UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Gº en Ingeniería Informática

TFG Ingeniería Informática:

NetExtractor 2.0

Presentado por Alberto Diez Busto

en Universidad de Burgos — 7 de julio de 2022Tutores: D. José Manuel Galán Ordax

y Dña. Virginia Ahedo García

D. José Manuel Galán Ordax y Dña. Virginia Ahedo García, profesores del departamento de Ingeniería Civil, área de Organización de Empresas.

Exponen:

Que el alumno D. Alberto Diez Busto, con DNI 71305786J, ha realizado el Trabajo final de Grado en Ingeniería Informática titulado NetExtractor 2.0.

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 7 de julio de 2022

Vo. Bo. del Tutor:

Vo. Bo. del Tutor:

Resumen

En la actualidad, la ciencia de redes es uno de los campos de investigación que más se está desarrollando y tiene mucha popularidad.

Dentro de este campo, una de sus ramificaciones que está siendo cada vez más importante es la red dinámica, debido a que es importante saber cómo ha ido modificándose la red a lo largo del tiempo para así analizar cómo han evolucionan los nodos y cómo se comportan los componentes. Esto con las redes convencionales no se puede tener en cuenta.

Por ello surgió el proyecto de NetExtractor 2.0. La intención de NetExtractor 2.0 es la de poder conocer como los personajes dentro de una novela o una película se van conociendo y empiezan a tener relaciones entre ellos a lo largo del tiempo, para así entender mejor como se han ido desarrollando las amistades o enemistades entre los personajes y analizar de una forma más correcta como se ha llegado al final de la red.

Descriptores

Generador de redes de interacción, visualización de red dinámica, informes de red dinámica, Python, aplicación web.

Abstract

Nowadays, network science is one of the research fields that is developing the most and is very popular.

Within this field, one of its branches that is becoming increasingly important is the dynamic network, because it is important to know how the network has been changing over time in order to analyze how the nodes have evolved and how they behave. the components. This with conventional networks cannot be taken into account.

That is why the NetExtractor 2.0 project arose. The intention of NetExtractor 2.0 is to be able to know how the characters in a novel or a movie get to know each other and begin to have relationships with each other over time, in order to better understand how friendships or enmities between them have developed. the characters and analyze in a more correct way how the end of the network has been reached.

Keywords

Interaction Network Generator, Dynamic Network Visualization, Dynamic Network Reports, Python, Web Application.

Índice General

Índice de Ilustraciones

A. Introducción

La ciencia de redes es un campo de investigación donde se estudian redes complejas y las características y métricas de dichas redes. En ellas tenemos a actores que están representados por nodos y las conexiones entre esos actores que serán los enlaces. Las redes complejas tienen una diversidad de ramificaciones como pueden ser redes informáticas, redes semánticas, redes sociales, etc.

La ciencia de redes se puede utilizar para diversos usos, desde capturar a un asesino mediante la red de contactos que tiene, hasta analizar grupos sociales para analizar cuales pueden ser los focos principales ante una posible transmisión de una enfermedad y poder así inmunizarlos para acabar con la enfermedad en el menor tiempo posible. Además, grandes empresas utilizan con frecuencia esta ciencia, como por ejemplo Google, que la utiliza para sugerir páginas ante una búsqueda. Esta ciencia nos puede ayudar en el ámbito profesional, ya que se puede encontrar la mejor opción a la hora de encontrar trabajo, o para difundir información a tus compañeros de la forma óptima. En definitiva, la ciencia de redes es cada vez más popular en la gran mayoría de ámbitos.

Este fue uno de los motivos para elegir este proyecto. Dicho proyecto viene de otros dos proyectos anteriores. El más actualizado, NetExtractor, permite al usuario crear un diccionario de personajes a partir de una novela y una película. Estos personajes iban a interaccionar entre ellos si estaban dentro de una misma escena por parte de la película, y si el número de palabras entre los dos personajes no sobrepasaba el máximo que es dado por el usuario en una novela. Una vez conocidos los nodos y los enlaces se creaba la red y se podían conocer las métricas resultantes de la red.

NetExtractor 2.0 presenta un nuevo termino que es red dinámica. La red dinámica es un grafo donde los nodos, enlaces y peso de cada uno de ellos se modificarán según el intervalo de tiempo en el que este la red. Por lo tanto, lo que se implementará en NetExtractor 2.0 es que, a partir de una novela o una película, se cree una red dinámica donde los personajes de las novelas o películas interaccionaran a lo largo del tiempo y se podrá tener una mejor idea de porque ha surgido la red final de la película o novela.

B. Objetivos del proyecto

En esta sección vamos a explicar cuáles son los objetivos de nuestro proyecto, que están divididos en objetivos generales, técnicos y personales.

Este proyecto parte de dos trabajos previos que son Ububooknet que fue presentado por el alumno Luis Miguel Cabrejas Arce, y NetExtractor que fue presentado por el alumno Jorge Navarro Gonzalez. NetExtractor, que es la versión más actualizada del proyecto se caracteriza en la extracción de los personajes de novelas si el formato es ePub o películas si le damos un guión de imsdb. Una vez se tienen los personajes se identificarán las interacciones que tienen entre ellos, si es una película dos personajes tendrán una interacción si aparecen en la misma escena, si es una novela se tendrá que especificar si se tienen en cuenta los capítulos o no y la distancia máxima de palabras que puede haber para que haya un enlace entre los dos personajes.

Objetivos generales

En el siguiente apartado describiremos cuales han sido los objetivos generales de nuestro proyecto, o lo que es lo mismo, los objetivos marcados al comienzo del proyecto.

- Crear una red dinámica tanto para el formato de película como para el formato de ePub. La red dinámica consistirá en una red en la que los personajes (nodos) e interacciones entre personajes (enlaces) irán apareciendo según el momento de la película o de la novela en que hayan aparecido.
- Generador de informe por cada intervalo de tiempo. Según el intervalo de tiempo en el que estes, se generará un informe de los datos que hayas elegido, de ese intervalo de tiempo.
- Generador de informe dinámico. Generará el informe de los datos que hayas asignado, desde el intervalo inicial hasta el intervalo que desee.
- Exportar la red dinámica en extensión apta para Gephi.
- Visualización de la animación de la red dinámica. Se podrá visualizar la animación del intervalo en el que se está actualmente o de la red dinámica entera.

Objetivos técnicos

Este apartado consta de todos los objetivos de carácter técnico, o sea, las herramientas que se han utilizado en el proyecto y los detalles de la implementación.

- El desarrollo del proyecto se ha realizado mediante la metodología Scrum.
- La herramienta asignada para la gestión de proyectos ha sido ZenHub.
- La herramienta asignada para el control de versiones ha sido GitHub.
- La herramienta asignada para la gestión del repositorio local y remoto ha sido GitKraken.
- El lenguaje de programación base utilizado ha sido Python. En Python se ha realizado la generación de la red dinámica como su posterior análisis. La librería utilizada para la creación de redes dinámicas ha sido DyNetX. Las librerías que hemos utilizado y ya estaban implementadas en el proyecto NetExtractor son: NetworkX, NumPy,

Matplotlib, Python-louvain, Scipy, BeautifulSoup4 y Ply. Además de estás se han tenido que añadir las librerías: IPython y Ffmpeg-python.

- Para el desarrollo web hemos utilizado Flask, que ya estaba implementada en el proyecto previo.

Objetivos personales

En este último apartado se definirán los objetivos personales que nos hemos propuesto:

- Utilizar la metodología SCRUM para comprender su utilidad y funcionamiento
- Ser capaz de desarrollar una aplicación web como lenguaje principal Python y utilizando Flask.
- Aprender a utilizar nuevas herramientas que desconocía.

C. Conceptos teóricos

En esta sección explicaremos los conceptos teóricos que han sido necesarios para poder realizar el proyecto.

Teoría de Grafos

Conceptos básicos de grafos

Diada: Tipo de relación entre dos elementos (nodos y enlaces, por ejemplo).

Red o grafo: representa un conjunto de nodos y sus relaciones. Es decir, un conjunto de diadas. En una red la relación puede representar cualquier factor, pero tiene que ser el mismo para toda la red.

Tipos de enlaces

Enlaces dirigidos: tienen origen y destino. Pueden ser recíprocas o no serlo.

Enlaces no dirigidos: bidireccionales.

Tipos de redes

Redes múltiples o simples

Redes múltiples: Si presentan enlaces múltiples. Es decir, 2 o más enlaces que relacionan el mismo par de nodos. También se considera a una red múltiple si esta presenta auto-enlaces (enlaces de un nodo a sí mismo).

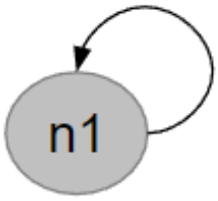


Figura 2 Autoenlace

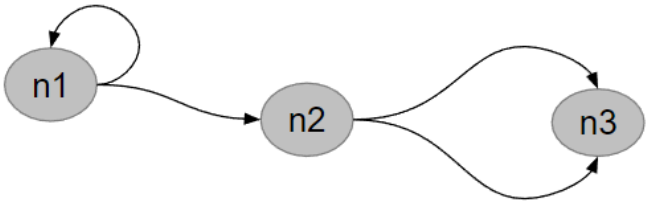


Figura 1 Red múltiple

Redes simples: Redes que no son múltiples, es decir, que no presentan ni enlaces múltiples ni auto-enlaces.

Redes binarias o pesadas

Redes binarias: Si los enlaces únicamente representan la existencia de una relación.

Redes pesadas: Si los enlaces tienen también asociado un valor cuantitativo (Ej: peso).

Redes acíclicas

Solo en caso de redes dirigidas. Una red es acíclica si no posee ciclos. Si podemos representar una red de tal forma que los enlaces siempre apunten hacia abajo, podemos asegurarnos que esa red es acíclica.

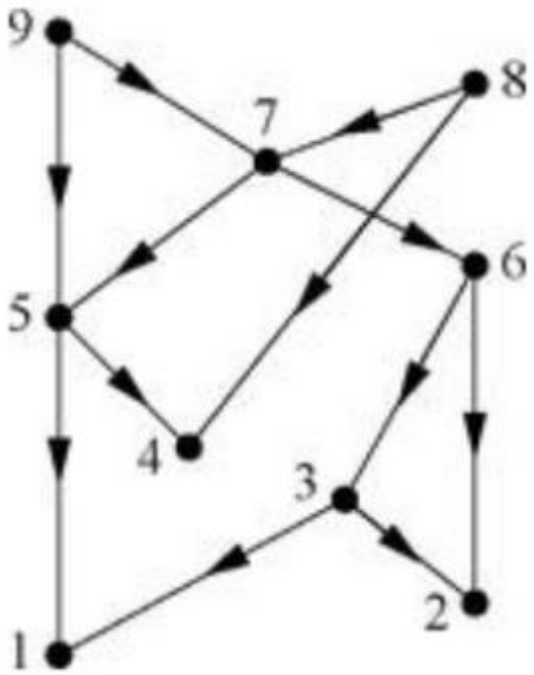


Figura 3 Red acíclica

Redes bipartitas y bimodales

Red la cual puede ser dividida en dos subconjuntos de nodos de tal forma que los nodos de un conjunto únicamente se relacionan con nodos del otro conjunto (y viceversa).

Una red bimodal es aquella que puede ser dividida en dos o más subconjuntos, pero que no tienen porqué relacionarse únicamente de un subconjunto al otro, sino que también pueden relacionarse entre ellos.

Una red bipartita se suele equiparar a una red bimodal, pero no son exactamente lo mismo, ya que no todas las redes bimodales son redes bipartitas. Aun así, ambos términos se suelen utilizar de forma idéntica.

Ejemplos:

- Redes bipartitas:
  - Red de contactos sexuales (solo individuos heterosexuales).
  - Red de relaciones compra-venta (suponiendo que una persona solo pueda ser o comprado ovendedor, no ambas).
- Red bimodal no bipartita:
  - Red de contactos sexuales con individuos de todas las sexualidades.
- Red bimodal bipartita:
  - Red de actores y películas en las que aparecen.

Siempre podemos representar una red bipartita o bimodal en dos redes unimodales.

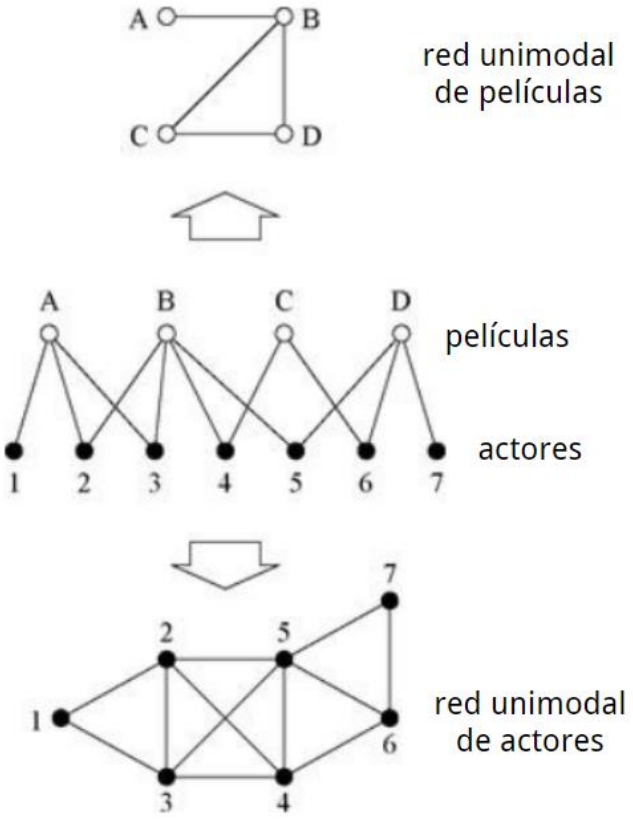


Figura 4 Red bipartita dividida

Todos estos conceptos se podrían resumir en la siguiente tabla:

Nodo	Clases	Unimodal Bimodal (e.g. bipartita) Multimodal
Enlace	Simetría	Dirigida No dirigida
	Peso	Binaria Pesada
	Múltiple y/o autoenlaces	Simple Múltiple
	Ciclos (redes dirigidas)	Cíclica Acíclica

Figura 5 Resumen

Patrones

Lattice network: red no dirigida en la que los nodos forman parte de un entramado regular.

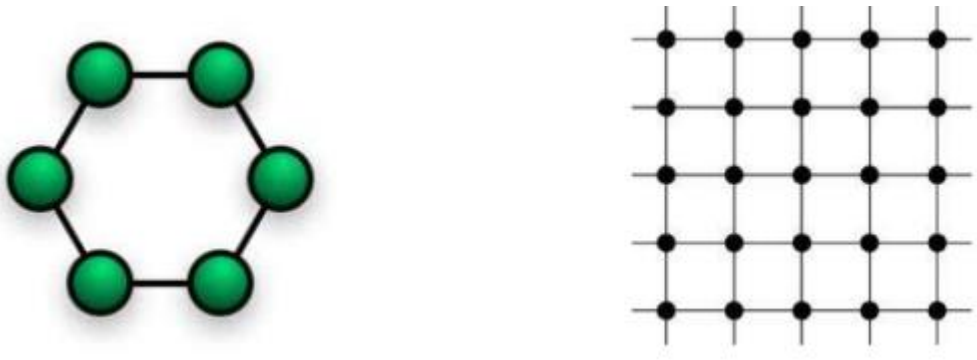


Figura 6 Lattice network

Árbol: red no dirigida en la cual todos los nodos están conectados entre sí y sin loops cerrados.

Nº de enlaces = Nº de nodos - 1

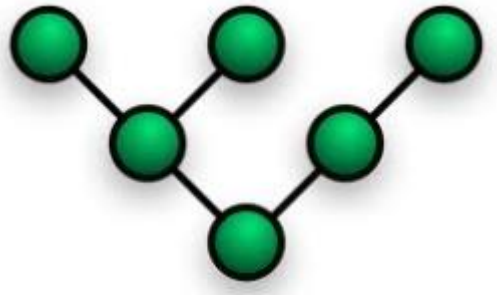


Figura 7 Árbol

Redes complejas

Red con muchos nodos y con un patrón de conexiones no regular. Ejemplo: Red neuronal.

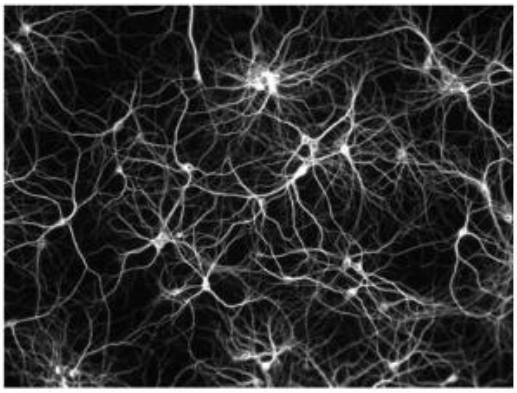


Figura 8 Red compleja

Representación de redes

Algunas opciones:

- Matemáticamente: Se utiliza una matriz denominada matriz de adyaciencia.
    - Las filas representarán los nodos de destino y las columnas los nodos de origen.
    - Las celdas valdrán 1 si existe un enlace que una el nodo origen de esa columna con el nodo destino de esa fila o 0 si no existe.
    - Si la red es ponderada (los enlaces tienen peso) los valores de las celdas no serán 0 o 1 sino el valor del peso en cuestión.
- Utilizando listas:
    - Lista de enlaces: {{'A','B', 1}, {'A','C', 1}, ...}
    - Lista de adyacencia: {'A': {'B', 1, 'C', '1', ...}, 'B': {'A', 1} ...}

Tamaño, densidad y grado

L = Nº de enlaces

N = Nº de nodos

Tamaño y densidad

- Redes pequeñas: menos de 10 nodos.
- Redes grandes: más de 1000 nodos.

La densidad de una red es la relación entre el número de enlaces y el número posible de enlaces.

- Para redes no dirigidas: 2 L / (N (N - 1))
- Para redes dirigidas: L / (N (N - 1))
  Una red es poco densa si N ≈ L

Grado

<u>Grado NO dirigido</u>

Nº de enlaces que tiene un nodo con otros nodos.

El grado medio será: 2 L / N

<u>Grado dirigido</u>

Tenemos que especificar:

- In-degree: Nº de enlaces que entran al nodo. Grado medio: L / N
- Out-degree: Nº de enlaces que salen del nodo. Grado medio: L / N
- Total-degree: Suma de ambos. k = k-in + k-out

Distribución de grado

Es mucho más representativo de una red que su grado medio, y muchas de las propiedades de las redes depende de ella. Se basa en calcular la probabilidad de que, cogiendo un nodo al azar, este tenga un grado k.

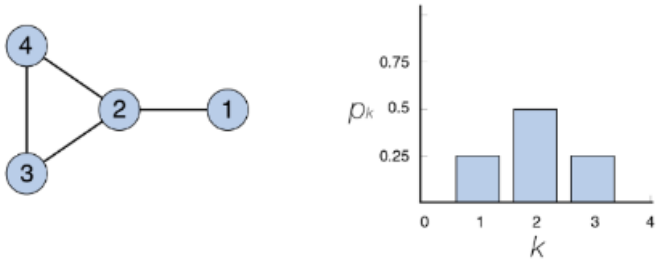


Figura 9 Distribución de grado

Ley de Metcalfe

El valor de una red depende proporcionalmente de N^2. No es perfecta, ya que hay más factores a tener en cuenta, como el número de enlaces.

Paseos y caminos

- Paseo: secuencia de nodos en la que cada nodo es, a su vez, origen del enlace que apunta al siguiente nodo y destino del enlace que sale del anterior nodo y que le apunta a él. Si no tenemos en cuenta la dirección se le llama cadena.
  - En los paseos simples (ningún nodo se repite) y cerrados:
    - En redes dirigidas lo llamamos ciclo, y las flechas estarán en el mismo sentido
    - En redes no dirigidas lo llamamos loop cerrado.
- Camino: un paseo en el que ningún nodo se repite (simple) y abierto (los nodos extremos solo tienen un enlace).

Distancia geodésica

- Longitud o distancia: Nº de enlaces atravesados en un camino. Si no existe camino entre dos nodos, se dice que la distancia es infinita.
- Camino más corto: Camino entre dos nodos de una red con la menor distancia de entre todos los posibles.
- Algoritmo de Dijkstra: Algoritmo utilizado para calcular los caminos más cortos.
- Diámetro: De entre todos los caminos más cortos de una red (de todas las posibles combinaciones de nodos), el de tamaño más grande.
- Camino medio de la red o average path lengh: valor medio de los caminos más cortos de una red.
- Caminos Eulerianos y Hamiltonianos
  - Caminos Eulerianos: Caminos que atraviesan todos los enlaces una única vez. Para ello, o existen 2 nodos de grado impar o ningún nodo de grado impar.

- Caminos Hamiltoniano: Caminos que atraviesan todos los nodos una única vez.

Componentes

Componente: el mayor conjunto de nodos posible tal que entre cada par exista al menos un camino.

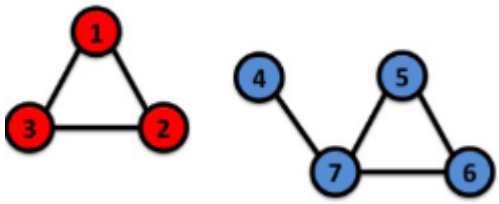


Figura 10 Componente

Componente mayor: Componente que contiene la mayoría de los nodos.

Modelo Small World

Modelo en el cual una red tiene un promedio de la longitud de los caminos cortos "significativamente" pequeña, ya que el Nº de nodos es bastante mayor en comparación. Ejemplos: Experimento de Milgram con las cartas (los 6 enlaces), el Nº de Beacon y el Nº de Erdös.

- Formalmente se dice que para que una red cumpla el modelo de Small World su distancia media tiene que escalar logarítmicamente con el tamaño de la red.
  $< d > \approx < \log N >$
  En redes sociales no se suele aplicar porque existe un elevado número de redundancias en las relaciones.

Cálculo de caminos más cortos

- Depth First Serch (DFS):
  - Empezamos por un nodo aleatorio.
  - Lo marcamos como visitado.
  - De sus vecinos no visitado elegimos uno (Ej: el que tenga el peso más bajo) y lo marcamos como visitado

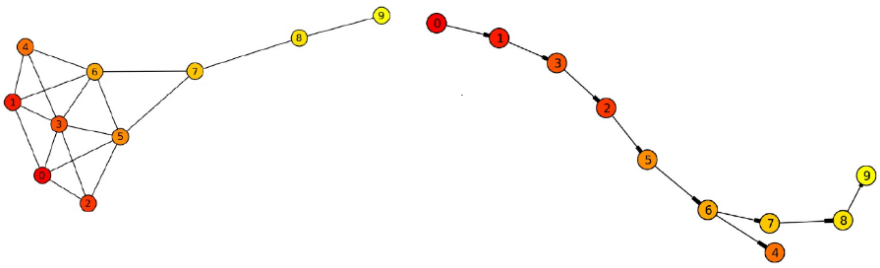
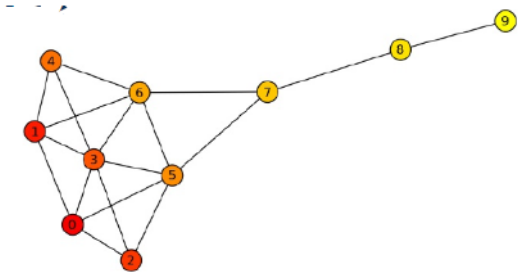


Figura 11 DFS

- Breadth First Search (BFS)
  - Similar al anterior, pero revisamos primero todos los vecinos del nodo antes de pasar al siguiente. Para ello nos ayudamos de una cola.



n'	Q	Visited
	{0}	{0}
0	{**1**,**2**,**3**,**5**}	{0,1,2,3,5}
1	{2,3,5,**4**,**6**}	{0,1,2,3,5,4,6}
2	{3,5,4,6}	{0,1,2,3,5,4,6}
3	{5,4,6}	{0,1,2,3,5,4,6}
5	{4,6,**7**}	{0,1,2,3,5,4,6,7}
4	{6,7}	{0,1,2,3,5,4,6,7}
6	{7}	{0,1,2,3,5,4,6,7}
7	{**8**}	{0,1,2,3,5,4,6,7,8}
8	{**9**}	{0,1,2,3,5,4,6,7,8,9}
9	{}	{0,1,2,3,5,4,6,7,8,9}

Figura 12 BFS

- Dijkstra:
  - Asignamos a todos los nodos una distancia tentativa. En primer lugar, marcamos el primer nodo
  - como una distancia de 0, y al resto como infinito. Marcamos todos los nodos como no visitados y les ponemos en la cola de no visitados.
  - Mientras que la cola NO esté vacía:
    - Sacamos de la cola aquel nodo que tenga una menor distancia tentativa y le marcamos como visitado.
    - Si ese nodo es el nodo de destino, paramos el algoritmo.
    - Para cada vecino que no haya sido visitado actualizamos su distancia tentativa, que será el menor valor entre la distancia tentativa actual y la suma de la distancia del nodo que hemos sacado y el peso de su enlace a este nodo.

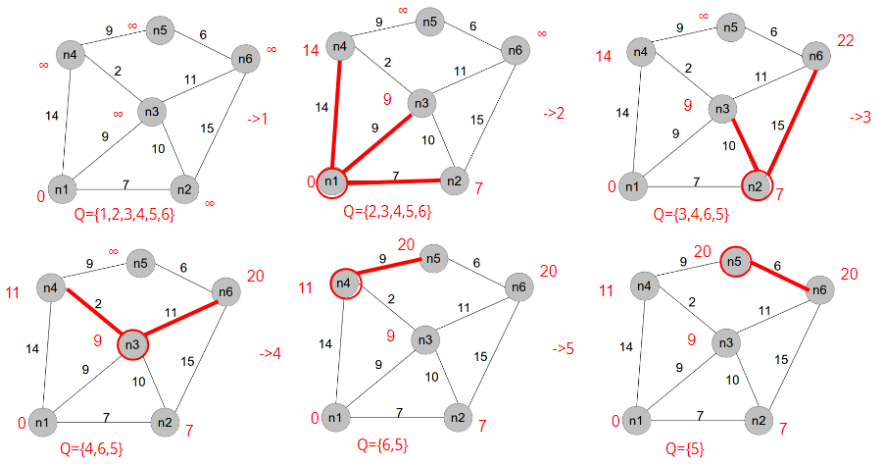


Figura 13 Dijkstra

Centralidad y centralización

¿Quién es más importante en una red?

Depende del criterio. Métricas:

- ¿Quién tiene más enlaces? -> Degree
- ¿Quién está más próximo a todos los demás? -> Closeness
- ¿Quién une grupos alejados de una red? -> Betweenness
- ¿Quién está conectado a los mejor conectados? -> Eigenvector, Katz, Pagerank, HubsAuthorities

Degree - Centralidad de grado

Un nodo es más importante cuanto mayor sea su grado (nº de enlaces).

Limitaciones:

- Depende exclusivamente del nº de enlaces, independientemente de la importancia de los nodos que una.
- Es una medida local, no depende del resto de la red.

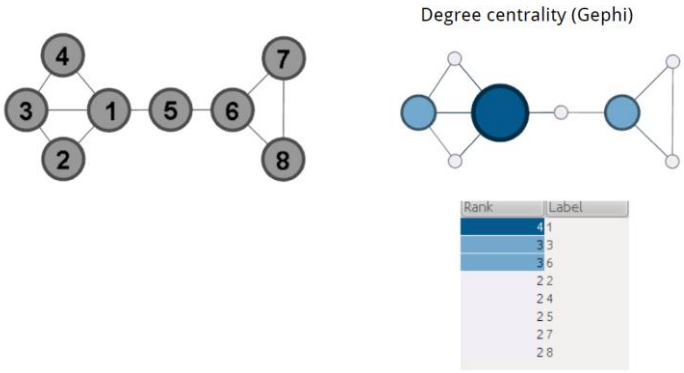


Figura 14 Centralidad de grado

Eigenvector centrality

La importancia de un nodo en la red crece si sus vínculos también son importantes.

Realizamos un sencillo algoritmo recursivo para calcular la importancia de los nodos:

- Damos una importancia de 1 a todos los nodos.
- Propagamos recursivamente la importancia a través de los enlaces.

Inconvenientes con redes dirigidas:

- La matriz de adyacencia es asimétrica. Si mi importancia depende de que otros nodos me apunten, se toma el valor derecho.
- Puede haber nodos con centralidad nula

Katz

Soluciona los inconvenientes de eigenvector, dando una importancia distinta a cada nodo.

PageRank

Defiende que un nodo no puede compartir su importancia sin desgaste, si no que esta también depende del número de nodos que le apuntan. Es el sistema que utiliza Google para el posicionamiento de las webs.

Problemas:

- Escasa visibilidad de nuevas páginas
- Manipulación externa intencionada
- Spamdexing

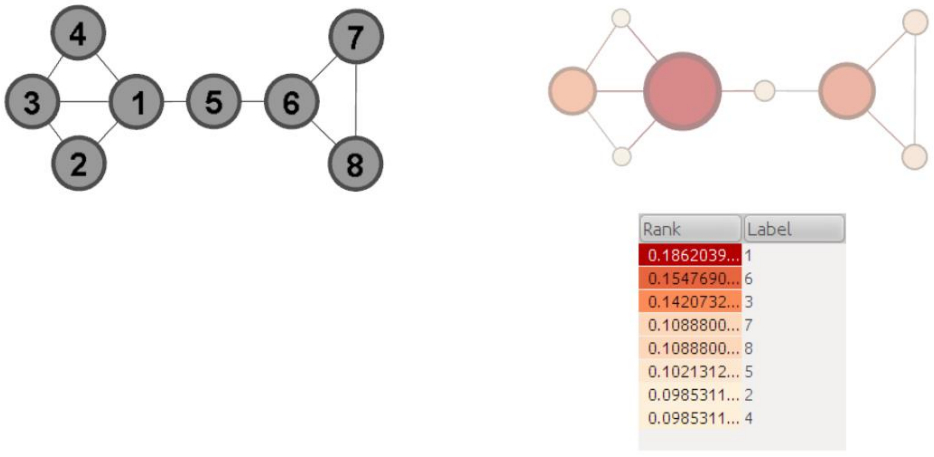


Figura 15 Centralidad de PageRank

Hubs-Authorities

Solo aplica a redes dirigidas. Distingue entre:

- Authority: nodos que son relevantes porque contienen información importante. Son los "apuntados".
- Hubs: nodos que nos dicen dónde están las mejores autoridades. Son "los que apuntan".

Los nodos pueden tener ambos roles simultaneamente.

Figura 16 Hubs-Authorities

Centralidad de cercanía

Los nodos más cercanos a los demás son más importantes. La medida de closeness de un nodo es la inversa de la distancia media al resto de nodos.

Inconvenientes:

- En redes smallworld no hay apenas diferencia entre la distancia media menor y la mayor, por lo que no discrimina muy bien este método de centralidad.
- Es indefinida en el caso en el que haya más de un único componente, ya que la distancia entre nodos de ambos será infinita.
    - Para solucionar este problema se utiliza una medida de closeness más elegante, la media armónica.

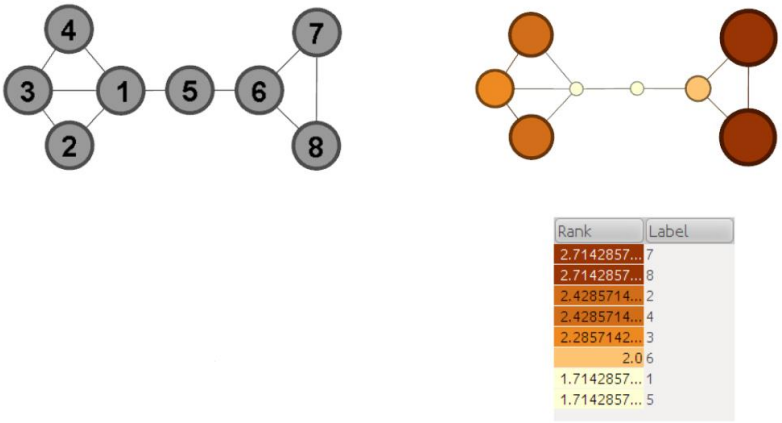
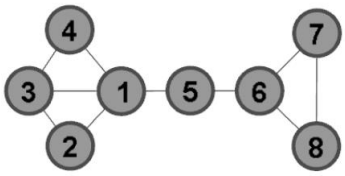
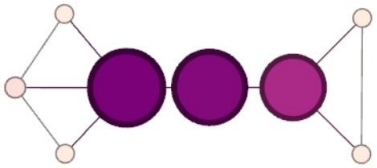


Figura 17 Closeness Centrality

Centralidad de intermediación

La importancia de un nodo depende de su grado de intermediación. Es el sumatorio de todos los caminos cortos que pasan por ese nodo sin ser este la cabeza o la cola del camino.





Lista de caminos cortos

Nodo 1	Nodo 2	Nodo 3	Nodo 4
{1,2}	{2,3}	{3,4}	{4,1,5}
{1,3}	**{2,3,4}, {2,1,4}**	{3,1,5}	{4,1,5,6}
{1,4}	{2,1,5}	{3,1,5,6}	{4,1,5,6,7}
{1,5}	{2,1,5,6}	{3,1,5,6,7}	{4,1,5,6,8}
{1,5,6}	{2,1,5,6,7}	{3,1,5,6,8}	
{1,5,6,7}	{2,1,5,6,8}		
{1,5,6,8}			
Nodo 5	**Nodo 6**	**Nodo 7**	**Nodo 8**
{5,6}	{6,7}	{7,8}	
{5,6,7}	{6,8}		
{5,6,8}			

Rank	Label
12.5	1
12.0	5
10.0	6
0.5	3
0.0	2
0.0	4
0.0	7
0.0	8

Figura 18 Centralidad de intermediación

Random-walk betweenness

La métrica anterior supone que los "mensajes" utilizan únicamente los caminos más cortos, pero pueden existir casos en los que los caminos largos sean relevantes. Por ello Newman propone utilizar a los caminantes aleatorios.

Estructura local

Grupos de vértices (nodos)

La mayoría de las redes se dividen en grupos o comunidades. Estos grupos se manifiestan con los enlaces que tienen los nodos de la red.

Cliques

Clique: subconjunto máximo de nodos de una red de tal manera que cada uno de ellos t iene un enlace con todos los demás.

Propiedades:

- Un nodo puede pertenecer a más de un clique.
- La existencia de cliques en redes poco densas indican la presencia de grupos interconectados.
- Una red en la cual todos los nodos están conectados entre sí se llama red completamente conectada. Identificar cliques requiere de mucho poder computacional.

k-plex

El requisito de que todos los miembros de un clique estén conectados con el resto es muy exigente, por lo que pueden existir grupos en la red pero que no cumplan esta condición.

Un k-plex de tamaño n es un subconjunto máximo de n vértices de una red de tal forma que cada uno está conectado con al menos n-k de los otros. Un nodo puede pertenecer a más de un k-plex. Es una métrica más proxima a grupos de redes sociales.

k-core

Es el subconjunto máximo de nodos tales que cada uno está conectado al menos a k otros del subconjunto. Dos conjuntos k-core no tienen porqué tener el mismo número de nodos. No pueden superponerse, un nodo no puede pertenecer a 2 o más k-cores. Son muy fáciles de computar.

k-clique

Es un subconjutno máximo de nodos tales que cada uno está a una distancia no mayor que k del resto. Un k-clan o k-club es un k-clique en el que los caminos que unen a sus miembros pertenecen al mismo subconjunto.

Resumen de subgrupos

Subgrupo	Máximo subconjunto de vértices en los que todos están...
clique	Conectados con todos
k-plex	Conectados con al menos (n-k)/n fracción de todos
k-core	Conectados con al menos k del resto
k-clique	A una distancia no mayor de k
k-clan	… y además los caminos pertenecen al subconjunto

Figura 19 Resumen subgrupos

Componentes

Ya habíamos visto esta definición antes, se trata de una subred en la cual cada nodo está conectado todos los demás mediante un camino y no tiene enlaces a otros nodos fuera del componente.

k-componente

Máximo subconjunto de nodos de una red de tal forma que todos los nodos están conectados con los demás por al menos k caminos independientes.

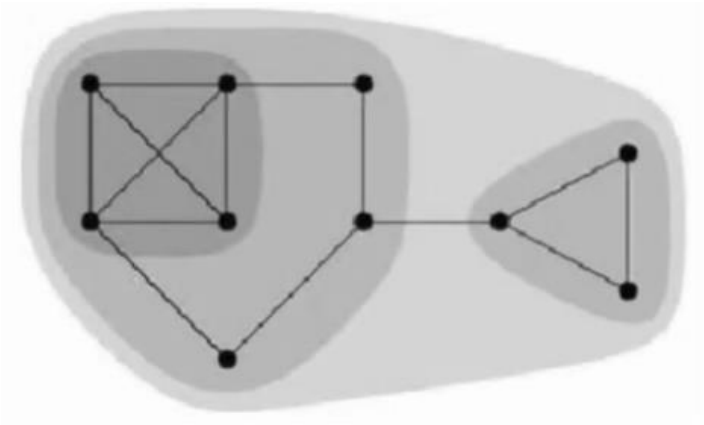


Figura 20 k-componente

Transitividad

Una relación es transitiva si A->B, B->C, entonces A->C. En redes este concepto es sinónimo al de relaciones trianguraes y loops de tamaño 3.

El coeficiente de clustering es la probabilidad de que dos nodos con un vecino común tengan a la vez un enlace.

C = (Nº de triángulos * 3) / (Nº de tripletes conectados)

Coeficiente de clustering de un nodo:

C = nº de pares de vecinos del nodo que están conectados entre ellos / nº total de pares de vecinos del nodo

En redes sociales, un nodo con poco coeficiente de clustering representa un agujero estructural, ya que la falta de vínculos entre sus vecinos reduce los caminos alternativos de información y les hace más dependientes de él. Es como una medida de centralidad inversa.

En ocasiones se utiliza el valor promedio del coeficiente de clustering individual como medida del clustering de la red.

Reciprocidad

En redes dirigidas. Frecuencia de loops de tamaño 2. Probabilidad de que un nodo señale a otro y este a su vez le señale a él.

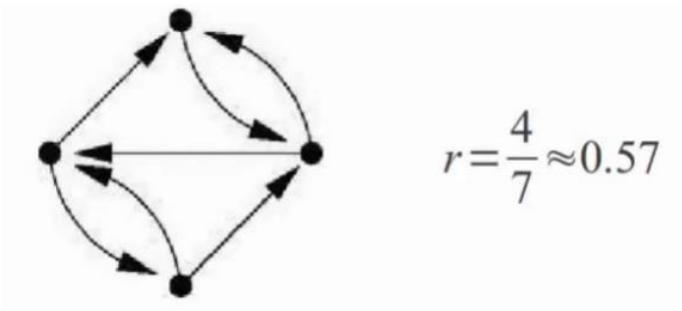


Figura 21 Reciprocidad

Similitud

Determina si dos nodos son similares. Dos clases:

- Equivalencia estructural: Dos nodos son similares en la medida en la que compartan vecinos en la red. Nº de vecinos en común.
- Equivalencia regular: dos nodos son similares en la medida en que sus vecinos compartan algún rasgo o similitud. Compleja de medir.

Homofilia y heterofilia

En el patrón de la homofilia (asortative mixing) los nodos tienden a tener más enlaces con aquellos otros nodos que consideren más iguales. En el patrón de heterofilia ocurre lo contrario.

Detección de comunidades

Estructuras locales

- Comunidades: grupos de nodos densamente conectados entre sí y con pocas conexiones entre grupos.
- Classes: Grupos de nodos con un patrón similar de conexiones a otros nodos.

Comunidades

¿Por qué son importantes?

La mayoría de las redes presentan una estructura modular (dividida en comunidades). Cuando existe estructura modular las propiedades globales medidas no son apropiadas para describir la estructura de las redes, pues estas no son homogéneas.

¿Para qué es útil?

- Para describir la estructura de la red
- Para identificar la estructura de cara a un objetivo
- Para crear nuevo conocimiento

Detección de comunidades

Varios métodos:

- Métodos aglomerativos
- Métodos de eliminación de enlaces
- Métodos basados en maximizar la modularizad
- Otros métodos: análisis esprectal.

Algoritmo de Girvman-Newman

Método iterativo por el cual vamos a ir eliminando los elementos de mayor betweenness.

Hasta que haya n comunidades cada una de un nodo:

- Calculamos el betweenness
- Eliminamos el enlace del nodo de mayor betweenness.
- Vemos si creamos una comunidad. Si es así, repetimos el proceso para cada comunidad.

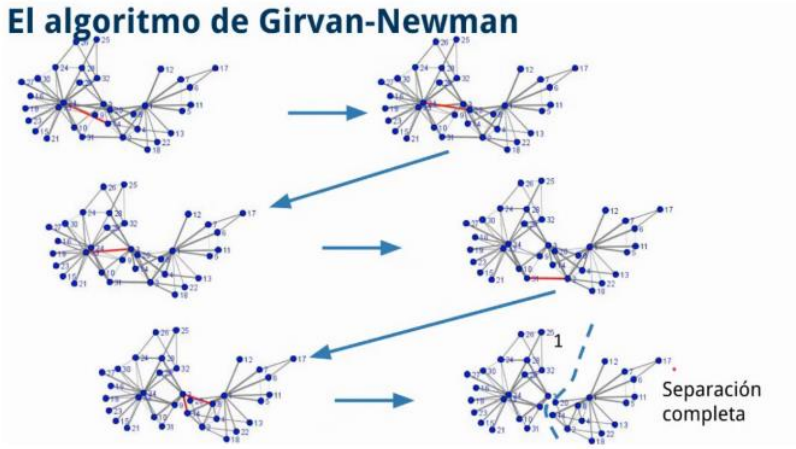


Figura 22 Algoritmo de Girvman-Newman

Modularidad

Es la fracción de enlaces que caen dentro de los grupos menos la fracción esperada si los enlaces hubiesen sido distribuidos al azar (manteniendo el grado de cada nodo).

Modelos de redes aleatorios

Es interesante realizar estudios de los modelos de redes en vez de redes concretas porque las conclusiones que saquemos del análisis de las redes concretas serán útiles sólo en esa red, y no en el resto.

Modelos estocásticos de redes: Abstracciones más simples de un tipo de red que reproducen alguna de las propiedades de las redes más complejas, bien para:

- Tener un modelo nulo con el que poder comparar las propiedades de una red compleja real.

- Ser candidato para estudiar la dinámica de ciertos fenómenos en las redes.
- Para disponer de representaciones más simples de las redes.
- Para explicar sus propiedades de forma matemática. Un modelo estocástico de red se encarga de definir un proceso aleatorio de formación de enlaces y nodos.

Modelo de Erdös-Rényi

En el modelo de red aleatoria más general fijamos el nº de nodos (N) y el nº de enlaces (M) y repartimos los enlaces de forma aleatoria (Modelo de Erdös-Rényi). El modelo de Gilbert es un modelo equivalente, ya que fija el nº de nodos (N) y la probabilidad de que dos nodos cualesquiera tengan un enlace (p).

Distribución de grado

Establecer un enlace con otro individuo es un evento independiente de probabilidad p, luego la probabilidad de que un individuo tenga k enlaces sigue una binomial.

$$p(k) = \binom{n-1}{k} p^k (1-p)^{n-k-1}$$

Figura 23 Distribición de grado

Las redes aleatorias no presentan hubs, ya que la mayoría de los nodos presentan un grado parecido. La distribución de grado de una red aleatoria de Erdös-Rényi no captura las distribuciones de las redes reales.

Componente gigante

Si variamos p desde 0 hasta 1 podemos ver que aparece una transición de fase correspondiente con la aparición de un componente gigante, cuyo tamaño crece en proporción a n.

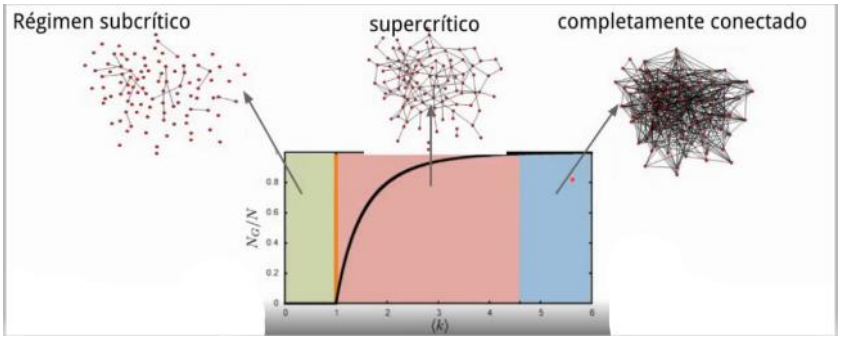


Figura 24 Componente gigante

Small World

En redes aleatorias la firma estadística de la propiedad smart world es que la distancia media escala con el logaritmo de n.

Clustering

En redes aleatorias, el clustering de un nodo es independiente de su grado. Si k es fijo, el clustering decrece con 1 / n. Las redes aleatorias de Erdös Rényi no capturan las propiedades de clustering de muchas redes reales.

D. Técnicas y Herramientas

En el siguiente apartado se describirán todas las herramientas que han sido utilizadas durante todo el proyecto.

D.1 Metodología ágil – SCRUM

La metodología SCRUM es el proceso por el cual se realizarán un conjunto de tareas periódicamente, cuyo objetivo principal será fomentar el trabajo en equipo.

Este método tiene como meta el alcanzar el resultado óptimo de un proyecto. Las practicas aplicadas con esta metodología se retroalimentan unas con otras y su integración tiene su origen en un estudio de cómo se debe coordinar a los equipos para poder ser competitivos.

El SCRUM se basa en realizar entregas parciales del proyecto final, estas entregas se realizarán de manera prioritaria, es decir se realizarán primero las tareas que mayor beneficio aporten. Por lo tanto, SCRUM es indicada para los proyectos que son más complejos, con requisitos que van cambiando y en los que la flexibilidad y la innovación cobran mucha importancia.

D.2 Herramienta para el control de versiones

-**Herramienta utilizada:** GitHub

GitHub es una plataforma de desarrollo colaborativo que sirve para alojar proyectos utilizando el sistema de control de versiones Git. Esta herramienta tiene como uso principal la creación de código fuente para programas de ordenador.

El motivo por el que he elegido esta herramienta es porque la he estado usando a lo largo de toda la carrera, por lo que me es más fácil el utilizarla.

D.3 Herramienta para la gestión del proyecto

-**Herramienta utilizada:** ZenHub

Utilizaremos esta herramienta debido a que está integrado en GitHub y que lo hemos utilizado a lo largo del grado de Informática, por lo que la mejor opción frente a las demás herramientas es ZenHub.

D.4 Herramienta para realización de la documentación

-**Herramienta utilizada:** Microsoft Word

Microsoft Word fue la herramienta elegida frente a LaTeX debido a la comodidad que nos presenta y la facilidad de uso.

D.5 Herramienta para gestionar las referencias bibliográficas

-**Herramienta utilizada:** Zotero

Zotero fue la herramienta escogida debido a su versatilidad y al tener una gran capacidad de compatibilidad con Microsoft Word que será la herramienta que utilizaremos para la documentación.

D.6 Lenguaje de programación

-**Herramienta utilizada:** Python

Se ha utilizado Python como lenguaje de programación debido a su manejo a lo largo de la carrera y a que el código del proyecto anterior también estaba en Python, por lo que cambiar el código a otro lenguaje podría ocasionar muchos fallos, además de ser muy costos. Además, Python tiene librerías como DyNetX destinadas a la generación de red dinámicas.

D.7 Generación de la red dinámica.

-**Herramienta utilizada:** DyNetX

DyNetX es una librería de Python creada para generar redes dinámicas. Creemos que es la mejor herramienta, porque además de crear la red dinámica, es muy fácil de usar, ya que para crear la red lo único que pedirá serán los nodos que crean el enlace y el intervalo de tiempo.

D.8 Interfaz gráfica.

-**Herramienta utilizada:** Flask

Como interfaz gráfica hemos utilizado Flask debido a su fácil uso y su rapidez.

E. Aspectos relevantes de desarrollo del proyecto

En esta sección hablaremos de las tomas de decisiones que hemos debido de tomar a lo largo del proyecto.

E.1 Inicio del proyecto

La idea principal por lo que decidí escoger el proyecto es que me gustan mucho las películas, y que el proyecto se basará en crear una red dinámica de cualquier película o novela donde se irá visualizando como los personajes van interaccionando a lo largo de los capítulos (novelas) o escenas (películas) era bastante interesante.

Por lo que me puse en contacto con mi tutor para ver si podía tener la opción de continuar el proyecto de mis compañeros Luis Miguel Cabrejas Arce y Jorge Navarro González. Mi tutor me explico la dificultad que tenía dicho proyecto ya que debería estudiarme en poco tiempo una asignatura que todavía no había cursado como era Nuevas Tecnologías y que el proyecto venía de dos trabajos anteriores, por lo que debería de entender muy bien su código.

A pesar de las desventajas decidí seguir con el proyecto, y una vez el tutor aprobó que realizara dicho proyecto, comencé con el desarrollo de la aplicación.

E.2 Metodologías

La metodología seguida fue SCRUM. Debido a que solo era una persona realizando el proyecto, hay algunas diferencias con la metodología habitual. A pesar de ello, se han intentado cumplir las características de dicha metodología:

- Diseñar el proyecto de una manera incremental.
- Reuniones con el tutor al comenzar cada sprint para conocer las tareas que debía implementar.
- Una vez se sabían las nuevas tareas se estimaban que duración iban a tener.
- Reuniones al finalizar los sprints para explicar al tutor las nuevas funcionalidades del proyecto.
- Mediante ZenHub se han ido finalizando las tareas según se iban completando para saber cuántas tareas quedaban por hacer.
- Tiempo para cada sprint de 2 a 3 semanas aproximadamente.

E.3 Desarrollo de los algoritmos

Para el desarrollo de los algoritmos lo primero que tuve que hacer es crear un prototipo que creará de forma correcta la red dinámica. Como lo estaba haciendo a parte del código del proyecto, me tuve que enterar de como funcionaba las creaciones de diccionario tanto de películas como de ePub, para que una vez que llevará el prototipo al código principal no me diera fallos de relación. Una vez terminado el prototipo se crearon los siguientes algoritmos:

- Creación de la red dinámica.

  Para la creación de la red dinámica lo primero que se debía saber es si el diccionario venía de un guion de película o de una novela, ya que estos tienen sus diferencias a la hora de crear de la red.

  Esto se solucionó al añadir una variable que nos dijera si el diccionario venia de una novela o no. Si al ejecutar la aplicación el usuario va a la ventana ParámetrosEpub, es que el diccionario es de una novela, por lo que la variable epub será verdadera, si por el contrario iba a la ventana Parámetros, es que el diccionario es un guion, por lo que la variable epub será falsa.

  Una vez sabemos que diccionario escoger tenemos dos opciones para crear la red dinámica. Si el diccionario es un guion hacemos los siguientes pasos:

  - Recorremos el guion para crear una lista que contenga el id del enlace, los dos nodos que se relacionan, el intervalo de tiempo donde se da hecho el enlace (la escena) y el peso del enlace que será 1.
  - Esa lista la ordenaremos para que vaya en orden según el intervalo de tiempo.
  - Con esa lista ordenada crearemos la red dinámica. Según el intervalo de tiempo que nos envíe el usuario se hará un bucle for que añada todos los enlaces hasta ese intervalo de tiempo. Se tendrá en cuenta el número de apariciones a la hora de añadir un enlace si uno de los nodos no cumple el número de apariciones. Si un nodo no tiene ningún enlace, pero sí que cumple el número de apariciones se comprobará si el momento en el que aparece está dentro del intervalo de tiempo, y si lo cumple, el nodo aparecerá en la red dinámica.

  Si por el contrario el diccionario es una novela hacemos los siguientes pasos:

  a. Recorremos la novela para crear una lista que contenga el id del enlace, los dos nodos que se relacionan, el intervalo de tiempo donde se da hecho el enlace (el capítulo) y el peso del enlace que será 1.
  b. Esa lista la ordenaremos para que vaya en orden según el intervalo de tiempo.
  c. Con esa lista ordenada crearemos la red dinámica. Según el intervalo de tiempo que nos envíe el usuario se hará un bucle for que añada todos los enlaces hasta ese intervalo de tiempo. Se tendrá en cuenta el número de apariciones a la hora de añadir un enlace si uno de los nodos no cumple el número de apariciones. Si un nodo no tiene ningún enlace, pero sí que cumple el número de apariciones se comprobará si el momento en el que aparece está dentro del intervalo de tiempo, y si lo cumple, el nodo aparecerá en la red dinámica.

- Exportación de la red dinámica.

  Lo que se deberá hacer es a partir de la red generada por la creación de la red dinámica, transformamos dicha red en una que se pueda leer en formato gexf.
- Animación red dinámica.

  También a partir de la red generado por la creación de la red dinámica y con la ayuda de la librería matplotlib.animation convertiros nuestra red dinámica en una animación.
- Informe dinámico.

  El informe dinámico estudiará la red dinámica en cada uno de sus intervalos. Dentro de las métricas tendremos dos tipos: métricas de red y métricas de nodo.

  Métricas de red: son métricas que solo modifican el valor de la red según el intervalo.

  Métricas de nodo: son métricas donde todos los nodos de la red se modificarán el valor según el intervalo.

E.4 Problemas derivados del código.

Solución a fallos previos de NetExtractor, los cuales eran que en las películas no se mostraban de forma correcta los nodos según el número mínimo de apariciones, y en las novelas si un nodo no tenía ningún enlace no aparecía en la red.

F. Trabajos relacionados

En los últimos años cada vez son más los proyectos dedicados a la ciencia de redes, estos pueden ser muy diversos, ya que te puedes encontrar desde casos reales como pueden ser estudios científicos hasta casos más ficticios como pueden ser películas.

Algunos ejemplos de trabajos relacionados son:

F.1 NetExtractor

Como este proyecto es el predecesor al mío, deberá de aparecer el primero.

NetExtractor consiste en la obtención, visualización y análisis de una red que se crea mediante un guion de una película o una novela de ePub.

La obtención de los personajes mediante un guion de película será mediante la lectura de un guion de la página imsdb. En ella se leerá la página en html, y se considerará personaje a aquel que este dentro de una escena y aparezca en negrita. Por la parte de novelas la aplicación leerá el ePub con un analizador léxico y las palabras que estén en mayúsculas se las considerará personajes.

En cuanto a las interacciones entre personajes, si es un guion, si dos personajes están en una misma escena se creará un enlace. Mientras que si es una novela habrá una interacción si dos personajes están dentro del rango de palabras. Este rango lo elegirá el usuario. Además de este rango el usuario podrá elegir si se tienen en cuenta los capítulos o no, si no se tienen en cuenta, el rango de palabras se puede llevar a otros capítulos anteriores o posteriores, si se tienen en cuenta los capítulos, los personajes solo podrán tener interacción con los personajes del mismo capítulo.

Por último, el usuario podrá visualizar la red gracias a network styling with d3, esta herramienta puede modificar parámetros de la red como puede ser el tamaño de los nodos, el tamaño de los enlaces, la distancia entre nodos, etc. Para finalizar el usuario podrá ver un informe con todas las métricas que quiera.

F.2 Network of Thrones

Un proyecto muy parecido al que estamos realizando, creado por Jie Shan y Andrew Beveridge. Este trabajo llevo a cabo los siguientes pasos para poder generar la red.

- La primero fue parsear todos los libros buscando palabras que estuvieran en mayúsculas. Además de emplear web scraping sobre una wiki que estuviera especializada en Juego de Tronos, para así poder extraer todos los personajes.
- Lo siguiente que hicieron fue eliminar todo tipo de ambigüedades. Dos personajes con el mismo nombre, pero diferente apellido generaría ambigüedad si solo aparece el nombre en el texto. Como solución, reescribieron el libro para quitar todas las ambigüedades posibles. Si se encontraban con una ambigüedad, esta se sustituirá por un identificador que tengan ellos definido.
- Para la generación de enlaces, intentaron detectar las interacciones por la cercanía de palabras. Las relaciones detectadas son: que dos personajes estén en el mismo lugar, que dos personajes tengan una conversación,

que un personaje hable sobre otro, que un personaje escuche hablar a otro personaje sobre otro personaje, que un personaje hable sobre dos o más personajes, etc.

- Finalmente se extraerá la red donde la distancia máxima entre los personajes para que tengan relación sea de 15 palabras.

G. Conclusiones y líneas de trabajo futuras

En esta sección se recogerán las conclusiones que nos ha dejado el proyecto y las futuras líneas de trabajo

G.1 Conclusiones

Creo que se han cumplido los objetivos marcados al inicio del proyecto. Se ha creado una red dinámica tanto para películas como para novelas. En dicha red dinámica el usuario es capaz de viajar a través de los intervalos de tiempo y ver como se ha ido modificando la red de personajes hasta el final de la película o novela. Además, se puede descargar la red dinámica en formato gexf, para que pueda ser leído por otras herramientas como gephi. También se proporciona una animación de la red dinámica. Por otro lado, también tenemos un informe dinámico, en donde el usuario puede ver como van cambiando las métricas del informe en cada uno de los intervalos.

En cuanto a los objetivos personales, los considero cumplidos, ya que he obtenido mucho más manejo y conocimiento con algunas herramientas con las que no había trabajado. Además, he mejorado bastante con el lenguaje de programación Python y tenemos una idea más desarrollada sobre para que sirve la metodología SCRUM.

La aplicación ahora dota de una red dinámica que se puede personalizar al gusto del usuario, además de poder verse como era la red en cualquier intervalo de tiempo. A su vez, se puede descargar la animación de la red dinámica, es decir, una animación donde vemos a la red en los diferentes intervalos de tiempo y poder ver como se ha ido modificando hasta llegar a la red que es, y también se puede descargar la red para que se pueda investigar en otras herramientas de redes complejas como puede ser gephi. Por último, también tenemos el informe dinámico, que nos expone como va cambiando la red y los nodos de esa red en las diferentes métricas de la ciencia de las redes.

G.2 Líneas de trabajo futuras

Detección de personajes

Implementar un mejor detector de personajes tanto para guiones como para novelas cuando se crea el diccionario de manera automática. Seria importante para reducir los falsos positivos que puedan aparecer. Esto se puede solucionar con herramientas que tiene la aplicación como borrar personajes, para borrar del diccionario los que se consideren que no son personajes, y juntar personajes, para juntar personajes que son la misma persona, pero escrito de diferente forma. Pero la forma óptima sería que el usuario no se tendría que preocupar por borrar o juntar a personajes.

Internacionalización

De momento solo se puede ver la web en inglés y español, si queremos que la aplicación llegue a más gente y pueda ser leída por más podría ser interesante dotar a la aplicación de más idiomas para ser entendida por más gente.

Bibliografía