PROJECT ASSIGNMENT PRESENTATION

2024/02/21

HOW AND WHAT WE USE

- 使用 Python 撰寫 parser 來統計 title 和 keyword 數量
- 運用 set 和 dictionary(map)

ASSIGNMENT 2.1

- Code

```
# iterate through each file
for file_path in file_paths:
    with open(file_path, 'r', encoding="utf-8") as file:
        # initialize title and insideTI
        title = ""
        insideTI = False
        # iterate through each line
        for line in file:
            # starts of a title
            if line.startswith("TI "):
                insideTI = True
                title += line[3:].strip()
            # contents inside the TI tag
            elif line.startswith("   ") and insideTI:
                title += line[3:].strip()
            else:
                # print the current (title, file) and correspond (title, file) if the title has been seen before
                if title in titles:
                    print("---")
                    print(f'Title "{title}" appears in "{file_path}".')
                    print(f'Title "{title}" appears in "{title_file[title]}".')
                    print("---")
                # add title count and add it into the set and dictionary
                if title != "":
                    count += 1
                    titles.add(title)
                    title_file[title] = file_path
                # re-initialize title and insideTI
                title = ""
                insideTI = False

print(f"Number of titles: {count}")
print(f"Number of unique titles: {len(titles)}")
```

- Output

```
Number of titles: 26287
Number of unique titles: 26269
```

ASSIGNMENT 2.2

- Code

```
# iterate through each file
for file_path in file_paths:
    with open(file_path, 'r', encoding="utf-8") as file:
        # initialize keyword and insideDE
        keyword = ""
        insideDE = False
        # iterate through each line
        for line in file:
            # starts of keywords
            if line.startswith("DE "):
                insideDE = True
                keyword += line[3:].strip()
            # contents inside the DE tag
            elif line.startswith("   ") and insideDE:
                keyword += line[3:].strip()
            else:
                # add each keyword into the set and dictionary
                if keyword != "":
                    keyword = keyword.split(';')
                    for word in keyword:
                        word = word.strip() # .lower() to avoid same keywords but with different upper/lower characters
                        keywords.add(word)
                        # if word in keyword_count, increase keyword_count[word] by one
                        # else, set keyword_count[word] to one
                        if keyword_count.get(word, False):
                            keyword_count[word] += 1
                        else:
                            keyword_count[word] = 1
                # re-initialize keyword and insideDE
                keyword = ""
                insideDE = False

# sort the dictionary with it's value in descending order
sorted_keywords = sorted(keyword_count.items(), key=lambda x: x[1], reverse=True)

with open('output.txt', 'w') as file:
    for key, value in sorted_keywords:
        print(f"{key}({value})", file=file)
```

- Output

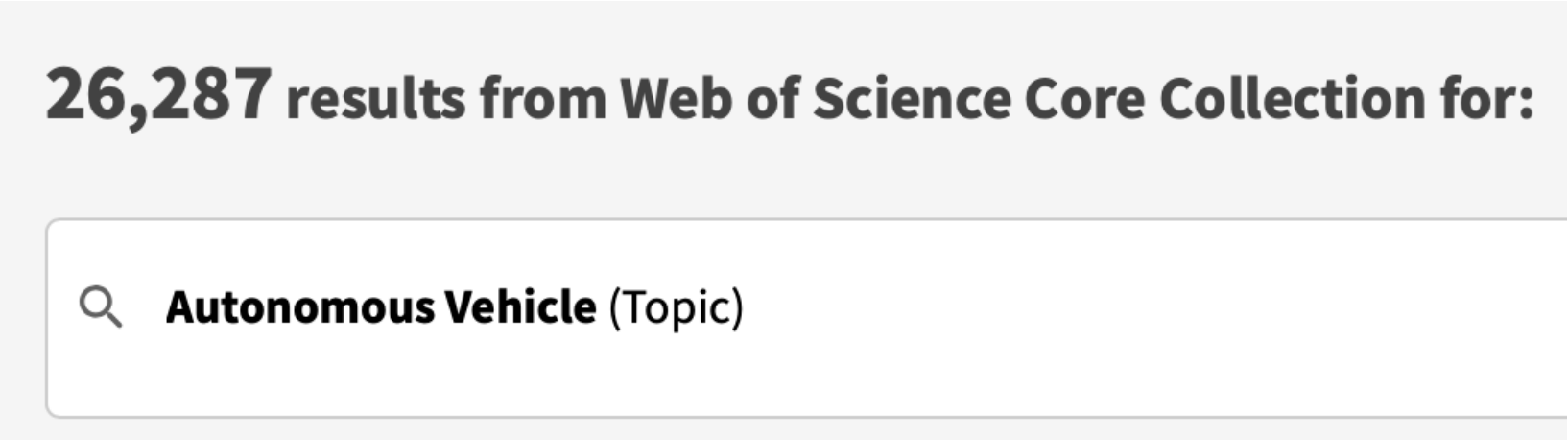
```
autonomous vehicles(2885)
autonomous vehicle(1102)
autonomous driving(982)
autonomous aerial vehicles(911)
path planning(737)
deep learning(723)
vehicle dynamics(721)
trajectory(715)
optimization(620)
task analysis(591)
safety(585)
sensors(571)
autonomous underwater vehicle(566)
navigation(529)
collision avoidance(520)
roads(498)
reinforcement learning(451)
uav(447)
object detection(435)
machine learning(415)
auv(414)
autonomous underwater vehicles(398)
feature extraction(359)
planning(356)
artificial intelligence(328)
unmanned aerial vehicles(327)
localization(322)
autonomous underwater vehicle (auv)(320)
computer vision(315)
cameras(305)
```

PROJECT ASSIGNMENT PRESENTATION

2024/03/06

SEARCH RESULTS FOR AUTONOMOUS VEHICLE

- 搜尋 Autonomous Vehicle 的資料筆數截圖（截取時間 2024/01/26）



DUPLICATE TITLES

- Code

```
        # starts of an author
        if line.startswith("AF "):
            insideAF = True
            author.append(line[3:].strip())
        # contents inside the AF tag
        elif line.startswith("   ") and insideAF:
            author.append(line[3:].strip())
        else:
            insideAF = False
        # starts of a title
        if line.startswith("TI "):
            insideTI = True
            title += line[3:].strip()
        # contents inside the TI tag
        elif line.startswith("   ") and insideTI:
            title += line[3:].strip()
        else:
            # print the current (title, file) and correspond (title, file) if the title has been seen before
            if title in titles:
                print("---")
                print(f'Title "{title}" appears in "{file_path}".')
                print(f'Title "{title}" appears in "{title_file[title]}".')
                # check if the authors are same
                print(len(author))
                for i in author:
                    print(i, end=" ")
                print()
                for i in title_author[title]:
                    print(i, end=" ")
                print()
                if len(set(author).intersection(title_author[title])) == len(author):
                    print(f'All Same!')
                else:
                    print(f'Different!')
                print("---")
            # add title count and add title, author into the set and dictionary
            if title != "" and len(author) != 0:
                count += 1
                titles.add(title)
                title_file[title] = file_path
                title_author[title] = author
                # re-initialize author and insideAF
                author = []
                insideAF = False
```

- Partial Output

```
---
Title "SoftPOSIT: Simultaneous pose and correspondence determination" appears in "autonomous_vehicle_data_records/18.txt".
Title "SoftPOSIT: Simultaneous pose and correspondence determination" appears in "autonomous_vehicle_data_records/17.txt".
4
David, P Dementhon, D Duraiswami, R Samet, H
David, P DeMenthon, D Duraiswami, R Samet, H
Different!
---
---
Title "Collaborative Motion Planning Based on the Improved Ant Colony Algorithmfor Multiple Autonomous Vehicles" appears in "autonomous_vehicle_data_records/26.txt".
Title "Collaborative Motion Planning Based on the Improved Ant Colony Algorithmfor Multiple Autonomous Vehicles" appears in "autonomous_vehicle_data_records/1.txt".
4
Alomari, Hakam W. Al-Badarneh, Amer F. Al-Alaj, Abdullah Khamaiseh, Samer Y.
Su, Shengchao Ju, Xiang Xu, Chaojie Dai, Yufeng
Different!
---
---
Title "Some statistical challenges in automated driving systems" appears in "autonomous_vehicle_data_records/21.txt".
Title "Some statistical challenges in automated driving systems" appears in "autonomous_vehicle_data_records/2.txt".
3
Caballero, William N. Insua, David Rios Naveiro, Roi
Caballero, William N. Insua, David Rios Naveiro, Roi
All Same!
---
```

DUPLICATE TITLES

- Results
  - 11 duplicate title papers have the same author, while the other 7 have different
  - 2 of the 11 papers have the author name not completely same

TRENDS

- 使用 Python 中的 Matplotlib 來呈現圖表
- 選取幾個比較有代表性的關鍵字來觀察
- 以下圖表皆以三年為一區間 (1995 ~ 2024)

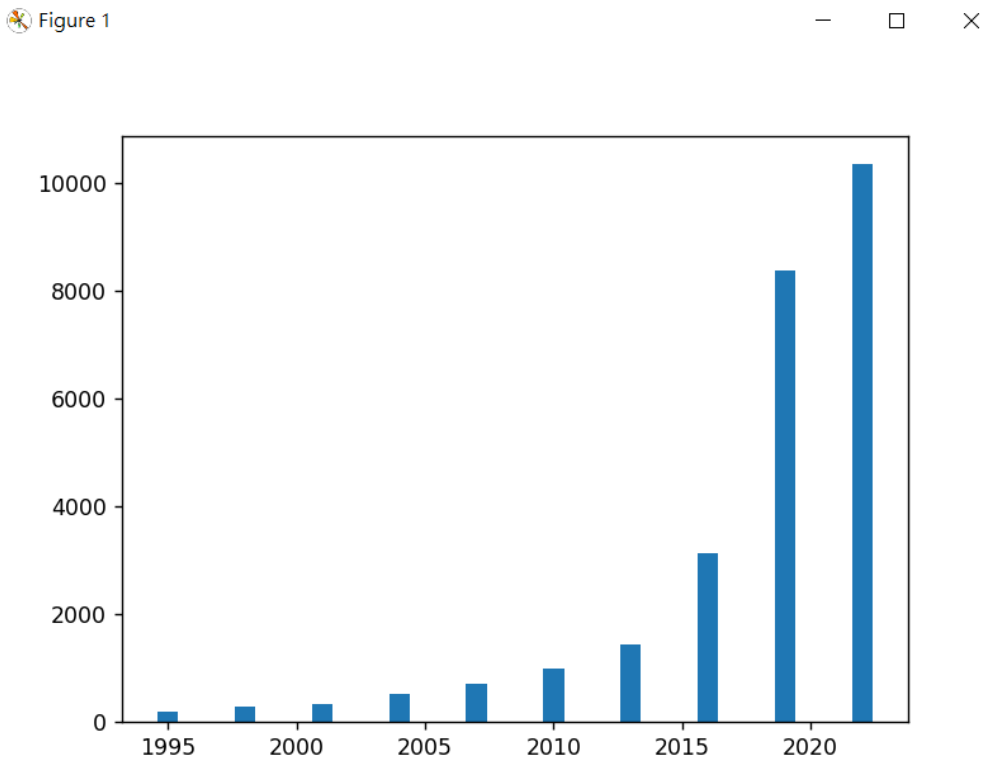
```
while x<=2024:
    x_axis.append(x)
    x+=y
y_axis=[0] * 10
for k, v in coun.items():
    y_axis[k]=v
plt.bar(x_axis, y_axis)
plt.title(s)
plt.yticks(range(0, max(y_axis)+25, 25))
plt.show()
```

TRENDS(AMOUNT OF PAPERS BY YEAR)

- Code

```
for file_path in file_paths:
    with open(file_path, 'r', encoding="utf-8") as file:
        # iterate through each line
        for line in file:
            # starts of keywords
            if line.startswith("PY "):
                z+=1
                year=line[3:].strip()
                ke = int((int(line[3:].strip()) - x) / y)
                if ke in coun:
                    coun[ke] += 1
                else:
                    coun[ke] = 1
```

- Graph

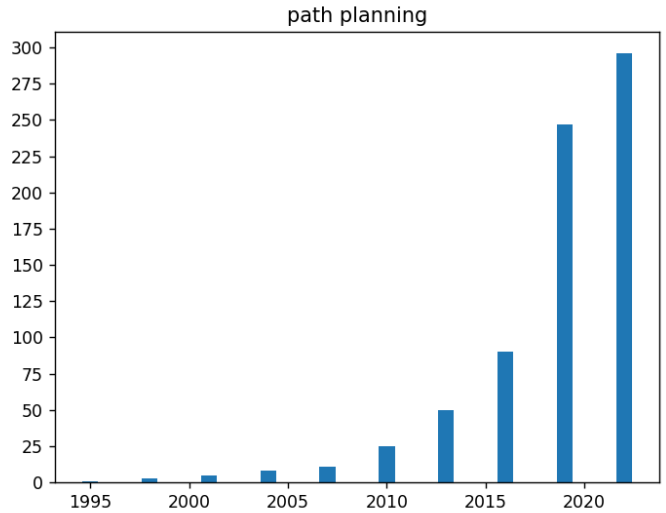


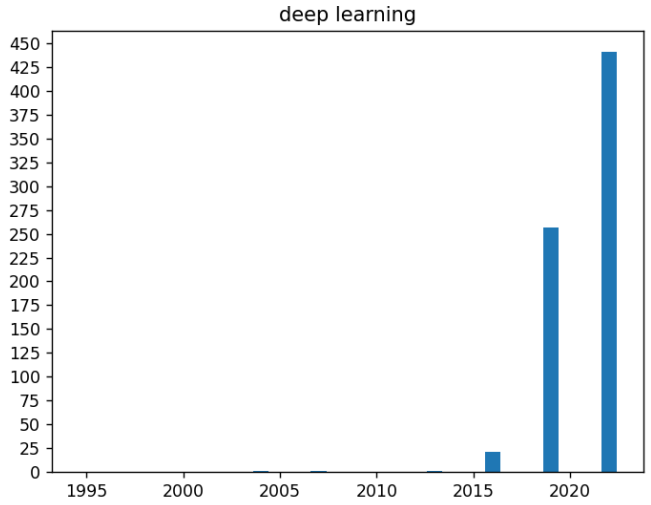
TRENDS

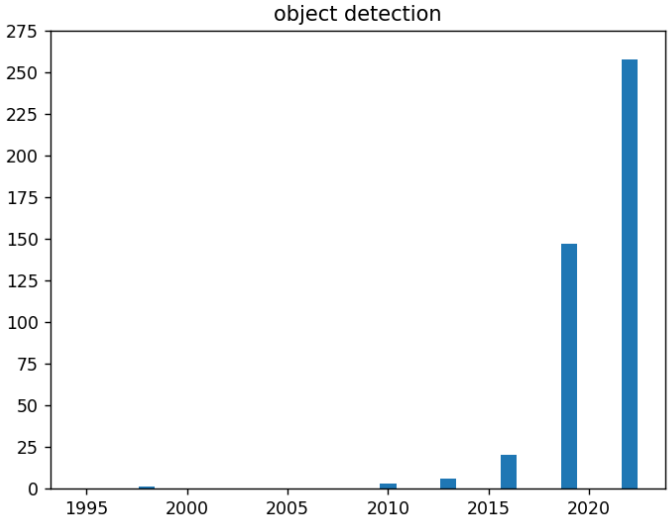
- Code

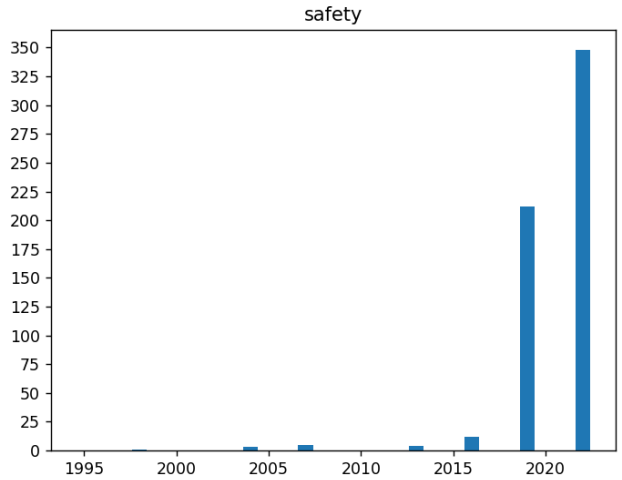
```
else:
    # add each keyword into the set and dictionary
    if keyword != "":
        keyword = keyword.split(';')
        for word in keyword:
            word = word.strip().lower() # .lower() to avoid same keywords bu
            if(word==s):
                line = file.readline()
                while line.startswith("PY ") != True:
                    line = file.readline()
                if line.startswith("PY "):
                            z+=1
                            year=line[3:].strip()
                            ke = int((int(line[3:].strip()) - x) / y)
                            if ke in coun:
                                coun[ke] += 1
                            else:
                                coun[ke] = 1
    # re-initialize keyword and insideDE
    keyword = ""
    insideDE = False
```

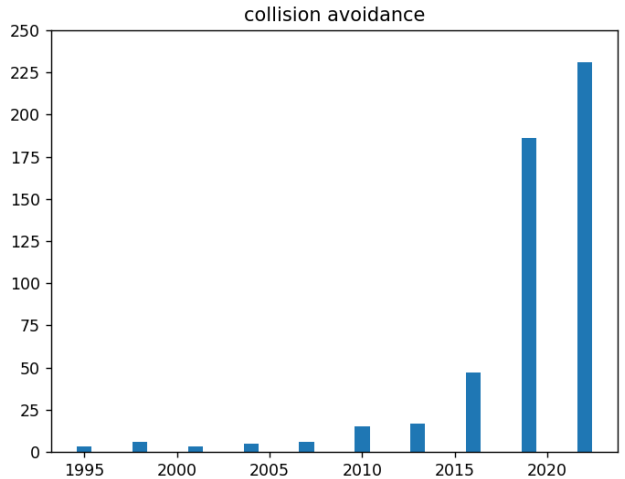
TRENDS

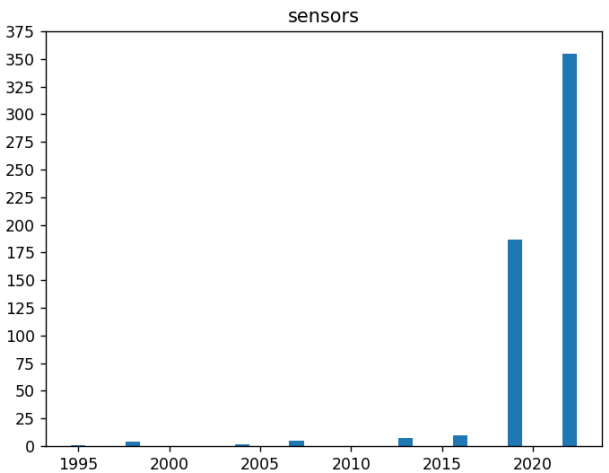

path planning
300
275
250
225
200
175
150
125
100
75
50
25
0
1995
2000
2005
2010
2015
2020


deep learning
450
425
400
375
350
325
300
275
250
225
200
175
150
125
100
75
50
25
0
1995
2000
2005
2010
2015
2020


object detection
275
250
225
200
175
150
125
100
75
50
25
0
1995
2000
2005
2010
2015
2020


safety
350
325
300
275
250
225
200
175
150
125
100
75
50
25
0
1995
2000
2005
2010
2015
2020


collision avoidance
250
225
200
175
150
125
100
75
50
25
0
1995
2000
2005
2010
2015
2020


sensors
375
350
325
300
275
250
225
200
175
150
125
100
75
50
25
0
1995
2000
2005
2010
2015
2020

PROJECT ASSIGNMENT PRESENTATION

2024/03/20

ANALYSIS BY COUNTRY

- 根據 PA（作者住址）來抓取國家
- 國家大部分會寫在最後面，但是並不是每一個都有按照格式填寫（可能是作者個人原因/國家格式不同）
- 抓取方式：
  - 目前用空白做分割並將最後一個字串移除非字母的字元，該字串就當作國家名稱（無法保證準確度）
  - 未來可能可以使用自然語言處理的相關工具協助判定
- 選取數個排名靠前的或是較有代表性的國家做圖表分析

ANALYSIS BY COUNTRY(AMOUNT OF PAPERS BY COUNTRY)

- Code

```
# iterate through each file
for file_path in file_paths:
    with open(file_path, 'r', encoding="utf-8") as file:
        # initialize address and insidePA
        address = ""
        insidePA = False
        # iterate through each line
        for line in file:
            # starts of address
            if line.startswith("PA "):
                insidePA = True
                address += line[3:].strip()
            # contents inside the PA tag
            elif line.startswith("   ") and insidePA:
                address += line[3:].strip()
            else:
                # pick out the country name and add it into the set and dictionary
                # but not every data has the same format, so some country cannot be captured by the following code
                if address != "":
                    # get the country name from address (usually the last word) and remove the non-alphabets
                    country = re.sub(r'[^a-zA-Z]', '', address.split(' ')[-1])
                    countries.add(country)
                    if country_count.get(country, False):
                        country_count[country] += 1
                    else:
                        country_count[country] = 1
                # re-initialize address and insidePA
                address = ""
                insidePA = False
```

- Output

```
USA(11585)
ENGLAND(5919)
SWITZERLAND(3924)
NETHERLANDS(2427)
GERMANY(573)
STATES(341)
KOREA(284)
CHINA(212)
JAPAN(149)
INDIA(89)
FRANCE(80)
SINGAPORE(79)
POLAND(60)
CANADA(57)
(49)
SPAIN(47)
CROATIA(46)
GB(35)
HUNGARY(31)
TAIWAN(23)
LITHUANIA(22)
ROMANIA(18)
NORWAY(15)
AUSTRIA(15)
YORKSHIREENGLAND(14)
TURKIYE(13)
CAMBSENGLAND(11)
SLOVAKIA(9)
BRAZIL(8)
BULGARIA(8)
AFRANCE(8)
IRELAND(8)
TURKEY(8)
ITALY(8)
```

ANALYSIS BY COUNTRY

- Code

```
else:
    # pick out the country name and add it into the set and dictionary
    # but not every data has the same format, so some country cannot be captured by the following code
    if address != "":
        # get the country name from address (usually the last word) and remove the non-alphabets
        country = re.sub(r'[^a-zA-Z]', '', address.split(' ')[-1])
        countries.add(country)
        if country_count.get(country, False):
            country_count[country] += 1
        else:
            country_count[country] = 1
        if country == target_country:
            line = file.readline()
            while line.startswith("PY ") != True:
                line = file.readline()
            if line.startswith("PY "):
                    t = int((int(line[3:].strip()) - year) / diff)
                    if t in year_count:
                        year_count[t] += 1
                    else:
                        year_count[t] = 1
    # re-initialize address and insidePA
    address = ""
    insidePA = False
```

ANALYSIS BY COUNTRY