'tracking', 'algorithm', 'based', 'on', 'c
orrelation', 'filter', '[', '基于相关滤波的视觉目标跟踪算法新进展', ']', 'with', 'ex
cellent', 'comprehensive', 'performance', ',', 'correlation', 'filter-based', 'tr
acking', 'algorithms', 'have', 'become', 'a', 'hotspot', 'of', 'the', 'theoretica
l', 'research', 'and', 'practical', 'application', 'in', 'the', 'field', 'of', 'v
isual', 'object', 'tracking', '.', 'despite', 'many', 'studies', ',', 'there', 'i
s', 'still', 'a', 'lack', 'of', 'systematic', 'analyses', 'on', 'the', 'existin
g', 'correlation', 'filter-based', 'tracking', 'algorithms', 'from', 'the', 'leve
l', 'of', 'tracking', 'framework', '.', 'therefore', ',', 'starting', 'from', 'th
e', 'basic', 'framework', 'of', 'object', 'tracking', 'algorithms', ',', 'the',
'characteristics', 'of', 'correlation', 'filter-based', 'tracking', 'algorithms',
'are', 'deeply', 'analyzed', ',', 'and', 'their', 'basic', 'problems', 'in', 'eac
h', 'working', 'stage', 'are', 'presented', 'in', 'this', 'paper', '.', 'on', 'th
is', 'basis', ',', 'the', 'main', 'technological', 'progress', 'of', 'correlatio
n', 'filter-based', 'tracking', 'algorithms', 'and', 'characteristics', 'of', 'co
rresponding', 'algorithms', 'in', 'recent', 'ten', 'years', 'are', 'summarized',

```
',', 'and', '20', 'typical', 'correlation', 'filter-based', 'tracking', 'algorith
ms', 'are', 'evaluated', 'and', 'analyzed', '.', 'finally', ',', 'the', 'outstand
ing', 'issues', 'to', 'be', 'urgently', 'solved', 'and', 'the', 'future', 'resear
ch', 'directions', 'of', 'correlation', 'filter-based', 'tracking', 'algorithms',
'are', 'discussed', '.']
['a', 'study', 'on', 'various', 'database', 'models', ':', 'relational', ',', 'gr
aph', ',', 'and', 'hybrid', 'databases', 'relational', 'database', 'is', 'a', 'po
pular', 'database', 'for', 'storing', 'various', 'types', 'of', 'information',
'.', 'but', 'due', 'to', 'the', 'ever-increasing', 'growth', 'of', 'data', ',',
'it', 'becomes', 'hard', 'to', 'maintain', 'and', 'process', 'the', 'database',
'.', 'so', ',', 'the', 'graph', 'model', 'is', 'becoming', 'more', 'and', 'more',
'popular', 'since', 'it', 'can', 'store', 'and', 'handle', 'big', 'data', 'more',
'efficiently', 'compared', 'to', 'relational', 'database', '.', 'but', 'both', 'r
elational', 'database', 'and', 'graph', 'database', 'have', 'their', 'own', 'adva
ntages', 'and', 'disadvantages', '.', 'to', 'overcome', 'their', 'limitations',
',', 'they', 'are', 'combined', 'to', 'make', 'a', 'hybrid', 'model', '.', 'thi
s', 'paper', 'discusses', 'relational', 'database', ',', 'graph', 'database',
',', 'their', 'advantages', ',', 'their', 'applications', 'and', 'also', 'talks',
'about', 'hybrid', 'model', '.']
['better', 'reporting', 'of', 'studies', 'on', 'artificial', 'intelligence', ':',
'consort-ai', 'and', 'beyond', 'an', 'increasing', 'number', 'of', 'studies', 'o
n', 'artificial', 'intelligence', '(', 'ai', ')', 'are', 'published', 'in', 'th
e', 'dental', 'and', 'oral', 'sciences', '.', 'the', 'reporting', ',', 'but', 'al
so', 'further', 'aspects', 'of', 'these', 'studies', ',', 'suffer', 'from', 'a',
'range', 'of', 'limitations', '.', 'standards', 'towards', 'reporting', ',', 'lik
e', 'the', 'recently', 'published', 'consolidated', 'standards', 'of', 'reportin
g', 'trials', '(', 'consort', ')', '-ai', 'extension', 'can', 'help', 'to', 'impr
ove', 'studies', 'in', 'this', 'emerging', 'field', ',', 'and', 'the', 'journal',
'of', 'dental', 'research', '(', 'jdr', ')', 'encourages', 'authors', ',', 'revie
wers', ',', 'and', 'readers', 'to', 'adhere', 'to', 'these', 'standards', '.', 'n
otably', ',', 'though', ',', 'a', 'wide', 'range', 'of', 'aspects', 'beyond', 're
porting', ',', 'located', 'along', 'various', 'steps', 'of', 'the', 'ai', 'lifecy
cle', ',', 'should', 'be', 'considered', 'when', 'conceiving', ',', 'conducting',
',', 'reporting', ',', 'or', 'evaluating', 'studies', 'on', 'ai', 'in', 'dentistr
y', '.']

Original number of tokens: 15882
```

In [ ]:

```
# Steps 2 and 3: remove stop words and punctuation
stop_words = set(stopwords.words('english'))
print('NLTK stopwords:')
print(stop_words)
print()

#stop_words.update(["use", "data", "system", "propos"])

# Here we include the punctuation in the stop words set. There are alternative w


stop_words.update(punctuation)

# For the field of computer science research papers, in addition to the standard
# we might consider adding words that serves little meanings

# stop_words.update({"use", "data", "system", "proposed", "study", "results", "a
#                    "approach", "methods", "research", "application", "technique",
#                    "algorithm", "process", "problem", "solution"})

#you can check updated stopwords
```

```
#print(stop_words)

filtered_sentence = []
for i in lc_tokens_list:
    filtered_sentence.append([token for token in i if token not in stop_words])

# Numbers are also removed
filtered_sentence = [ ' '.join(i) for i in filtered_sentence ]
filtered_sentence = [ re.sub(r'\d+', '', sentence) for sentence in filtered_sent

# number of tokens
uniques = np.unique([tok for doc in filtered_sentence for tok in doc.split()])
print("Number of tokens after stopword and punctuation removal: {}\n".format(len


print('After removing stop words, punctuation and numbers:')
for sentence in filtered_sentence[:10]:
    print(sentence)
print()
```

```
NLTK stopwords:
{'mightn', 'each', "wouldn't", 'just', "wasn't", 'of', 'both', "hadn't", 's', 'no
w', 'isn', 'don', 'itself', 'through', 'herself', 'how', "it's", 'doesn', 'm', 'h
aven', 'we', 'as', 'up', 'theirs', "shouldn't", "isn't", 'an', 'so', 'which', 'b
y', 'be', 'further', 'having', 'was', "mustn't", 'again', 'before', "you'd", 'ou
t', 'yourselves', 'have', 'such', 'hasn', "weren't", 'she', 'there', 'from', 'onc
e', 'yourself', 'until', 'between', 'to', "you're", 'after', 'it', 'down', 'onl
y', 'then', 'against', 'yours', 'those', 'themselves', 're', 'own', 've', 'a', 'i
n', 'do', 'are', 'whom', 'with', 'were', 'they', 'when', 'where', "won't", "has
n't", 'any', 'off', 'no', 'hadn', 'about', 'll', 'am', 'all', 'than', "you've",
'most', 'he', 'being', 'does', 'ours', 'y', "couldn't", "don't", 'been', 'the',
"doesn't", 'shan', "didn't", 'wasn', "that'll", 'them', 'if', 'that', 'or', 'unde
r', 'their', 'while', 'o', 'will', 'should', "mightn't", 'mustn', 'your', 'could
n', 'shouldn', 'very', 'why', 'ma', 'doing', 'same', "you'll", 'did', 'what', 'to
o', 'needn', 'few', 'him', 'can', "needn't", 'but', 'at', 'nor', 'wouldn', 'mor
e', 'is', 'won', 'myself', 'me', 'its', 'd', 'weren', 'ourselves', 'into', "are
n't", 'hers', 'during', 'you', 'on', "shan't", 'this', 'aren', "haven't", 'has',
'some', 'these', 'other', 'himself', 'and', 'over', 'i', 'ain', 'because', 'for',
't', 'here', 'didn', 'above', 'below', "should've", "she's", 'my', 'who', 'his',
'our', 'her', 'not', 'had'}

Number of tokens after stopword and punctuation removal: 14820

After removing stop words, punctuation and numbers:
anomaly detection wide area imagery geniş alan görüntülerinde anomali tespiti stu
dy detecting anomalies wide area imagery collected aircraft set anomalies identif
ied anything normal course action purpose two different data sets used experiment
s carried data sets anomaly detection convolutional neural network model tries ge
nerate next image using past images designed images pre-processed given model ano
maly detection performed comparing estimated image true image
person re-identification deep kronecker-product matching group-shuffling random w
alk person re-identification re-id aims robustly measure visual affinities person
images wide applications intelligent surveillance associating persons images acro
ss multiple cameras generally treated image retrieval problem given probe person
image affinities probe image gallery images pg affinities used rank retrieved gal
lery images exist two main challenges effectively solving problem  person images
usually show significant variations different person poses viewing angles spatial
layouts correspondences person images therefore vital information tackling proble
m state-of-the-art methods either ignore spatial variation utilize extra pose inf
ormation handling challenge  existing person re-id methods rank gallery images co
nsidering pg affinities ignore affinities gallery images gg affinity affinities c
ould provide important clues accurate gallery image ranking utilized post-process
ing stages current methods article propose unified end-to-end deep learning frame
work tackle two challenges handling viewpoint pose variations compared person ima
ges propose novel kronecker product matching operation match warp feature maps di
fferent persons comparing warped feature maps results accurate pg affinities full
y utilize available pg gg affinities accurately ranking gallery person images nov
el group-shuffling random walk operation proposed kronecker product matching grou
p-shuffling random walk operations end-to-end trainable shown improve learned vis
ual features integrated deep learning framework proposed approach outperforms sta
te-of-the-art methods market- cuhk dukemtmc datasets demonstrates effectiveness g
eneralization ability proposed approach code available https //github.com/yantaos
hen/kpm_rw_person_reid
crack detection images masonry using cnns significant body research crack detecti
on computer vision methods concrete asphalt less attention given masonry train co
nvolutional neural network cnn images brick walls built laboratory environment te
st ability detect cracks images brick-and-mortar structures laboratory real-world
images taken internet also compare performance cnn variety simpler classifiers op
erating handcrafted features find cnn performed better domain adaptation laborato
ry real-world images simple models however also find performance significantly be
```

tter performing reverse domain adaptation task simple classifiers trained real-wo
rld images tested laboratory images work demonstrates ability detect cracks image
s masonry using variety machine learning methods provides guidance improving reli
ability models performing domain adaptation crack detection masonry
towards energy efficient code generator mobile phones using smartphone become par
t everyday life last years devices help us many areas life sport job weather etc
sometimes also annoying battery life time important find solutions reduce energy
consumption smartphones one possible method 'computation offloading part processe
s executed remote device e.g cloud lot example already shown computation offloadi
ng reduce energy usage mobile devices however amount energy saving may differ dec
ision making offloading process controlled several techniques decision making the
ory based scheduling theory paper going introduce new system called ecgm energy e
fficient code generator mobile phones ecgm decides automatically compile time tas
k run smartphone task offloaded benefit system demonstrated measurements based en
ergy-efficiency scheduling technique
sub-polyhedral scheduling using unit- two-variable-per-inequality polyhedra polyh
edral compilation successful design implementation complex loop nest optimizers p
arallelizing compilers algorithmic complexity scalability limitations remain one
important weakness address using sub-polyhedral under-aproximations systems const
raints resulting affine scheduling problems propose sub-polyhedral scheduling tec
hnique using unit- two-variable-per-inequality u tvpi polyhedra technique relies
simple polynomial time algorithms under-approximate general polyhedron u tvpi pol
yhedra modify state-of-the-art pluto compiler using scheduling technique show maj
ority polybench . kernels under-approximations yield polyhedra non-empty solving
under-approximated system leads asymptotic gains complexity shows practically sig
nificant improvements compared traditional lp solver also verify code generated s
ub-polyhedral parallelization prototype matches performance pluto-optimized code
under-approximation preserves feasibility copyright
extracting multiple viewpoint models relational databases much time process minin
g projects spent finding understanding data sources extracting event data needed
result fraction time spent actually applying techniques discover control predict
business process moreover current process mining techniques assume single case no
tion however real-life processes often different case notions intertwined example
events order handling process may refer customers orders order lines deliveries p
ayments therefore propose use multiple viewpoint mvp models relate events objects
relate activities classes required event data much closer existing relational dat
abases mvp models provide holistic view process also allow extraction classical e
vent logs using different viewpoints way existing process mining techniques used
viewpoint without need new data extractions transformations provide toolchain all
owing discovery mvp models annotated performance frequency information relational
databases moreover demonstrate classical process mining techniques applied select
ed viewpoint
program result checker lexical analysis gnu c compiler theory program result chec
king established well-suited method construct formally correct compiler frontends
never proved practicality real-life compilers proof necessary establish result ch
ecking method choice implement compilers correctly show lexical analysis gnu c co
mpiler formally specified checked within theorem prover isabelle/hol utilizing pr
ogram checking thereby demonstrate formal specification verification techniques a
ble handle real-life compilers
advances visual object tracking algorithm based correlation filter 基于相关滤波的视
觉目标跟踪算法新进展 excellent comprehensive performance correlation filter-based t
racking algorithms become hotspot theoretical research practical application fiel
d visual object tracking despite many studies still lack systematic analyses exis
ting correlation filter-based tracking algorithms level tracking framework theref
ore starting basic framework object tracking algorithms characteristics correlati
on filter-based tracking algorithms deeply analyzed basic problems working stage
presented paper basis main technological progress correlation filter-based tracki
ng algorithms characteristics corresponding algorithms recent ten years summarize
d typical correlation filter-based tracking algorithms evaluated analyzed finall
y outstanding issues urgently solved future research directions correlation filte

```
r-based tracking algorithms discussed
study various database models relational graph hybrid databases relational databa
se popular database storing various types information due ever-increasing growth
data becomes hard maintain process database graph model becoming popular since st
ore handle big data efficiently compared relational database relational database
graph database advantages disadvantages overcome limitations combined make hybrid
model paper discusses relational database graph database advantages applications
also talks hybrid model
better reporting studies artificial intelligence consort-ai beyond increasing num
ber studies artificial intelligence ai published dental oral sciences reporting a
lso aspects studies suffer range limitations standards towards reporting like rec
ently published consolidated standards reporting trials consort -ai extension hel
p improve studies emerging field journal dental research jdr encourages authors r
eviewers readers adhere standards notably though wide range aspects beyond report
ing located along various steps ai lifecycle considered conceiving conducting rep
orting evaluating studies ai dentistry
```

In [ ]:

```python
# Step 4: stemming
porter = PorterStemmer()

#or snowball stemmer
#stemmer = SnowballStemmer("english",ignore_stopwords=True)
stemmed_tokens_list = []

for i in filtered_sentence:
        stemmed_tokens_list.append([porter.stem(j) for j in i.split()])

# number of tokens
uniques = np.unique([tok for doc in stemmed_tokens_list for tok in doc])
print("Number of tokens after stemming: {}\n".format(len(uniques)))

print('After stemming:')
for tokens in stemmed_tokens_list[:10]:
        for token in tokens:
                print(token,end=" ")
        print(" ")
```

Number of tokens after stemming: 10259

After stemming:
anomali detect wide area imageri geniş alan görüntülerind anomali tespiti studi d
etect anomali wide area imageri collect aircraft set anomali identifi anyth norma
l cours action purpos two differ data set use experi carri data set anomali detec
t convolut neural network model tri gener next imag use past imag design imag pre
-process given model anomali detect perform compar estim imag true imag
person re-identif deep kronecker-product match group-shuffl random walk person re
-identif re-id aim robustli measur visual affin person imag wide applic intellig
surveil associ person imag across multipl camera gener treat imag retriev problem
given probe person imag affin probe imag galleri imag pg affin use rank retriev g
alleri imag exist two main challeng effect solv problem person imag usual show si
gnific variat differ person pose view angl spatial layout correspond person imag
therefor vital inform tackl problem state-of-the-art method either ignor spatial
variat util extra pose inform handl challeng exist person re-id method rank galle
ri imag consid pg affin ignor affin galleri imag gg affin affin could provid impo
rt clue accur galleri imag rank util post-process stage current method articl pro
pos unifi end-to-end deep learn framework tackl two challeng handl viewpoint pose
variat compar person imag propos novel kroneck product match oper match warp feat
ur map differ person compar warp featur map result accur pg affin fulli util avai
l pg gg affin accur rank galleri person imag novel group-shuffl random walk oper
propos kroneck product match group-shuffl random walk oper end-to-end trainabl sh
own improv learn visual featur integr deep learn framework propos approach outper
form state-of-the-art method market- cuhk dukemtmc dataset demonstr effect gener
abil propos approach code avail http //github.com/yantaoshen/kpm_rw_person_reid
crack detect imag masonri use cnn signific bodi research crack detect comput visi
on method concret asphalt less attent given masonri train convolut neural network
cnn imag brick wall built laboratori environ test abil detect crack imag brick-an
d-mortar structur laboratori real-world imag taken internet also compar perform c
nn varieti simpler classifi oper handcraft featur find cnn perform better domain
adapt laboratori real-world imag simpl model howev also find perform significantl
i better perform revers domain adapt task simpl classifi train real-world imag te
st laboratori imag work demonstr abil detect crack imag masonri use varieti machi
n learn method provid guidanc improv reliabl model perform domain adapt crack det
ect masonri
toward energi effici code gener mobil phone use smartphon becom part everyday lif
e last year devic help us mani area life sport job weather etc sometim also annoy
batteri life time import find solut reduc energi consumpt smartphon one possibl m
ethod 'comput offload part process execut remot devic e.g cloud lot exampl alread
i shown comput offload reduc energi usag mobil devic howev amount energi save may
differ decis make offload process control sever techniqu decis make theori base s
chedul theori paper go introduc new system call ecgm energi effici code gener mob
il phone ecgm decid automat compil time task run smartphon task offload benefit s
ystem demonstr measur base energy-effici schedul techniqu
sub-polyhedr schedul use unit- two-variable-per-inequ polyhedra polyhedr compil s
uccess design implement complex loop nest optim parallel compil algorithm complex
scalabl limit remain one import weak address use sub-polyhedr under-aproxim syste
m constraint result affin schedul problem propos sub-polyhedr schedul techniqu us
e unit- two-variable-per-inequ u tvpi polyhedra techniqu reli simpl polynomi time
algorithm under-approxim gener polyhedron u tvpi polyhedra modifi state-of-the-ar
t pluto compil use schedul techniqu show major polybench . kernel under-approxim
yield polyhedra non-empti solv under-approxim system lead asymptot gain complex s
how practic signific improv compar tradit lp solver also verifi code gener sub-po
lyhedr parallel prototyp match perform pluto-optim code under-approxim preserv fe
asibl copyright
extract multipl viewpoint model relat databas much time process mine project spen
t find understand data sourc extract event data need result fraction time spent a
ctual appli techniqu discov control predict busi process moreov current process m
ine techniqu assum singl case notion howev real-lif process often differ case not

```
ion intertwin exampl event order handl process may refer custom order order line
deliveri payment therefor propos use multipl viewpoint mvp model relat event obje
ct relat activ class requir event data much closer exist relat databas mvp model
provid holist view process also allow extract classic event log use differ viewpo
int way exist process mine techniqu use viewpoint without need new data extract t
ransform provid toolchain allow discoveri mvp model annot perform frequenc inform
relat databas moreov demonstr classic process mine techniqu appli select viewpoin
t
program result checker lexic analysi gnu c compil theori program result check est
ablish well-suit method construct formal correct compil frontend never prove prac
tic real-lif compil proof necessari establish result check method choic implement
compil correctli show lexic analysi gnu c compil formal specifi check within theo
rem prover isabelle/hol util program check therebi demonstr formal specif verif t
echniqu abl handl real-lif compil
advanc visual object track algorithm base correl filter 基于相关滤波的视觉目标跟踪算
法新进展 excel comprehens perform correl filter-bas track algorithm becom hotspot
theoret research practic applic field visual object track despit mani studi still
lack systemat analys exist correl filter-bas track algorithm level track framewor
k therefor start basic framework object track algorithm characterist correl filte
r-bas track algorithm deepli analyz basic problem work stage present paper basi m
ain technolog progress correl filter-bas track algorithm characterist correspond
algorithm recent ten year summar typic correl filter-bas track algorithm evalu an
alyz final outstand issu urgent solv futur research direct correl filter-bas trac
k algorithm discuss
studi variou databas model relat graph hybrid databas relat databas popular datab
as store variou type inform due ever-increas growth data becom hard maintain proc
ess databas graph model becom popular sinc store handl big data effici compar rel
at databas relat databas graph databas advantag disadvantag overcom limit combin
make hybrid model paper discuss relat databas graph databas advantag applic also
talk hybrid model
better report studi artifici intellig consort-ai beyond increas number studi arti
fici intellig ai publish dental oral scienc report also aspect studi suffer rang
limit standard toward report like recent publish consolid standard report trial c
onsort -ai extens help improv studi emerg field journal dental research jdr encou
rag author review reader adher standard notabl though wide rang aspect beyond rep
ort locat along variou step ai lifecycl consid conceiv conduct report evalu studi
ai dentistri
```

In [ ]:

```
#5. Check most frequent words - candidates to add to the stopword list
listofall = [ item for elem in stemmed_tokens_list for item in elem]

freq = FreqDist(listofall)
wnum=freq.B()
print("\nMost common words (total %d)"%wnum)
print(freq.most_common(30))
```

```
Most common words (total 10259)
[('use', 1793), ('data', 1238), ('system', 1208), ('propos', 1082), ('model', 93
7), ('method', 880), ('comput', 868), ('robot', 806), ('imag', 792), ('perform',
774), ('base', 728), ('algorithm', 719), ('databas', 701), ('result', 685), ('sec
ur', 665), ('paper', 635), ('approach', 621), ('compil', 602), ('applic', 594),
('design', 569), ('gener', 548), ('learn', 543), ('develop', 535), ('detect', 51
3), ('process', 512), ('.', 512), ('inform', 507), ('network', 505), ('present',
504), ('implement', 481)]
```

We now remove stopwords like use, data, system as they appear after the stemming. We remove the top 30 common words, which contribute little meanings to the research title's focus.

In [ ]:

```python
# Assuming 'freq' is your FreqDist object and 'stemmed_tokens_list' is your list
most_common_words = [word for word, freq in freq.most_common(30)]

# Convert the list to a set for faster membership testing
common_words_set = set(most_common_words)

# Now filter out these common words from the stemmed tokens
filtered_tokens_list = [[token for token in tokens if token not in common_words_

#5. Check most frequent words - candidates to add to the stopword list
listofall = [ item for elem in filtered_tokens_list for item in elem]

freq_filtered = FreqDist(listofall)
wnum=freq_filtered.B()
print("\nMost common words after filtering (total %d)"%wnum)
print(freq_filtered.most_common(30))
```

```
Most common words after filtering (total 10229)
[('differ', 470), ('improv', 445), ('relat', 445), ('provid', 441), ('show', 43
7), ('optim', 437), ('techniqu', 430), ('time', 429), ('studi', 417), ('program',
417), ('evalu', 386), ('also', 385), ('effici', 380), ('work', 375), ('problem',
366), ('analysi', 365), ('object', 361), ('scheme', 358), ('research', 349), ('ne
w', 348), ('featur', 347), ('control', 337), ('key', 336), ('structur', 333), ('c
ompar', 332), ('requir', 331), ('vision', 327), ('two', 324), ('task', 320), ('fr
amework', 316)]
```

2. K-means clustering and topic detection

(b) Cluster the preprocessed data with K-means trying $K = 3, \ldots, 10$. Evaluate the clustering quality with the Davies-Bouldin index and select the best K. Then evaluate the most frequent unigrams and most frequent bigrams in each cluster. (It is possible that the lists still contain some uninformative stopwords that you need to exclude.) Try to conclude what is the topic of each cluster. This is the baseline solution, so don't worry, if all the topics are not yet clear.

Report here the results of the K-means approach. What was the best clustering (K and Davies-Bouldin index), the most frequent unigrams and bigrams in clusters (e.g., in a table), and your conclusion on the topics.

Now, we proceed to add unigram, bigram and both grams tf-idf models

In [ ]:

```python
#6. Present as tf-idf
cleaned_documents = [' '.join(sentence_tokens) for sentence_tokens in filtered_t

# copy of clean_documents for part (c) SVD
SVD_cleaned_documents = deepcopy(cleaned_documents)

print('The preprocessed clean documents:')
for document in cleaned_documents[:10]:
        print(document)
```

The preprocessed clean documents:
anomali wide area imageri geniş alan görüntülerind anomali tespiti studi anomali wide area imageri collect aircraft set anomali identifi anyth normal cours action purpos two differ set experi carri set anomali convolut neural tri next past pre-process given anomali compar estim true
person re-identif deep kronecker-product match group-shuffl random walk person re-identif re-id aim robustli measur visual affin person wide intellig surveil associ person across multipl camera treat retriev problem given probe person affin probe galleri pg affin rank retriev galleri exist two main challeng effect solv problem person usual show signific variat differ person pose view angl spatial layout correspond person therefor vital tackl problem state-of-the-art either ignor spatial variat util extra pose handl challeng exist person re-id rank galleri consid pg affin ignor affin galleri gg affin affin could provid import clue accur galleri rank util post-process stage current articl unifi end-to-end deep framework tackl two challeng handl viewpoint pose variat compar person novel kroneck product match oper match warp featur map differ person compar warp featur map accur pg affin fulli util avail pg gg affin accur rank galleri person novel group-shuffl random walk oper kroneck product match group-shuffl random walk oper end-to-end trainabl shown improv visual featur integr deep framework outperform state-of-the-art market- cuhk dukemtmc dataset demonstr effect abil code avail http //github.com/yantaoshen/kpm_rw_person_reid
crack masonri cnn signific bodi research crack vision concret asphalt less attent given masonri train convolut neural cnn brick wall built laboratori environ test abil crack brick-and-mortar structur laboratori real-world taken internet also compar cnn varieti simpler classifi oper handcraft featur find cnn better domain adapt laboratori real-world simpl howev also find significantli better revers domain adapt task simpl classifi train real-world test laboratori work demonstr abil crack masonri varieti machin provid guidanc improv reliabl domain adapt crack masonri
toward energi effici code mobil phone smartphon becom part everyday life last year devic help us mani area life sport job weather etc sometim also annoy batteri life time import find solut reduc energi consumpt smartphon one possibl 'comput offload part execut remot devic e.g cloud lot exampl alreadi shown offload reduc energi usag mobil devic howev amount energi save may differ decis make offload control sever techniqu decis make theori schedul theori go introduc new call ecgm energi effici code mobil phone ecgm decid automat time task run smartphon task offload benefit demonstr measur energy-effici schedul techniqu
sub-polyhedr schedul unit- two-variable-per-inequ polyhedra polyhedr success complex loop nest optim parallel complex scalabl limit remain one import weak address sub-polyhedr under-aproxim constraint affin schedul problem sub-polyhedr schedul techniqu unit- two-variable-per-inequ u tvpi polyhedra techniqu reli simpl polynomi time under-approxim polyhedron u tvpi polyhedra modifi state-of-the-art pluto schedul techniqu show major polybench kernel under-approxim yield polyhedra non-empti solv under-approxim lead asymptot gain complex show practic signific improv compar tradit lp solver also verifi code sub-polyhedr parallel prototyp match pluto-optim code under-approxim preserv feasibl copyright
extract multipl viewpoint relat much time mine project spent find understand sourc extract event need fraction time spent actual appli techniqu discov control predict busi moreov current mine techniqu assum singl case notion howev real-lif often differ case notion intertwin exampl event order handl may refer custom order order line deliveri payment therefor multipl viewpoint mvp relat event object relat activ class requir event much closer exist relat mvp provid holist view also allow extract classic event log differ viewpoint way exist mine techniqu viewpoint without need new extract transform provid toolchain allow discoveri mvp annot frequenc relat moreov demonstr classic mine techniqu appli select viewpoint
program checker lexic analysi gnu c theori program check establish well-suit construct formal correct frontend never prove practic real-lif proof necessari establish check choic correctli show lexic analysi gnu c formal specifi check within theorem prover isabelle/hol util program check therebi demonstr formal specif verif techniqu abl handl real-lif

```
advanc visual object track correl filter 基于相关滤波的视觉目标跟踪算法新进展 excel
comprehens correl filter-bas track becom hotspot theoret research practic field v
isual object track despit mani studi still lack systemat analys exist correl filt
er-bas track level track framework therefor start basic framework object track ch
aracterist correl filter-bas track deepli analyz basic problem work stage basi ma
in technolog progress correl filter-bas track characterist correspond recent ten
year summar typic correl filter-bas track evalu analyz final outstand issu urgent
solv futur research direct correl filter-bas track discuss
studi variou relat graph hybrid relat popular store variou type due ever-increas
growth becom hard maintain graph becom popular sinc store handl big effici compar
relat relat graph advantag disadvantag overcom limit combin make hybrid discuss r
elat graph advantag also talk hybrid
better report studi artifici intellig consort-ai beyond increas number studi arti
fici intellig ai publish dental oral scienc report also aspect studi suffer rang
limit standard toward report like recent publish consolid standard report trial c
onsort -ai extens help improv studi emerg field journal dental research jdr encou
rag author review reader adher standard notabl though wide rang aspect beyond rep
ort locat along variou step ai lifecycl consid conceiv conduct report evalu studi
ai dentistri
```

Unigram tf-idf vectorizer

In [ ]:

```
# Ignoring terms that appear in less than 5% of the documents or in more than 25

unigram_tfidf_vectorizer = TfidfVectorizer(
    min_df=0.05,
    max_df=0.25,
    smooth_idf=False,
    norm='l2',            # Ensures all our feature vectors have a euclidian nor
    ngram_range=(1,1)     # Extract only unigrams
)

#only tf part:
#tfidf_vectorizer = TfidfVectorizer(use_idf=False)

unigram_tfidf_vectorizer.fit(cleaned_documents)
unigram_tf_idf_vectors = unigram_tfidf_vectorizer.transform(cleaned_documents)

print("\nThe shape of the tf-idf vectors (number of documents, number of feature
print(unigram_tf_idf_vectors.shape)


print("\nThe tf-idf values of the first document (unigrams)\n")
feature_names = unigram_tfidf_vectorizer.get_feature_names_out()
feature_index = unigram_tf_idf_vectors[0,:].nonzero()[1]
tfidf_scores = zip(feature_index, [unigram_tf_idf_vectors[0, x] for x in feature
for w, s in [(feature_names[i], s) for (i, s) in tfidf_scores]:
    print(w, s)
```

```
The shape of the tf-idf vectors (number of documents, number of features) for uni
gram model is
(1143, 349)

The tf-idf values of the first document (unigrams)

wide 0.4270667202788506
two 0.14821467742261452
studi 0.14557646252298534
set 0.4974544721165923
purpos 0.2292533250306582
neural 0.1984329905770054
identifi 0.20727323621885707
given 0.22354864669609772
experi 0.16858331667935514
estim 0.21095129460311848
convolut 0.2176249810162788
compar 0.14710144650586116
collect 0.22125896429309003
area 0.3889064878787454
```

Bigram tf-idf vectorizer

In [ ]:

```python
# Ignoring terms that appear in less than 1.5% of the documents or in more than
# This min df relaxation compared to unigram mode helps more bigrams to be consi

bigram_tfidf_vectorizer = TfidfVectorizer(
    min_df=0.015,
    max_df=0.25,
    smooth_idf=False,
    norm='l2',             # Ensures all our feature vectors have a euclidian nor
    ngram_range=(2,2)      # Extract only bigrams
)


bigram_tfidf_vectorizer.fit(cleaned_documents)
bigram_tf_idf_vectors = bigram_tfidf_vectorizer.transform(cleaned_documents)

print("\nThe shape of the tf-idf vectors (number of documents, number of feature
print(bigram_tf_idf_vectors.shape)


print("\nThe tf-idf values of the first document (bigrams)\n")
feature_names = bigram_tfidf_vectorizer.get_feature_names_out()
feature_index = bigram_tf_idf_vectors[0,:].nonzero()[1]
tfidf_scores = zip(feature_index, [bigram_tf_idf_vectors[0, x] for x in feature_
for w, s in [(feature_names[i], s) for (i, s) in tfidf_scores]:
    print(w, s)
```

```
The shape of the tf-idf vectors (number of documents, number of features) for big
ram model is
(1143, 50)

The tf-idf values of the first document (bigrams)

two differ 0.7839204046175395
convolut neural 0.6208613365513053
```

Both grams tf-idf vectorizer

In [ ]:

```
# Ignoring terms that appear in less than 0.5% of the documents or in more than
# The very small min_df accounts for the explosion of terms caused by bigrams

both_tfidf_vectorizer = TfidfVectorizer(
    min_df=0.005,
    max_df=0.25,
    smooth_idf=False,
    norm='l2',            # Ensures all our feature vectors have a euclidian nor
    ngram_range=(1,2)     # Extract only both unigrams and bigrams
)


both_tfidf_vectorizer.fit(cleaned_documents)
both_tf_idf_vectors = both_tfidf_vectorizer.transform(cleaned_documents)

print("\nThe shape of the tf-idf vectors (number of documents, number of feature
print(both_tf_idf_vectors.shape)


print("\nThe tf-idf values of the first document (both grams)\n")
feature_names = both_tfidf_vectorizer.get_feature_names_out()
feature_index = both_tf_idf_vectors[0,:].nonzero()[1]
tfidf_scores = zip(feature_index, [both_tf_idf_vectors[0, x] for x in feature_in
for w, s in [(feature_names[i], s) for (i, s) in tfidf_scores]:
    print(w, s)
```

```
The shape of the tf-idf vectors (number of documents, number of features) for bot
h grams model is
(1143, 2619)

The tf-idf values of the first document (both grams)

wide 0.1655509884341766
two differ 0.11437782918690984
two 0.057454924916052634
true 0.12705410179616836
tri 0.12248788556946724
studi 0.056432229717395616
set 0.19283656545780464
purpos 0.08886928612902774
process 0.11751825835945813
pre process 0.14698109324214143
pre 0.10688834399803271
past 0.11341776593920974
normal 0.10121662633184234
next 0.11987204952994096
neural 0.07692188636595902
imageri 0.24782734524724012
identifi 0.08034877807752544
given 0.08665788661663103
experi 0.06535075992705697
estim 0.08177456513167825
cours 0.10912987584470467
convolut neural 0.09058671197041691
convolut 0.08436159739089297
compar 0.05702338466748522
collect 0.08577029887677916
carri 0.09619373005334125
area 0.15075830173503266
anomali 0.772781964651437
aircraft 0.14335573382378414
action 0.11249536208095705
```

Clustering with K-means for the three n-grams model

In [ ]:
```python
# Clustering the documents based on unigram tf-idf vectorizer

min_score = 1e6
best_k = 0
for i in range(3,11):
    kmeans = KMeans(n_clusters=i, n_init=20)
    kmeans.fit(unigram_tf_idf_vectors)
    labels = kmeans.labels_
    db_index = davies_bouldin_score(unigram_tf_idf_vectors.toarray(), labels)
    if db_index < min_score:
        min_score = db_index
        best_k = i

print("Optimal number of clusters of the unigram model:", best_k)
print("Davies-Bouldin Index of the unigram model:", min_score, "\n")
kmeans = KMeans(n_clusters=best_k, n_init=10)
kmeans.fit(unigram_tf_idf_vectors)
labels = kmeans.labels_
db_index = davies_bouldin_score(unigram_tf_idf_vectors.toarray(), labels)
```

```
clusters = {i: [] for i in range(best_k)}

for point, label in zip(filtered_tokens_list, labels):
    clusters[label].append(point)

for i in range(best_k):
    listofcluster = [ item for elem in clusters[i] for item in elem]
    cluster_freq = FreqDist(listofcluster)
    print("Cluster ", i, ":", cluster_freq.most_common(15))
```

```
Optimal number of clusters of the unigram model: 10
Davies-Bouldin Index of the unigram model: 4.744800668411147

Cluster  0 : [('encrypt', 292), ('scheme', 271), ('key', 194), ('attack', 163),
('cryptographi', 156), ('protocol', 148), ('authent', 121), ('cloud', 108), ('pro
vid', 97), ('iot', 93), ('effici', 86), ('cryptograph', 75), ('new', 72), ('devi
c', 72), ('techniqu', 71)]
Cluster  1 : [('dataset', 180), ('train', 153), ('deep', 130), ('neural', 122),
('featur', 118), ('accuraci', 112), ('segment', 102), ('differ', 97), ('improv',
95), ('predict', 94), ('convolut', 89), ('achiev', 85), ('classif', 77), ('eval
u', 75), ('show', 71)]
Cluster  2 : [('quantum', 246), ('cryptographi', 52), ('key', 45), ('protocol', 4
0), ('attack', 37), ('oper', 35), ('post-quantum', 31), ('state', 31), ('commun',
31), ('gate', 29), ('scheme', 28), ('time', 27), ('circuit', 27), ('architectur',
26), ('also', 25)]
Cluster  3 : [('vision', 186), ('video', 151), ('technolog', 115), ('research', 8
8), ('studi', 81), ('review', 75), ('techniqu', 72), ('recognit', 61), ('work', 6
0), ('analysi', 59), ('visual', 59), ('provid', 56), ('monitor', 56), ('structu
r', 55), ('deep', 53)]
Cluster  4 : [('control', 218), ('measur', 154), ('estim', 75), ('sensor', 74),
('environ', 72), ('studi', 67), ('differ', 61), ('task', 58), ('simul', 55), ('ef
fect', 52), ('optim', 51), ('motion', 51), ('dynam', 50), ('improv', 46), ('spee
d', 45)]
Cluster  5 : [('provid', 118), ('studi', 112), ('task', 106), ('integr', 100),
('relat', 98), ('differ', 95), ('test', 94), ('construct', 94), ('research', 93),
('optim', 93), ('work', 91), ('oper', 91), ('time', 90), ('also', 89), ('evalu',
89)]
Cluster  6 : [('queri', 237), ('relat', 234), ('sql', 59), ('store', 48), ('grap
h', 45), ('effici', 44), ('user', 44), ('languag', 39), ('time', 37), ('ontolog',
37), ('manag', 36), ('schema', 35), ('differ', 34), ('optim', 33), ('structur', 3
2)]
Cluster  7 : [('program', 204), ('code', 172), ('optim', 142), ('graph', 139),
('memori', 128), ('parallel', 122), ('transform', 92), ('time', 77), ('techniqu',
76), ('regist', 73), ('problem', 64), ('show', 61), ('new', 60), ('alloc', 57),
('execut', 56)]
Cluster  8 : [('object', 214), ('track', 129), ('featur', 45), ('evalu', 36), ('a
ccuraci', 35), ('improv', 33), ('challeng', 32), ('show', 31), ('visual', 30),
('train', 30), ('differ', 27), ('deep', 27), ('filter', 26), ('research', 26),
('howev', 26)]
Cluster  9 : [('languag', 183), ('program', 105), ('semant', 53), ('type', 44),
('proof', 35), ('transform', 33), ('framework', 33), ('construct', 30), ('theor
i', 29), ('formal', 26), ("'s", 25), ('teach', 24), ('softwar', 23), ('verifi', 2
3), ('code', 22)]
```

In [ ]:

```
# Clustering the documents based on bigram tf-idf vectorizer

min_score = 1e6
best_k = 0
for i in range(3,11):
    kmeans = KMeans(n_clusters=i, n_init=20)
```

```python
    kmeans.fit(bigram_tf_idf_vectors)
    labels = kmeans.labels_
    db_index = davies_bouldin_score(bigram_tf_idf_vectors.toarray(), labels)
    if db_index < min_score:
        min_score = db_index
        best_k = i

print("Optimal number of clusters of the bigram model:", best_k)
print("Davies-Bouldin Index of the bigram model:", min_score, "\n")

# Fit the KMeans model to find the best_k clusters
kmeans = KMeans(n_clusters=best_k, n_init=20)
kmeans.fit(bigram_tf_idf_vectors)
labels = kmeans.labels_

# Extract the top bigrams for each cluster center
feature_names = bigram_tfidf_vectorizer.get_feature_names_out()

for i in range(best_k):
    # Get indices of the top features for this cluster
    top_feature_indices = kmeans.cluster_centers_[i].argsort()[-10:][::-1]

    print(f"Cluster {i} and bigram TF-IDF score:", end=" ")
    for idx in top_feature_indices:
        print(f"({feature_names[idx]}: {kmeans.cluster_centers_[i][idx]:.4f})",
    print("\n")
```

```
Optimal number of clusters of the bigram model: 8
Davies-Bouldin Index of the bigram model: 1.2864844842709722

Cluster 0 and bigram TF-IDF score: (program languag: 0.0270), (real world: 0.025
4), (experiment show: 0.0249), (recent year: 0.0233), (execut time: 0.0222), (ope
n sourc: 0.0215), (time consum: 0.0185), (two differ: 0.0184), (well known: 0.018
1), (solv problem: 0.0181),

Cluster 1 and bigram TF-IDF score: (convolut neural: 0.6531), (neural cnn: 0.257
1), (artifici intellig: 0.0784), (experiment show: 0.0687), (time consum: 0.033
7), (learning bas: 0.0271), (end to: 0.0268), (to end: 0.0268), (open sourc: 0.02
61), (solv problem: 0.0259),

Cluster 2 and bigram TF-IDF score: (vision bas: 0.7797), (three dimension: 0.066
0), (improv accuraci: 0.0590), (real world: 0.0561), (futur research: 0.0440), (w
ide rang: 0.0392), (low cost: 0.0385), (real tim: 0.0369), (learning bas: 0.035
3), (convolut neural: 0.0321),

Cluster 3 and bigram TF-IDF score: (case studi: 0.9006), (et al: 0.0453), (experi
conduct: 0.0227), (three dimension: 0.0227), (two differ: 0.0222), (time consum:
0.0217), (real world: 0.0210), (secret key: 0.0190), (deep learning: 0.0186), (le
arning bas: 0.0164),

Cluster 4 and bigram TF-IDF score: (high level: 0.8721), (program languag: 0.128
5), (convolut neural: 0.0326), (experi conduct: 0.0214), (high perform: 0.0212),
(large scal: 0.0209), (case studi: 0.0196), (recent advanc: 0.0185), (wide rang:
0.0173), (internet thing: 0.0160),

Cluster 5 and bigram TF-IDF score: (state of: 0.4422), (of the: 0.4331), (the ar
t: 0.4312), (learning bas: 0.0591), (real world: 0.0527), (deep learning: 0.039
5), (experiment show: 0.0386), (artifici intellig: 0.0348), (convolut neural: 0.0
343), (futur research: 0.0326),

Cluster 6 and bigram TF-IDF score: (ellipt curv: 0.6534), (curv cryptographi: 0.4
573), (public key: 0.0833), (low cost: 0.0699), (internet thing: 0.0610), (secret
key: 0.0515), (et al: 0.0513), (thing iot: 0.0484), (key cryptographi: 0.0463),
(key encrypt: 0.0359),

Cluster 7 and bigram TF-IDF score: (real tim: 0.8296), (state of: 0.0456), (solv
problem: 0.0400), (real time: 0.0386), (experiment show: 0.0381), (the art: 0.032
5), (of the: 0.0321), (power consumpt: 0.0300), (execut time: 0.0287), (convolut
neural: 0.0192),
```

In [ ]:

```python
# Clustering the documents based on bigram tf-idf vectorizer

min_score = 1e6
best_k = 0
for i in range(3,11):
    kmeans = KMeans(n_clusters=i, n_init=20)
    kmeans.fit(both_tf_idf_vectors)
    labels = kmeans.labels_
    db_index = davies_bouldin_score(both_tf_idf_vectors.toarray(), labels)
    if db_index < min_score:
        min_score = db_index
        best_k = i

print("Optimal number of clusters of the both grams model:", best_k)
print("Davies-Bouldin Index of the both grams model:", min_score, "\n")
```

```
# Assuming 'both_tf_idf_vectors' is your TF-IDF matrix and 'both_tfidf_vectorize

# Fit the KMeans model to find the best_k clusters
kmeans = KMeans(n_clusters=best_k, n_init=20)
kmeans.fit(both_tf_idf_vectors)
labels = kmeans.labels_

# Extract the top bigrams for each cluster center
feature_names = both_tfidf_vectorizer.get_feature_names_out()

for i in range(best_k):
    # Get indices of the top features for this cluster
    top_feature_indices = kmeans.cluster_centers_[i].argsort()[-10:][::-1]

    print(f"Cluster {i} and TF-IDF score:", end=" ")
    for idx in top_feature_indices:
        print(f"({feature_names[idx]}: {kmeans.cluster_centers_[i][idx]:.4f})",
    print("\n")
```

```
Optimal number of clusters of the both grams model: 5
Davies-Bouldin Index of the both grams model: 6.968635009292602

Cluster 0 and TF-IDF score: (quantum: 0.4007), (gate: 0.0624), (key: 0.0573), (cr
yptographi: 0.0559), (qubit: 0.0554), (protocol: 0.0540), (circuit: 0.0473), (pos
t quantum: 0.0465), (attack: 0.0428), (post: 0.0413),

Cluster 1 and TF-IDF score: (encrypt: 0.0940), (scheme: 0.0708), (cryptographi:
0.0548), (key: 0.0536), (attack: 0.0522), (iot: 0.0482), (protocol: 0.0476), (clo
ud: 0.0407), (authent: 0.0407), (cryptograph: 0.0354),

Cluster 2 and TF-IDF score: (relat: 0.0467), (program: 0.0453), (queri: 0.0451),
(languag: 0.0449), (graph: 0.0366), (code: 0.0340), (optim: 0.0288), (transform:
0.0243), (sql: 0.0242), (memori: 0.0239),

Cluster 3 and TF-IDF score: (control: 0.0575), (measur: 0.0395), (soft: 0.0361),
(sensor: 0.0303), (estim: 0.0287), (environ: 0.0275), (task: 0.0271), (simul: 0.0
251), (human: 0.0240), (optim: 0.0221),

Cluster 4 and TF-IDF score: (vision: 0.0407), (object: 0.0403), (deep: 0.0361),
(dataset: 0.0307), (video: 0.0307), (featur: 0.0298), (track: 0.0288), (accuraci:
0.0273), (train: 0.0268), (segment: 0.0261),
```

Therefore, the baseline solution seems to have 5 topics:

- Topic 1: Database, SQL and queries
- Topic 2: General programming and operating system/hardware
- Topic 3: Computer vision and robotics
- Topic 4: Security and cryptography
- Topic 5: Quantum studies and application in security

3. Additional experiments

(c) Try to improve your results! Here you can freely try any methods covered in the course. You can improve the preprocessing (e.g., lemmatization), clustering (e.g., try dimension reduction or another clustering method) or the evaluation of the most important terms (e.g., utilize the title, perform SVD per cluster and look at the leading singular vector or analyze only the centroid or most central documents). Conclude the main (3–10) topics of the document collection based on your experiments!

Report here your experiments in the (c) part. Describe briefly what you tried and the results (the most important terms and concluded topics). Evaluate also if your experiment was successful, i.e., if it produced better results than the baseline. It is suggested to divide this section into subsections, if you tried many approaches.

Experiment with only the title (unigram tf-idf vectorizer)

In [ ]:
```
titledata = scopusdata['TITLE'].to_list()

title_tokens = [word_tokenize(document) for document in corpus]

title_lc_tokens_list = []

for token_document in title_tokens:
    title_lc_tokens_list.append([token.lower() for token in token_document])


# original number of tokens
uniques = np.unique([token for token_document in title_lc_tokens_list for token



# Steps 2 and 3: remove stop words and punctuation

title_filtered_sentence = []
for i in title_lc_tokens_list:
    title_filtered_sentence.append([token for token in i if token not in stop_wc

# Numbers are also removed
title_filtered_sentence = [ ' '.join(i) for i in title_filtered_sentence ]
title_filtered_sentence = [ re.sub(r'\d+', '', sentence) for sentence in title_f

# Step 4 Stemming
title_stemmed_tokens_list = []

for i in title_filtered_sentence:
        title_stemmed_tokens_list.append([porter.stem(j) for j in i.split()])

title_filtered_tokens_list = [[token for token in tokens if token not in common_

#6. Present as tf-idf
title_cleaned_documents = [' '.join(sentence_tokens) for sentence_tokens in titl


unigram_tfidf_vectorizer = TfidfVectorizer(
    min_df=0.05,
    max_df=0.25,
    smooth_idf=False,
    norm='l2',           # Ensures all our feature vectors have a euclidian nor
    ngram_range=(1,1)    # Extract only unigrams
)
unigram_tfidf_vectorizer.fit(title_cleaned_documents)
```

```
title_unigram_tf_idf_vectors = unigram_tfidf_vectorizer.transform(title_cleaned_

feature_names = unigram_tfidf_vectorizer.get_feature_names_out()
feature_index = title_unigram_tf_idf_vectors[0,:].nonzero()[1]
tfidf_scores = zip(feature_index, [title_unigram_tf_idf_vectors[0, x] for x in f
```

In [ ]:

```
# Clustering the documents based on unigram tf-idf vectorizer

min_score = 1e6
best_k = 0
for i in range(3,11):
    kmeans = KMeans(n_clusters=i, n_init=20)
    kmeans.fit(unigram_tf_idf_vectors)
    labels = kmeans.labels_
    db_index = davies_bouldin_score(title_unigram_tf_idf_vectors.toarray(), labe
    if db_index < min_score:
        min_score = db_index
        best_k = i

print("Optimal number of clusters of the unigram model:", best_k)
print("Davies-Bouldin Index of the unigram model:", min_score, "\n")
kmeans = KMeans(n_clusters=best_k, n_init=10)
kmeans.fit(title_unigram_tf_idf_vectors)
labels = kmeans.labels_
db_index = davies_bouldin_score(title_unigram_tf_idf_vectors.toarray(), labels)

clusters = {i: [] for i in range(best_k)}

for point, label in zip(filtered_tokens_list, labels):
    clusters[label].append(point)

for i in range(best_k):
    listofcluster = [ item for elem in clusters[i] for item in elem]
    cluster_freq = FreqDist(listofcluster)
    print("Cluster ", i, ":", cluster_freq.most_common(15))
```

```
Optimal number of clusters of the unigram model: 5
Davies-Bouldin Index of the unigram model: 4.841051642953007

Cluster  0 : [('vision', 286), ('object', 276), ('dataset', 220), ('deep', 219),
('featur', 205), ('train', 204), ('video', 187), ('accuraci', 186), ('visual', 16
9), ('track', 166), ('differ', 165), ('neural', 160), ('research', 159), ('stud
i', 155), ('improv', 154)]
Cluster  1 : [('program', 305), ('relat', 301), ('languag', 260), ('queri', 254),
('graph', 183), ('code', 179), ('transform', 129), ('type', 118), ('semant', 11
7), ('show', 104), ('optim', 102), ('analysi', 100), ('tool', 99), ('time', 97),
('parallel', 96)]
Cluster  2 : [('quantum', 254), ('cryptographi', 54), ('key', 45), ('protocol', 4
0), ('oper', 39), ('attack', 39), ('scheme', 34), ('post-quantum', 33), ('state',
33), ('gate', 32), ('commun', 32), ('circuit', 32), ('program', 31), ('time', 3
1), ('also', 30)]
Cluster  3 : [('encrypt', 292), ('scheme', 264), ('key', 194), ('attack', 161),
('cryptographi', 156), ('protocol', 148), ('authent', 122), ('cloud', 108), ('pro
vid', 100), ('iot', 93), ('effici', 83), ('cryptograph', 75), ('techniqu', 73),
('new', 71), ('devic', 70)]
Cluster  4 : [('control', 249), ('optim', 239), ('studi', 174), ('improv', 162),
('task', 161), ('differ', 154), ('time', 145), ('show', 143), ('provid', 133),
('also', 131), ('environ', 129), ('problem', 128), ('work', 128), ('oper', 127),
('increas', 126)]
```

Therefore, the solution with only the title seems to have 5 topics (The order of the topics may randomly change each time the algorithm is run):

- Topic 1: General programming and operating system/hardware
- Topic 2: Quantum studies and application in security
- Topic 3: Database, SQL and queries
- Topic 4: Computer vision and robotics
- Topic 5: Security and cryptography

Experiment with lemmatization (unigram tf-idf vectorizer)

In [ ]:
```
nltk.download('averaged_perceptron_tagger')
nltk.download('wordnet')

from nltk.stem import WordNetLemmatizer
from nltk.corpus import wordnet

lemmatizer = WordNetLemmatizer()

def get_wordnet_pos(word):
    """Map POS tag to first character lemmatize() accepts."""
    tag = nltk.pos_tag([word])[0][1][0].upper()
    tag_dict = {"J": wordnet.ADJ,
                "N": wordnet.NOUN,
                "V": wordnet.VERB,
                "R": wordnet.ADV}

    return tag_dict.get(tag, wordnet.NOUN)

lemmatized_tokens_list = []

lemmatized_tokens_list = [[lemmatizer.lemmatize(word, get_wordnet_pos(word)) for
cleaned = [' '.join(sentence_tokens) for sentence_tokens in lemmatized_tokens_li

print('The preprocessed clean documents:')
for document in cleaned[:10]:
        print(document)
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

The preprocessed clean documents:
anomali wide area imageri geniş alan görüntülerind anomali tespiti studi anomali wide area imageri collect aircraft set anomali identifi anyth normal cours action purpos two differ set experi carri set anomali convolut neural tri next past pre-process give anomali compar estim true
person re-identif deep kronecker-product match group-shuffl random walk person re-identif re-id aim robustli measur visual affin person wide intellig surveil associ person across multipl camera treat retriev problem give probe person affin probe galleri pg affin rank retriev galleri exist two main challeng effect solv problem person usual show signific variat differ person pose view angl spatial layout correspond person therefor vital tackl problem state-of-the-art either ignor spatial variat util extra pose handl challeng exist person re-id rank galleri consid pg affin ignor affin galleri gg affin affin could provid import clue accur galleri rank util post-process stage current articl unifi end-to-end deep framework tackl two challeng handl viewpoint pose variat compar person novel kroneck product match oper match warp featur map differ person compar warp featur map accur pg affin fulli util avail pg gg affin accur rank galleri person novel group-shuffl random walk oper kroneck product match group-shuffl random walk oper end-to-end trainabl show improv visual featur integr deep framework outperform state-of-the-art market- cuhk dukemtmc dataset demonstr effect abil code avail http //github.com/yantaoshen/kpm_rw_person_reid
crack masonri cnn signific bodi research crack vision concret asphalt less attent give masonri train convolut neural cnn brick wall built laboratori environ test abil crack brick-and-mortar structur laboratori real-world take internet also compar cnn varieti simpler classifi oper handcraft featur find cnn well domain adapt laboratori real-world simpl howev also find significantli well revers domain adapt task simpl classifi train real-world test laboratori work demonstr abil crack masonri varieti machin provid guidanc improv reliabl domain adapt crack masonri
toward energi effici code mobil phone smartphon becom part everyday life last year devic help u mani area life sport job weather etc sometim also annoy batteri life time import find solut reduc energi consumpt smartphon one possibl 'comput offload part execut remot devic e.g cloud lot exampl alreadi show offload reduc energi usag mobil devic howev amount energi save may differ decis make offload control sever techniqu decis make theori schedul theori go introduc new call ecgm energi effici code mobil phone ecgm decid automat time task run smartphon task offload benefit demonstr measur energy-effici schedul techniqu
sub-polyhedr schedul unit- two-variable-per-inequ polyhedron polyhedr success complex loop nest optim parallel complex scalabl limit remain one import weak address sub-polyhedr under-aproxim constraint affin schedul problem sub-polyhedr schedul techniqu unit- two-variable-per-inequ u tvpi polyhedron techniqu reli simpl polynomi time under-approxim polyhedron u tvpi polyhedron modifi state-of-the-art pluto schedul techniqu show major polybench kernel under-approxim yield polyhedron non-empti solv under-approxim lead asymptot gain complex show practic signific improv compar tradit lp solver also verifi code sub-polyhedr parallel prototyp match pluto-optim code under-approxim preserv feasibl copyright
extract multipl viewpoint relat much time mine project spent find understand sourc extract event need fraction time spent actual appli techniqu discov control predict busi moreov current mine techniqu assum singl case notion howev real-lif often differ case notion intertwin exampl event order handl may refer custom order order line deliveri payment therefor multipl viewpoint mvp relat event object relat activ class requir event much closer exist relat mvp provid holist view also allow extract classic event log differ viewpoint way exist mine techniqu viewpoint without need new extract transform provid toolchain allow discoveri mvp annot frequenc relat moreov demonstr classic mine techniqu appli select viewpoint
program checker lexic analysi gnu c theori program check establish well-suit construct formal correct frontend never prove practic real-lif proof necessari establish check choic correctli show lexic analysi gnu c formal specifi check within theorem prover isabelle/hol util program check therebi demonstr formal specif verif techniqu abl handl real-lif
advanc visual object track correl filter 基于相关滤波的视觉目标跟踪算法新进展 excel

comprehens correl filter-bas track becom hotspot theoret research practic field visual object track despit mani studi still lack systemat analys exist correl filter-bas track level track framework therefor start basic framework object track characterist correl filter-bas track deepli analyz basic problem work stage basi main technolog progress correl filter-bas track characterist correspond recent ten year summar typic correl filter-bas track evalu analyz final outstand issu urgent solv futur research direct correl filter-bas track discus
studi variou relat graph hybrid relat popular store variou type due ever-increas growth becom hard maintain graph becom popular sinc store handl big effici compar relat relat graph advantag disadvantag overcom limit combin make hybrid discus relat graph advantag also talk hybrid
well report studi artifici intellig consort-ai beyond increas number studi artifici intellig ai publish dental oral scienc report also aspect studi suffer rang limit standard toward report like recent publish consolid standard report trial consort -ai extens help improv studi emerg field journal dental research jdr encourag author review reader adher standard notabl though wide rang aspect beyond report locat along variou step ai lifecycl consid conceiv conduct report evalu studi ai dentistri

In [ ]:
```
unigram_tfidf_vectorizer = TfidfVectorizer(
    min_df=0.05,
    max_df=0.25,
    smooth_idf=False,
    norm='l2',            # Ensures all our feature vectors have a euclidian nor
    ngram_range=(1,1)     # Extract only unigrams
)
unigram_tfidf_vectorizer.fit(cleaned)
lemmatized_unigram_tf_idf_vectors = unigram_tfidf_vectorizer.transform(cleaned)

print("\nThe shape of the tf-idf vectors (number of documents, number of feature
print(lemmatized_unigram_tf_idf_vectors.shape)


print("\nThe tf-idf values of the first document (unigrams)\n")
feature_names = unigram_tfidf_vectorizer.get_feature_names_out()
feature_index = lemmatized_unigram_tf_idf_vectors[0,:].nonzero()[1]
tfidf_scores = zip(feature_index, [lemmatized_unigram_tf_idf_vectors[0, x] for x
for w, s in [(feature_names[i], s) for (i, s) in tfidf_scores]:
    print(w, s)
```

The shape of the tf-idf vectors (number of documents, number of features) for unigram model is
(1143, 348)

The tf-idf values of the first document (unigrams)

wide 0.42968762964661683
two 0.14912427121218227
studi 0.14646986558212088
set 0.5005073512666091
purpos 0.23066025317240396
neural 0.19965077426086789
identifi 0.2085452725089655
give 0.1956466039096827
experi 0.1696179128512244
estim 0.21224590314532887
convolut 0.21896054598615544
compar 0.14800420839493267
collect 0.22261683102604074
area 0.3912932078193654

In [ ]:

```
# Clustering the documents based on unigram tf-idf vectorizer

min_score = 1e6
best_k = 0
for i in range(3,11):
    kmeans = KMeans(n_clusters=i, n_init=20)
    kmeans.fit(lemmatized_unigram_tf_idf_vectors)
    labels = kmeans.labels_
    db_index = davies_bouldin_score(lemmatized_unigram_tf_idf_vectors.toarray(),
    if db_index < min_score:
        min_score = db_index
        best_k = i

print("Optimal number of clusters of the unigram model:", best_k)
print("Davies-Bouldin Index of the unigram model:", min_score, "\n")
kmeans = KMeans(n_clusters=best_k, n_init=10)
kmeans.fit(lemmatized_unigram_tf_idf_vectors)
labels = kmeans.labels_
db_index = davies_bouldin_score(lemmatized_unigram_tf_idf_vectors.toarray(), lab

clusters = {i: [] for i in range(best_k)}

for point, label in zip(filtered_tokens_list, labels):
    clusters[label].append(point)

for i in range(best_k):
    listofcluster = [ item for elem in clusters[i] for item in elem]
    cluster_freq = FreqDist(listofcluster)
    print("Cluster ", i, ":", cluster_freq.most_common(15))
```

```
Optimal number of clusters of the unigram model: 5
Davies-Bouldin Index of the unigram model: 4.709833356812028

Cluster  0 : [('vision', 319), ('object', 312), ('studi', 301), ('differ', 288),
('control', 268), ('improv', 261), ('train', 248), ('featur', 247), ('task', 24
6), ('research', 237), ('measur', 234), ('dataset', 231), ('evalu', 226), ('accur
aci', 226), ('deep', 223)]
Cluster  1 : [('quantum', 246), ('cryptographi', 52), ('key', 45), ('protocol', 4
0), ('attack', 37), ('oper', 35), ('post-quantum', 31), ('state', 31), ('commun',
31), ('gate', 29), ('scheme', 28), ('time', 27), ('circuit', 27), ('architectur',
26), ('also', 25)]
Cluster  2 : [('encrypt', 302), ('scheme', 277), ('key', 198), ('cryptographi', 1
89), ('attack', 167), ('protocol', 143), ('authent', 124), ('iot', 122), ('provi
d', 109), ('cloud', 109), ('effici', 100), ('devic', 96), ('cryptograph', 94),
('techniqu', 86), ('new', 79)]
Cluster  3 : [('program', 345), ('code', 212), ('languag', 212), ('optim', 194),
('graph', 146), ('memori', 145), ('transform', 134), ('parallel', 131), ('techniq
u', 106), ('time', 102), ('show', 97), ('problem', 96), ('new', 92), ('analysi',
90), ('regist', 90)]
Cluster  4 : [('relat', 305), ('queri', 261), ('sql', 82), ('manag', 78), ('stor
e', 65), ('languag', 64), ('user', 63), ('schema', 56), ('graph', 55), ('differ',
53), ('effici', 53), ('ontolog', 53), ('structur', 53), ('analysi', 51), ('time',
49)]
```

Therefore, the solution with lemmatization seems to have 5 topics (The order of the topics may randomly change each time the algorithm is run):

- Topic 1: General programming and operating system/hardware
- Topic 2: Quantum studies and application in security

- Topic 3: Database, SQL and queries
- Topic 4: Computer vision and robotics
- Topic 5: Security and cryptography

Experiment with SVD (unigram tf-idf vectorizer)

SVD (Singular Value Decomposition) is a dimensionality reduction technique used as LSA (Latent Semantic Analysis) in text clustering and topic modeling. SVC is applied after preprocessing but before clustering.

In [ ]:
```
# Assuming 'unigram_tf_idf_vectors' is your TF-IDF matrix from the TfidfVectoriz
# Ignoring terms that appear in less than 5% of the documents or in more than 25

unigram_tfidf_vectorizer = TfidfVectorizer(
    min_df=0.05,
    max_df=0.25,
    smooth_idf=False,
    norm='l2',              # Ensures all our feature vectors have a euclidian nor
    ngram_range=(1,1)       # Extract only unigrams
)

#only tf part:
#tfidf_vectorizer = TfidfVectorizer(use_idf=False)

unigram_tfidf_vectorizer.fit(SVD_cleaned_documents)
unigram_tf_idf_vectors = unigram_tfidf_vectorizer.transform(SVD_cleaned_document

print("\nThe shape of the tf-idf vectors (number of documents, number of feature
print(unigram_tf_idf_vectors.shape)


# Clustering the documents based on unigram tf-idf vectorizer

min_score = 1e6
best_k = 0
for i in range(3,11):
    kmeans = KMeans(n_clusters=i, n_init=20)
    kmeans.fit(unigram_tf_idf_vectors)
    labels = kmeans.labels_
    db_index = davies_bouldin_score(unigram_tf_idf_vectors.toarray(), labels)
    if db_index < min_score:
        min_score = db_index
        best_k = i

print("Optimal number of clusters of the unigram model:", best_k)
print("Davies-Bouldin Index of the unigram model:", min_score, "\n")
kmeans = KMeans(n_clusters=best_k, n_init=10)
kmeans.fit(unigram_tf_idf_vectors)
labels = kmeans.labels_
db_index = davies_bouldin_score(unigram_tf_idf_vectors.toarray(), labels)

clusters = {i: [] for i in range(best_k)}

for point, label in zip(filtered_tokens_list, labels):
    clusters[label].append(point)

for i in range(best_k):
```

```
    listofcluster = [ item for elem in clusters[i] for item in elem]
    cluster_freq = FreqDist(listofcluster)
    print("Cluster ", i, ":", cluster_freq.most_common(15))
```

```
The shape of the tf-idf vectors (number of documents, number of features) for uni
gram model is
(1143, 349)
Optimal number of clusters of the unigram model: 5
Davies-Bouldin Index of the unigram model: 4.8797884670479

Cluster  0 : [('encrypt', 296), ('scheme', 274), ('key', 196), ('cryptographi', 1
83), ('attack', 167), ('protocol', 149), ('authent', 124), ('iot', 122), ('clou
d', 109), ('provid', 107), ('effici', 99), ('cryptograph', 93), ('devic', 91),
('techniqu', 83), ('new', 79)]
Cluster  1 : [('quantum', 252), ('cryptographi', 52), ('key', 45), ('protocol', 4
0), ('oper', 38), ('attack', 37), ('state', 33), ('gate', 32), ('circuit', 32),
('post-quantum', 31), ('program', 31), ('commun', 31), ('time', 31), ('also', 2
9), ('scheme', 28)]
Cluster  2 : [('control', 258), ('studi', 221), ('measur', 209), ('differ', 175),
('estim', 163), ('test', 163), ('structur', 156), ('task', 154), ('provid', 152),
('improv', 149), ('environ', 142), ('evalu', 141), ('work', 141), ('vision', 13
5), ('sensor', 132)]
Cluster  3 : [('object', 272), ('deep', 200), ('dataset', 198), ('featur', 193),
('train', 187), ('vision', 184), ('video', 163), ('neural', 149), ('visual', 14
8), ('accuraci', 134), ('track', 133), ('differ', 126), ('research', 126), ('impr
ov', 120), ('recognit', 112)]
Cluster  4 : [('program', 332), ('relat', 306), ('languag', 275), ('queri', 257),
('code', 217), ('optim', 201), ('graph', 200), ('memori', 141), ('transform', 13
8), ('time', 136), ('parallel', 136), ('techniqu', 134), ('show', 129), ('effic
i', 125), ('analysi', 125)]
```

In [ ]:

```
from scipy.sparse import vstack

# Divide the TF-IDF matrix into separate matrices for each cluster
clustered_documents = {i: [] for i in range(best_k)}
for doc_id, cluster_id in enumerate(labels):
    clustered_documents[cluster_id].append(unigram_tf_idf_vectors[doc_id])

# Apply SVD to each cluster's TF-IDF matrix and interpret the leading singular v
for i in range(best_k):
    # Convert the list of TF-IDF vectors for this cluster to a sparse matrix
    cluster_tf_idf_matrix = vstack(clustered_documents[i])

    svd = TruncatedSVD(n_components=1)
    svd.fit(cluster_tf_idf_matrix)
    leading_singular_vector = svd.components_[0]

    terms = unigram_tfidf_vectorizer.get_feature_names_out()

    # Get the terms with the highest coefficients in the leading singular vector
    top_indices = leading_singular_vector.argsort()[-5:][::-1]
    top_terms = [(terms[idx], leading_singular_vector[idx]) for idx in top_indic

    print(f"\nCluster {i} leading singular vector terms:")
    for term, coefficient in top_terms:
        print(f"{term} (coefficient: {coefficient:.4f})")
```

```
Cluster 0 leading singular vector terms:
encrypt (coefficient: 0.4741)
scheme (coefficient: 0.3690)
key (coefficient: 0.2680)
attack (coefficient: 0.2421)
cryptographi (coefficient: 0.2371)

Cluster 1 leading singular vector terms:
program (coefficient: 0.4419)
languag (coefficient: 0.3099)
code (coefficient: 0.2745)
graph (coefficient: 0.2196)
memori (coefficient: 0.1874)

Cluster 2 leading singular vector terms:
queri (coefficient: 0.5814)
relat (coefficient: 0.5026)
manag (coefficient: 0.1621)
store (coefficient: 0.1517)
graph (coefficient: 0.1319)

Cluster 3 leading singular vector terms:
quantum (coefficient: 0.9091)
protocol (coefficient: 0.1301)
cryptographi (coefficient: 0.1226)
key (coefficient: 0.1210)
attack (coefficient: 0.0895)

Cluster 4 leading singular vector terms:
vision (coefficient: 0.1947)
object (coefficient: 0.1771)
deep (coefficient: 0.1485)
track (coefficient: 0.1365)
train (coefficient: 0.1352)
```

Conclusion of the topics of the documents

Based on all experiments and also the baseline, we can conclude that this corpus must have at least 5 topics. These are the topics that kept reoccuring in both the baseline and the experiments:

- Topic 1: General programming and operating system/hardware, whose top keywords are program, language, code, graph, memory
- Topic 2: Quantum studies and application in security, whose top keywords are quantum, protocol, cryptography, key and attack
- Topic 3: Database, SQL and queries, whose top keywords are query, relational, management, store, graph
- Topic 4: Computer vision and robotics, whose top keywords are vision, object, deep (possibly deep learning), track (possibly in reinforcement learning automation), train
- Topic 5: Security and cryptography, whose top keywords are encryption, scheme, key, attack, cryptograph

4. Appendix

All the code for this exercise has been added with respect to each part for closest referencing. Therefore, we do not attach any more code here in the Appendix section