keyword_analysis

November 9, 2017

```
In [1]: import pickle
        import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        %matplotlib inline

In [2]: dat = pickle.load(open("extracted_files/extracted_raw.p", "rb"))
        dat = list(filter(None, dat))
        df = pd.DataFrame(dat)

In [3]: df.columns

Out[3]: Index(['abstract', 'authors', 'cite_count', 'cover_date', 'doi', 'keywords',
               'publication_name', 'reference_count', 'subject_area', 'title', 'type',
               'volume'],
              dtype='object')

In [4]: abstracts = " ".join(list(df.abstract))

In [5]: abstracts = abstracts.lower()

In [6]: from nltk.tokenize import sent_tokenize, word_tokenize

In [7]: words = word_tokenize(abstracts)

In [8]: keywords = pickle.load(open("keywords.p", "rb"))

In [9]: also_keywords = []
        for key in keywords:
            also_keywords.append(key.split())

        keywords = []
        for sublist in also_keywords:
            for item in sublist:
                keywords.append(item.lower())

In [25]: key_dict = {}
         for key in keywords:
             for word in words:
                 if word.lower() == key.lower():
                     if key in key_dict.keys():
                         key_dict[key] += 1
                     else:
                         key_dict[key] = 1
```

```
In [27]: import operator
         # x = {1: 2, 3: 4, 4: 3, 2: 1, 0: 0}
         sorted_x = sorted(key_dict.items(), key=operator.itemgetter(1))
```

```
In [28]: for_plotting = sorted_x[-15:]
```

```
In [29]: for_plotting
```

```
Out[29]: [('problem', 1668),
          ('optimization', 1692),
          ('computational', 1988),
          ('model', 2052),
          ('for', 2084),
          ('intelligence', 2590),
          ('artificial', 2754),
          ('network', 2888),
          ('algorithm', 5418),
          ('learning', 6258),
          ('system', 8138),
          ('data', 8352),
          ('and', 11298),
          ('the', 12384),
          ('of', 38465)]
```

```
In [30]: to_remove = "of,the,and,for,a,i,in,an"
         to_remove = to_remove.split(",")
         to_remove
```

```
Out[30]: ['of', 'the', 'and', 'for', 'a', 'i', 'in', 'an']
```

```
In [31]: # for i, val in enumerate(for_plotting):
         #     print(i,val)
         #     if key in to_remove:
         i = 0
         while 1:
             if for_plotting[i][0] in to_remove:
                 del for_plotting[i]
             else:
                 i += 1
             if i >= len(for_plotting):
                 break
```

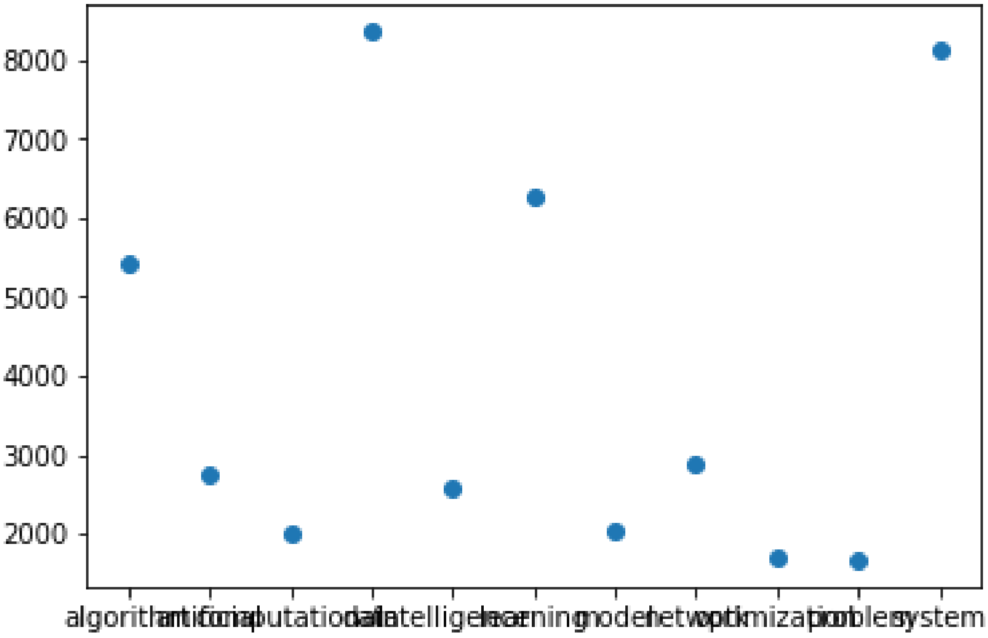
```
In [32]: for_plotting
```

```
Out[32]: [('problem', 1668),
          ('optimization', 1692),
          ('computational', 1988),
          ('model', 2052),
          ('intelligence', 2590),
          ('artificial', 2754),
          ('network', 2888),
          ('algorithm', 5418),
          ('learning', 6258),
          ('system', 8138),
          ('data', 8352)]
```

```
In [33]: x = []
         y = []
         for val in for_plotting:
             x.append(val[0])
             y.append(val[1])
         # pickle.dump([x,y], open("for_plotting.p", "wb"))
```

```
In [34]: # x, y = pickle.load(open("for_plotting.p", "rb"))
```

```
In [35]: plt.scatter(x,y)
         plt.show()
```



```
In [37]: li = list(df.subject_area)
In [38]: flat_li = []
         for sublist in li:
             for item in sublist:
                 flat_li.append(item)
```

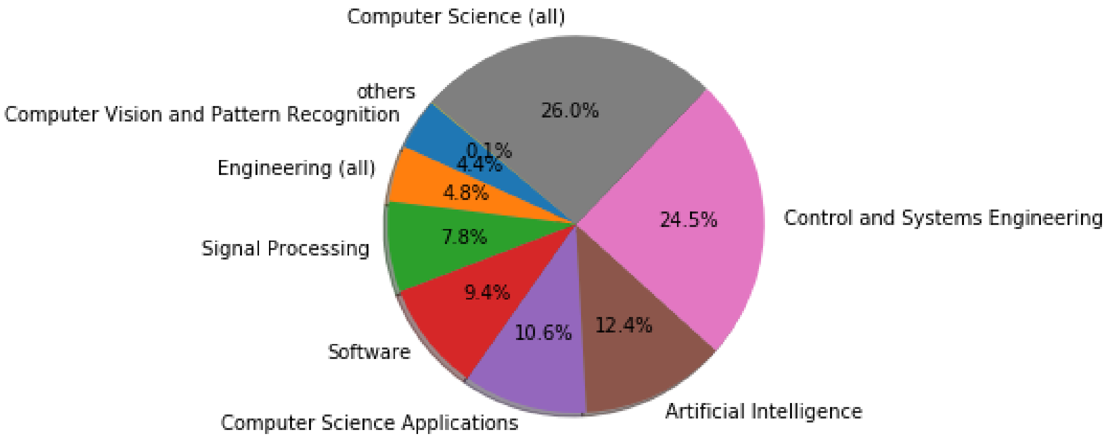
```
In [41]: from collections import Counter
```

```
In [42]: count = Counter(flat_li)
```

```
In [44]: sorted_count = sorted(count.items(), key=operator.itemgetter(1))
```

```
In [52]: temp = sorted_count[-8:]

         labels = []
         sizes = []

         for item in temp:
             labels.append(item[0])
             sizes.append(item[1])

         labels.append("others")
         sizes.append(1)
```

```
In [54]: # Data to plot
         # colors = ['gold', 'yellowgreen', 'lightcoral', 'lightskyblue']
         # explode = (0.1, 0, 0, 0)  # explode 1st slice

         # Plot
         plt.pie(sizes, labels=labels,
                 autopct='%1.1f%%', shadow=True, startangle=140)

         plt.axis('equal')
         plt.show()
```


Computer Science (all)
others
Computer Vision and Pattern Recognition
Engineering (all)
Signal Processing
Software
Computer Science Applications
Artificial Intelligence
Control and Systems Engineering