Prediccion de enlaces en redes sociales:

Concepto y experimentacion basado en Algoritmo bio-inspirado

Diego Lozano Mejia
Universidad Peruana de ciencias Aplicadas
Monterrico, Peru
u201413062@upc.edu.pe

Gabriel E Ramirez Reategui
Universidad Peruana de ciencias Aplicadas
Monterrico, Peru
g_e_ramirez@outlook.com

ABSTRACT

Uno de los campos más estudiados en las ciencias de la computación, es los algoritmos bio-inspirados, presentes desde los primeros años de la ciencia, estos han aparecido como intentos de utilizar la naturaleza para poder implementar soluciones en los programas. Lamentablemente, durante mucho tiempo, el costo de procesamiento requerida era demasiado alta para ser implementado más alla de presentaciones teóricas. Por otro lado, como se mencionó anteriormente,uno de los campos más importantes en la actualidad del momento son las redes sociales. No sólo revisado por matemáticos o científicos de la computación, sino otras áreas que colaboran como puede ser la sociología, biología, entre otros. Un aspecto sumamente importante es la predicción de aristas, y es algo que se tomó en consideración en el primer proyecto. Este Artículo incluye una recapitulación los conceptos básicos de las redes, incluyendo conceptos de predicción de aristas, entre otros. Pero también, este incluirá la implementación de una red multi-capa(MLP) con la intención de mejorar nuestro algoritmo, demostrando los pro y contra de la implementación del mismo.para este testeo se utilizará un conjunto de los de los algoritmos más comunes, Common Neighbors, Adamic-Adar y Katz, comparando su performance al predecir nuevas conecciones por medio de la velocidad de procesamiento, Área bajo la curva y precisión. Como resultado se obtuvo que, la precisión mejora significativamente, pero el tiempo de procesamiento también.

KEYWORDS

Machine Learning, Redes Complejas, Redes Sociales, MLP

ACM Reference Format:
Diego Lozano Mejia and Gabriel E Ramirez Reategui. . Prediccion de enlaces en redes sociales:: Concepto y experimentacion basado en Algoritmo bio-inspirado. In *Proceedings of UPC Machine Learning (UPC'18).* ACM, New York, NY, USA, 6 pages.

1 RESUMEN

El proyecto busca dar una introducción al área de las redes complejas, para ser más exacto, dentro de las Redes Sociales, las cuales permiten entender muchas interacciones entre personas. Además de dar una explicación sobre el funcionamiento de los algoritmos bio inspirados y como este se puede aplicar en nuestro proyecto
Para esto, se dará una breve recapitulación a los conceptos de redes, redes complejas y sus propiedades, además una explicación sobre las redes sociales y su uso dentro de diversas áreas de estudio, por otro lado, se dará una explicación de los algoritmos bio inspirados, así como la teoria e implementacion del algoritmo de perceptrón multicapa con el cual el proyecto 1 fue mejorado. Para poder lograr esto, en segundo lugar, se realizará un experimento usando una base de datos relacionada a Facebook, minimizando el algoritmo a un 15% para poder mostrar resultados concluyentes comparando los posibles resultados con los algoritmos previos. Para un resultado óptimo, se correrá el algoritmo 10 veces y sacando los datos de tiempo de ejecucion, precision y Área bajo la curva, se nos permitirá llegar a una conclusión de cuál conviene usar en determinadas situaciones.

El dataset elegido para este proceso es un dataset que cuenta con mas de 88000 conexiones de entre mas de 4000 nodos. Este data será preprocesado de manera sencilla para eliminar el ruido que pueda aparecer. Ademas sera reducida a un total de 11000 nodos para poder ejecutarlo de manera mas rapida
Para la ejecución del algoritmo, se opto por el uso del lenguaje python y las librerias pandas y networkx ya que permite un manejo rápido de data orienta a redes complejas e inclusive incluye funciones ya diseñadas para este fin. Las versiones utilizada para este proceso es el de Python 3.6.

2 INTRODUCCIÓN

En el campo de las ciencias de la computación, desde siempre ha existido un gran interés en simular el comportamiento de la naturaleza en nuestros algoritmos, algoritmos como el genético, las redes neuronales o los autómatas celulares . Uno de los principales estudios se dan sobre los conocidos como redes neuronales, un modelo de predicción que simula la manera que las neuronas trabajan. Gracias a las tecnologías actuales, es mucho más efectivo el uso de diferentes algoritmos bio-inspirados en áreas tan complejas como las redes sociales. Este uso permite una mejora en el modelo de predicción previamente implementado.
Es por eso, que es importante estudiar sobre las posibles soluciones y modelos con la intención de mejorar los modelos probabilísticos de predicción en las redes sociales. Para esto se ha decidido la implementación de un algoritmo multicapa perceptrón que se explicara con más detalle a continuación.

3 CONCEPTOS

Para poder comprender el proyecto, es importante recapitularcómo las redes funcionan y algunos de sus conceptos fundamentales. En las siguientes secciones se explicarán los conceptos básicos y definiciones de los temas a tratar utilizando los conceptos presentados durante el proyecto 1. En primer lugar, se verán los conceptos las redes, redes complejas, como algunas de sus propiedades básicas. Luego,se verán los conceptos sobre la predicción de enlace, el

UPC'18, Junio 2018, Monterrico, Lima Peru
.

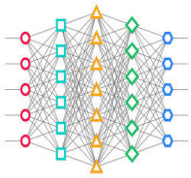

Figure 1: Ejemplo de red neuronal

porqué de su importancia y los algoritmos que fueron usados en la experimentación.
En segundo lugar, se hablara del funcionamiento y teoria tras los perceptron multi-capa y sobre algunos conceptos de otros algoritmos bio-inspirados.

3.1 Redes

La definicion de una red es un set de datos, a los cuales se llamara nodos, conectados por aristas, estas redes se pueden encuentrar en el dia a dia y se puede observar varios ejemplos tanto físicos como conceptuales, varios de estos ejemplos pueden ser: World Wide Web, redes sociales, redes laborales, redes neuronales. Las Redes se encuentran dentro de la teoría de grafos, la cual es uno de los pilares de la matemática discreta. Pero las redes no se limitan a ser parte de la matematica, estas son estudiadas por otras ciencias, como la sociologia.
Las Redes han visto en los últimos años un aumento de importancia y en su estudio, en especial las redes de gran tamaño, gracias a su potencial estadístico y al factor determinante que en la actualidad, gracias a Big Data y Machine Learning, se nos da la posibilidad de procesarlas y obtener bastante información de las mismas.
Para poder entender mejor el articulo, se ha agregado un glosario

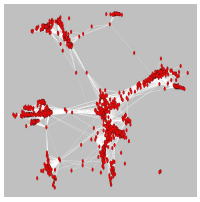

Figure 2: Muestra de la Red visualizada

de algunos términos pertenecientes a las partes de un grafo. Pese a que el grafo tiene una gran cantidad de propiedades y data en el mismo, se ha decidido por mencionar solo los mas óptimos:

- Vértice o nodo: Es la unidad fundamental del grafo.
- Arista: Encargada de conectar vértices, puede significar conexión únicamente o puede poseer otros atributos adicionales. En física es conocida como enlace, en otras áreas puede ser conocido como link.
- Dirigida/No Dirigida: Concepto sobre las aristas, encargado de determinar si existe una dirección establecida en la arista, en caso no existir, se considera que a,b es igual a b,a.
- Grado: Número de aristas conectados a un vértice.
- Geodesic path: El camino más corto entre un nodo y otro, este puede ser único como existir más de un camino.
- Diámetro: Este se obtiene calculando la longitud del camino geodésico más largo.

Existen diversos tipos de redes, explicados a continuación:

- Red Simple: La red más simple, nodos conectados por aristas.
- Red Compleja: Esta red puede tener diversos tipos de nodos, entre ellos pueden llevar pesos. Asimismo, las aristas pueden ser dirigidas o tener valores dentro del mismo. Uno de los ejemplos más estudiados en la actualidad, y al cual se examinara a fondo durante este trabajo es el de la Red Social.
- Hiperarista: El concepto más simple una red con hiper-arista es el de dos nodos en una arista.

Las redes complejas se pueden ver a lo largo de distintas áreas de investigación y desarrollo, dando base a muchas investigaciones actuales, algunas de las múltiples redes complejas son las: Redes sociales, Redes de información, Redes Tecnológicas y Redes biológicas. Todas fundamentadas en la misma teoría de grafos.
El enfoque de este trabajo dentro de las Redes Sociales, no hablamos de las plataformas web que existen como Facebook, sino, sobre la teoría detrás de la interacción entre diversas personas.
Para una definición más directa, una red social tiene como elementos un grupo de personas(o un grupo de grupos) con algun patron o interacción en común.

3.2 Predicción de nodos

La predicción de enlaces es el problema de detectar las conexiones posibles de los nodos. Existen muchos ejemplos de esta tarea, entre ellos, uno de los más comunes es la predicción de nuevas amistades. En lugares como facebook, este se puede aplicar de manera muy efectiva y poder luego recomendar nuevos amigos.
Esta tarea puede ser empleada de distintas maneras, y en este caso se han tomado en consideración dos aproximaciones diferentes:
Método en vecinos:
Dentro de este método se encuentran los algoritmos de Common Neighbors y de Adamic Adar, este método está basado en el encotrar nodos adyacentes que comparten información entre sí, lo que permite determina que, si los nodos comparte un nodo en común, este es posiblemente una nueva conexión.

3.2.1 Common Neighbors. Este método es considerado el método más directo, haciendo la codificación de este algoritmo mucho más fácil de implementar que otros métodos. El método de Common Neighbors se basa en, si ambos comparten un nodo en común, es probable que esos dos nodos sean vecinos igualmente.

Figure 3: Codigo de AUC

3.2.2 Adamic-Adar. Adamic Adar asocia versiones basados en las características comunes que tengan, busca características comunes, el algoritmo funciona en base a pesos, lo que permite que se dé prioridad a determinadas características.

Método basado en datos del camino:

Dentro de este método se encuentran los algoritmos de SimRank y Katz, estos estan basados principalmente en la distancia de los caminos y busca el camino más corto.

3.2.3 Katz. Katz:El algoritmo de Katz propone que a mayor cantidad de caminos entre X e Y, y cuánto más cortas sean, más fuerte será la conexión entre ambos.

Katz toma el tamanio de cada uno de los caminos desde x a y. Estos valores, seran multiplicados por un valor beta, que mientras mas pequeno sea el camino, tomará un valor más grande. Esto permite que los caminos con menor recorrido tengan más relevancia.

$$\text{score}(x,y) := \sum_{\ell=1}^{\infty} \beta^{\ell} \cdot |\text{paths}_{x,y}^{\langle \ell \rangle}|,$$

Figure 4: Formula de katz

3.3 Algoritmos bio-inspirados

Desde sus inicios, la computación se ha inspirado en la naturaleza como fundamento para muchos algoritmos y soluciones.

En un inicio, la capacidad de procesamiento complicaba la creación de estos algoritmos, pues requerían un poder de procesamiento diferente a la programación clásica.

Son los algoritmos que se basan en cierto comportamiento de la naturaleza, sea animales , células, entre otros. También conocido como Computación inspirada por la biología, esta es un conjunto de áreas como pueden ser:

- Inteligencia artifial.
- Computación Evolutiva.
- Bio Robotica.
- Vda artificial.

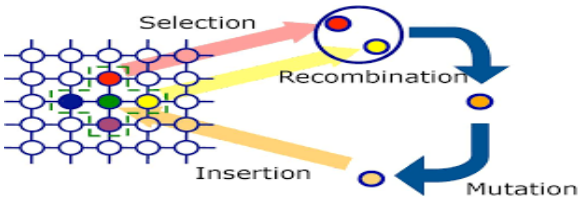

Figure 5: Ejemplo de algoritmo genetico

Para nuestro proyecto se tomo en consideracion dos posibles algoritmos bio-inspirados, los cuales se explicaran a continuacion:

- Algoritmo genetico.
- Red Neuronal.

3.3.1 Algoritmo genetico. La base del algoritmo genético es la implementación de mutaciones dado la mezcla entre "padres" para la creación de nuevos ejemplares. Simulando la evolución biológica de los organismos naturales.

Una de sus ventajas es que s óptimo como método adaptativo para resolver problemas de búsqueda y optimización.Lamentablemente, es un algoritmo costoso en cuestión de procesamiento.

Nota: La razon por la cual no se implemento este algoritmo es por su dificultad y costo de procesamiento a la hora de acercarse a resultados ideales, al tener katz y otros algoritmos con predicciones cercanas al 97% no era un algoritmo optimo para el proyecto.

3.3.2 Perceptron Multi-capa. El algoritmo perceptron multicapa, llamado MLP por sus siglas en ingles, es un algoritmo bio-inspirado basado en las redes neuronales, en este caso, busca solucionar problemas no lineales gracias al uso de multiples capas y neuronas para el entrenamiento. El aprendizaje que se suele usar en este tipo de redes recibe el nombre de retropropagacion del error (backpropagation)

Es importante mencionar que el algoritmo MLP es un algoritmo supervisado, pues requiere de una clase para poder trabajar, es por eso que en nuestro codigo fue necesario la implementacion de una modificacion de supervisado.

El MLP tiene distintas capas, las cuales se han separado en:

Capa de entrada: Constituida por aquellas neuronas que introducen los patrones de entrada en la red. En estas neuronas no se produce procesamiento.

Capas ocultas: Formada por aquellas neuronas cuyas entradas provienen de capas anteriores y cuyas salidas pasan a neuronas de capas posteriores.

Capa de salida: Neuronas cuyos valores de salida se corresponden con las salidas de toda la red.

Nota: Se ha elegido este algoritmo gracias a su rapida implementacion brindada por python y la libreia scikit-learn.

4 EXPERIMENTACIÓN

Una vez comprendido las bases de las redes y la predicción de links, este proyecto busca aplicar la teoría utilizando los algoritmos previamente aplicados en el proyecto 1(Common Neighbors,

Adamic-Adar,Katz) y luego el MLP. Para este procedimiento, se ha decidido usar Python, en su versión 3.6, y las librerías pandas(para el manejo de data frames) y networkX(manejo de redes), ademas, a diferencia del caso anterior, se planteo el uso de una maquina virtual para acelerar el proceso de ejecucion. Se eligió estas librerías gracias a que nos presentan los algoritmos y tipo de datos listo para trabajar. A continuación, se mencionara el entorno de trabajo realizado para la experimentación://

- Sistema Operativo: Ubuntu 16
- Lenguaje de programación: Python 3.6
- Librerías utilizadas: Numpy, NetworkX,pandas
- CPU: Desconocido(Maquina Virtual)
- Ram: 4 GB
- Entorno de trabajo: Jupyter
- Lenguaje para pre-procesamiento: R

Una vez listo el entorno de trabajo, el siguiente paso es descargar el dataSet de Social Circles Facebook, disponible dentro de la base de datos de SNAP, con el archivo facebook_combined.txt. Una vez obtenido el archivo, se requiere un pre-procesamiento previo para poder trabajar de manera más efectiva el algoritmo.

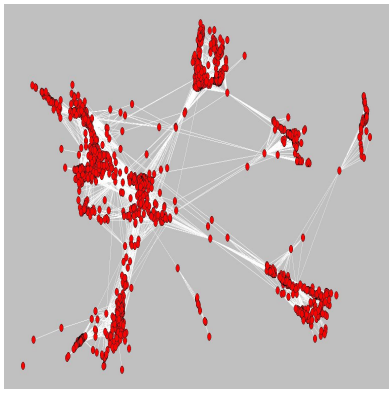

Figure 6: Red de entrenamiento

Para esto, se usó el entorno Rstudio que nos permite trabajar con data tabulada de manera rápida. Por medio de este proceso, se eliminó los nodos que tenían un grado muy bajo ademas, se redujo el numero de nodos a 11000. Al ser una separación aleatoria, se realizó este proceso 10 veces, pues la intención es correr los algoritmos en 10 datasets similares mas no iguales, para obtener una data y poder promediar.

```
def minimizedCN(H,nodeH,dataTestEntrenamiento):
    df = []
    for n in nodeH:
        print(n)
        for n2 in nodeH:
            if H.has_edge(n,n2) == True or n == n2 or len(sorted(nx.common_neighbors(H, n, n2))) == 0 :
                continue
            else:
                df.append([n,n2,list(nx.common_neighbors(H, n, n2))])
    return df

def minimizedAdamic(nodeH,dataTestEntrenamiento,H,dfAdamic):
    dfA = []
    for n in nodeH:
        for n2 in nodeH:
            if H.has_edge(n,n2) == True or n == n2 or sorted(nx.adamic_adar_index(H, [(n, n2)]))[0][2] is 0 :
                continue
            else:
                dfA.append([n,n2,sorted(nx.adamic_adar_index(H, [(n, n2)]))[0][2]])
    return dfA
```

Figure 7: Codigo de Common Neighbors

Para poder ejecutar una comparación efectiva, se ha considerado los siguientes datos a obtener por cada algoritmo aplicado: Tiempo de ejecución, precisión. Estos datos seran comparados con los resultados anteriormente obtenido

Todo lo mencionado a continuación es aplicado para cada DataSet generado. Una vez leído el archivo, este se guarda dentro de un tipo de dato específico de networkx, que permite trabajar redes. El siguiente paso a tomar en cuenta es el de separar los datos en dos, data de entrenamiento y data de prueba, para esto se ha decidido tomar una razón de 80% para data de entrenamiento y 20% data de testeo.

Luego de ser generado, es importante recordar que la red neuronal es un algoritmo supervisado, y los datos de las redes complejas son no supervisadas, por lo cual es necesario transformar este grafo a una data supervisada(con clase). Para esto se hace una conversion, la cual se puede observar a continuacion:

```
if len(df3.merge(dfTest, how='inner' , on = ['0','1'])) > 0:
    f=f.append([str(n) + ' ' + str(n2), len(list(nx.common_neighbors(H, n, n2))),sorted(nx.adamic_adar_index(H, [(n, n2)]))[0][2], n2,cont1,cont2, 1])
elif len(df3.merge(dfTest, how='inner' , on = ['1','0'])) > 0:
    f=f.append([str(n) + ' ' + str(n2), len(list(nx.common_neighbors(H, n, n2))),sorted(nx.adamic_adar_index(H, [(n, n2)]))[0][2], n2,cont1,cont2, 1])
else:
    f=f.append([str(n) + ' ' + str(n2), len(list(nx.common_neighbors(H, n, n2))),sorted(nx.adamic_adar_index(H, [(n, n2)]))[0][2], n2,cont1,cont2, 0])
```

Figure 8: Codigo de transformacion

Luego, se realiza el entrenamiento del algoritmo de red neuronal, el cual utiliza cada algoritmo para la función, lo que da resultado un arreglo de entrenamiento, el cual contiene las conexiones esperadas(en este caso, probable amistad).
Una vez realizado el entrenamiento de cada función, estas son colocadas bajo la función Test(), el cual nos brindara el resultado de la precisión, esto luego imprimira en pantalla los resultados obtenidos, tanto el tiempo de ejecuccion como la precision.
Una vez finalizado se pasa esta data a tablas para poder realizar una apreciación de resultados.

5 CONCLUSIONES

Como se pudo observar en los resultados, existen diferentes factores para poder efectuar el algoritmo, todo depende de cual es la intención principal. En primer lugar, es obvio que el algoritmo de MLP va a consumir más tiempo de procesamiento, ya que este corre previamente todos los otros algoritmos mencionados.
Por otro lado, MLP supera los resultados obtenidos por un algoritmo de tiempo mayor que es el simrank, por lo cual ha demostrado que utilizando un algoritmo bio-inspirado, se puede optimizar algoritmos rapidos para una obtencion de resultados cercanos a otros algoritmos mas pesados en procesamiento sin sacrificar tanto tiempo de ejecucion.

```
def AUC(datav,dataTestEntrenamiento):
    print(datav)
    print(dataTestEntrenamiento)
    DataVector = pd.DataFrame(data = datav, columns = ['N1','N2','Score'])
    print(DataVector)
    acertados = (DataVector.merge(dataTestEntrenamiento, how='inner', on = ['N1','N2']))

    general = pd.concat([DataVector,acertados])

    general.drop_duplicates()

    n = int((acertados.size + 10) / 10)

    gEscogidos = general.sample(n = n)
    aEscogidos = acertados.sample(n = n)

    generalContador = 0
    acertadoContador = 0

    for i in range(n):
        if (aEscogidos.iat[i,2] > gEscogidos.iat[i,2]):
            acertadoContador = acertadoContador + 1
        else:
            generalContador = generalContador + 1

    auc = ((acertadoContador + (generalContador * 0.5))/n)
    return auc
```

Figure 9: Codigo de captura de AUC

```
def AUC(datav,dataTestEntrenamiento):
    print(datav)
    print(dataTestEntrenamiento)
    DataVector = pd.DataFrame(data = datav, columns = ['N1','N2','Score'])
    print(DataVector)
    acertados = (DataVector.merge(dataTestEntrenamiento, how='inner', on = ['N1','N2']))

    general = pd.concat([DataVector,acertados])

    general.drop_duplicates()

    n = int((acertados.size + 10) / 10)

    gEscogidos = general.sample(n = n)
    aEscogidos = acertados.sample(n = n)

    generalContador = 0
    acertadoContador = 0

    for i in range(n):
        if (aEscogidos.iat[i,2] > gEscogidos.iat[i,2]):
            acertadoContador = acertadoContador + 1
        else:
            generalContador = generalContador + 1

    auc = ((acertadoContador + (generalContador * 0.5))/n)
    return auc
```

Figure 10: Codigo de captura de AUC

Table 1: Resultados previos de prueba Common Neighbors

Repetición	AUC	Tiempo
1	0.967	197.019
2	0.967	194.62
3	0.967	193.197
4	0.967	193.19
5	0.967	193.195
6	0.96947	194.95s
7	0.9684	194.62s
8	0.970	193.55
9	0.966	202.16
10	0.967	191.94

Por lo tanto, como se mencionó en el documento anterior, es importante decidir cual es el método adecuado.Se puede decir que es una mejora con respecto a la anterior situacion de performance vs tiempo de ejecucion. Desde el punto de vista analizado, y para este caso específico, se recomienda la implementacion y uso del algoritmo MLP para la mejora de los modelos de prediccion, ya que su tiempo de entrenamiento no es muy alto, y su programacion es sencillo gracias a las librerias en python.

Table 2: Resultados previos de prueba Adamic-Adar

Repetición	AUC	Tiempo
1	0.9678	197.019
2	0.9755	274.972
3	0.9772	273.141
4	0.9772	273.141
5	0.976	271.56
6	0.979	282.577
7	0.975	273.13
8	0.977	272.566
9	0.9759	270.61
10	0.977	273.085

Table 3: Resultados de prueba de MLP

Repetición	precision	Tiempo
1	0.98271	687.214
2	0.9825	662.608
3	0.9836	658.928
4	0.9844	661.224
5	0.9835	670.145
6	0.9831	630.6734
7	0.98271	687.214
8	0.9825	662.608
9	0.9840	658.928
10	0.9822	654.139

Table 4: Resultados de prueba SimRank(algoritmo pesado)

Repetición	AUC	Tiempo
1	0.9840	3750.352
2	0.984375	3794.223
3	0.9865	3784.34
4	0.9842	3761.137
5	0.9864	3766.51
6	0.9855	3706.589
7	0.98397	3779.265
8	0.984	3705.991
9	0.9846	3674.305
10	0.9829	3699.129

6 BIBLIOGRAFIA

REFERENCES

[1] M.E.J. Newman *The Structure and Function of Complex Networks* 2003 Society for Industrial and Applied Mathematics

[2] L. Getoor y C. P. Diehl *Link Mining: A Survey* Department of Computer Science/UMIACS University of Maryland

[3] D. Liben-Nowell y J. Kleinberg *The Link Prediction Problem for Social Networks* Laboratory for Computer Science Massachusetts Institute of Technology

[4] Weiren Yu · Wenjie Zhang *A Space and Time Efficient Algorithm for SimRank Computation* School of Computer Science & Technology Donghua University, Shanghai, China

[5] Barabás *Network Science Book*

[6] Jorge Valverde Rebaza *Mining user behavior in location-based social networks*