

자연어처리 프로젝트(2)

k-드라마 데이터셋을 활용한 k-드라마 네트워크 분석


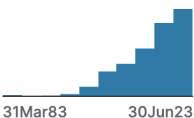
20171768 김영기

(0) 드라마 데이터셋 선택 이유

과제의 기존 목표인 한국 영화 네트워크를 구성하기 위해 리서치하던 도중 유사한 관련 과제 및 논문을 많이 발견하였습니다. 좀 더 색다른 정보를 얻기 위해 영화 데이터가 아닌 드라마 데이터를 활용하여 네트워크를 구성, 분석하였습니다.

(1) k-드라마 데이터셋

데이터 수집을 위해 [TMDB](#)(The Movie Database)를 활용하였습니다. 데이터를 직접 크롤링하기 보단, 제공되는 데이터셋을 사용하기 위해 2022년까지의 k-드라마 데이터를 담고 있는 캐글의 '[TMDB - KDramas \(2022\)](#)' 데이터셋을 사용하였습니다.

tmdb_id	name	original_name	keywords	airing_date	poster
ID of the series (directly from TMDB, can be further used with their API)	Name of the series	Original name of the series	Keywords appended to the series	First airing date	Path to the image (TMDB)
	1501 unique values	1504 unique values	[null] 36% romance 3% Other (924) 61%		[null] https://img.tmbd.org/t/p/w500/93405.jpg Other (14)
93405	Squid Game	오징어 게임	secret organization, challenge, survival, fictional game show, cruelty, game, death match, prize	2021-09-17	https://img.tmbd.org/t/p/w500/93405.jpg

(2) 그래프 구성

1. 그래프 구축

파이썬을 활용하여 csv파일의 정보를 읽어 베이컨 수와 같은 방식의 네트워크 그래프를 구성하였습니다. 그래프는 파이썬의 [networkx](#) 패키지를 활용했습니다.

예시 코드(그래프 구성만 염두해 둔 간략화된 코드)

```
import pandas as pd
import networkx as nx

data = pd.read_csv("series.csv")
graph = nx.Graph()

for index, row in data.iterrows():
```

```

movie_title = row['original_name']
cast_ids = row['cast_ids']

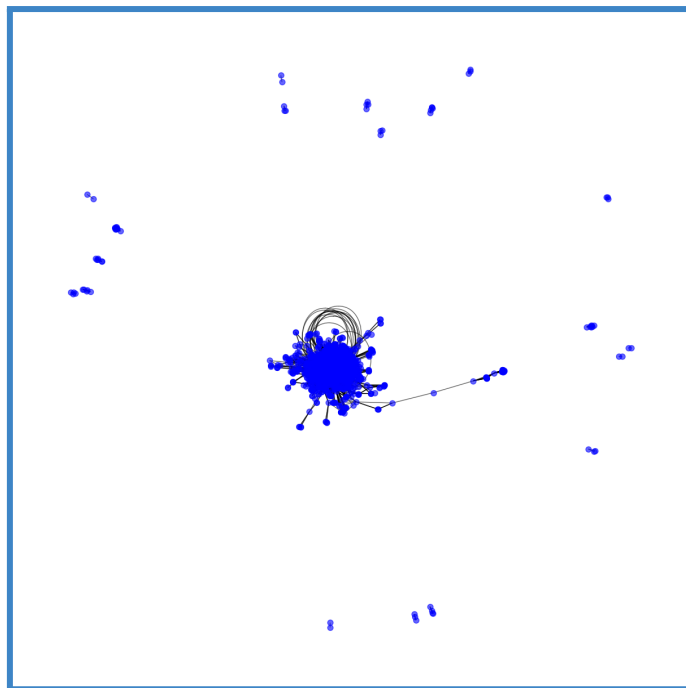
if pd.isnull(cast_ids):
    continue

cast_ids = str(cast_ids).split(',')
cast_ids = [int(x) for x in cast_ids]

# 같은 작품에 출연한 배우들 간의 연결을 추가합니다.
for i in range(len(cast_ids)):
    for j in range(i + 1, len(cast_ids)):
        graph.add_edge(cast_ids[i], cast_ids[j])

```

결과 :



결과를 확인하면 몇개의 아주 작은 그래프를 제외한다면, 모두 하나의 큰 그래프로 연결되어있는 형태입니다.

2. 그래프 사이 거리 계산

networkx의 `shortest_path` 함수를 활용하여 최단거리와 해당 경로를 계산할 수 있습니다.

```

def find_kevin_bacon_number_with_movies(graph, source_id, target_id,
actor_name_dict):
    try:
        path = nx.shortest_path(graph, source_id, target_id)
        result = [actor_name_dict[source_id]]
        for i in range(len(path) - 1):
            movie_title = graph[path[i]][path[i + 1]]["movie"]

```

```

        result.append(movie_title)
        result.append(actor_name_dict[path[i + 1]])
    return " -> ".join(str(x) for x in result)
except nx.NetworkXNoPath:
    return "No path found between {} and {}".format(source_id, target_id)
except KeyError:
    return "Actor ID not found in the actor_name_dict."

```

예시 :

```

Ahn Nae-sang -> 18 어게인 -> Wi Ha-jun -> 오징어 게임 -> Lee Jung-jae
Ha Jung-woo -> 수리남 -> Park Hae-soo -> 오징어 게임 -> Lee Jung-jae

```

(3) 핵심 노드 분석

핵심 노드는 대표적인 5가지 알고리즘을 활용하여 분석해보았습니다.

1. Degree Centrality (연결 중심성)

노드의 연결 수를 기준으로 중심성을 측정합니다. 높은 연결 중심성을 가진 노드는 다른 노드와 많은 연결을 가지고 있습니다.

```

degree centrality = nx.degree_centrality(graph)

```

top 5:

```

Top 5 actors by Degree Centrality: [('Ahn Nae-sang', 0.1521219366407651), ('Kim
Mi-kyeong', 0.12163777644949192), ('Jeon Kuk-hwan', 0.10789001793185893), ('Kim
Hee-jung', 0.1075911536162582), ('Jang Hyun-sung', 0.10639569635385535)]

```

2. Eigenvector Centrality (고유벡터 중심성)

노드의 중요도가 연결된 다른 노드의 중요도에 기반합니다. 높은 공벡터 중심성을 가진 노드는 중요한 노드들과 연결되어 있습니다.

```

eigenvector centrality = nx.eigenvector_centrality(graph)

```

top 5:

```

Top 5 actors by Eigenvector Centrality: [('Ahn Nae-sang', 0.11360912848217507),
('Jeon Kuk-hwan', 0.0979426221113596), ('Kim Mi-kyeong', 0.09785355341172743),
('Uhm Hyo-seop', 0.09780844199943635), ('Kim Hee-jung', 0.09120783043490083)]

```

3. Closeness Centrality (근접 중심성)

노드가 다른 모든 노드에 얼마나 가까운지를 측정합니다. 높은 근접 중심성을 가진 노드는 다른 노드와의

거리가 짧습니다.

```
closeness centrality = nx.closeness centrality(graph)
```

top 5:

```
Top 5 actors by Closeness Centrality: [('Ahn Nae-sang', 0.48715774218553437),  
('Kim Mi-kyeong', 0.47608103917551997), ('Uhm Hyo-seop', 0.46890358791160336),  
('Kim Hee-jung', 0.4671940079644267), ('Jeon Kuk-hwan', 0.46671755719872876)]
```

4. Betweenness Centrality (매개 중심성)

그래프 내의 노드들 간의 모든 최단 경로 중, 특정 노드가 얼마나 자주 등장하는지를 측정합니다. 높은 매개 중심성을 가진 노드는 그래프 내에서 다른 노드들 사이의 정보 전달이나 상호작용에 있어 중요한 역할을 하는 "브리지" 역할을 합니다. 이러한 노드들은 그래프 내에서 중요한 연결점이며, 정보 전달이나 관계의 매개 역할을 담당합니다.

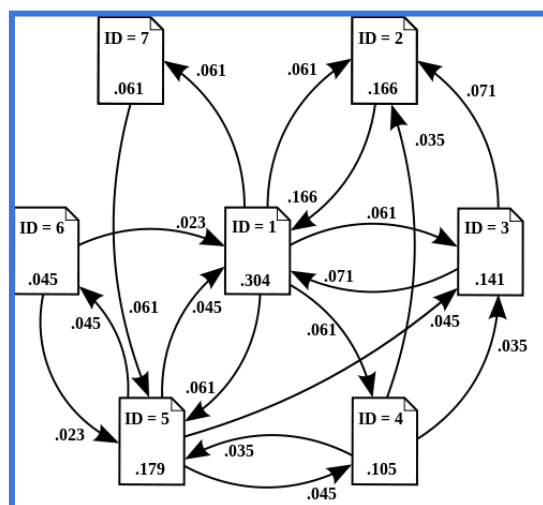
```
betweenness centrality = nx.betweenness centrality(graph)
```

top 5:

```
Top 5 actors by Betweenness Centrality: [('Ahn Nae-sang', 0.020752046259715298),  
('Choi Ji-su', 0.0193795626003334), ('Jung Soo Hyun', 0.016815920131303737),  
('Ji Won', 0.016241064225003283), ('Lee Jong Hyuk', 0.015665850932376255)]
```

5. PageRank

웹 페이지 랭킹을 위해 Google에서 개발된 알고리즘으로, 노드의 중요도를 반복적으로 계산하여 추정합니다. 중요한 노드들과 연결되어 있는 노드는 더 높은 PageRank 값을 가집니다.



```
pagerank = nx.pagerank(graph)
```

top 5:

```
Top 5 actors by PageRank: [('Ahn Nae-sang', 0.0025424344045976532), ('Kim  
Mi-kyeong', 0.0019906681619910323), ('Kim Hee-jung', 0.0017779341019577906),  
('Jang Hyun-sung', 0.0017730718524803495), ('Jeon Kuk-hwan',  
0.0016942320089073015)]
```

(4) 결론

색다른 결과를 얻기 위해 k-드라마 데이터셋을 활용하여 k-드라마 배우 네트워크를 분석해보았습니다. 분석 결과 여러 지표에서 높은 중요도를 가진 노드인 안내상(Ahn Nae-sang) 배우가 가장 중요한 노드라는 결론을 얻게 되었습니다.