Implementación de un red neuronal de atención sobre grafos para determinar la causalidad entre variables aleatorias.

Tomás Gómez Zuleta*
Instituto de física, Universidad de Antioquia U de A, calle 70 No. 52-21, Medellín, Colombia.
(Dated: August 23, 2025)

En el presente trabajo se aborda el problema de determinar las relaciones causales entre diversas variables para un conjunto de datos. Para ello se emplea el algoritmo PC que tiene como retorno un grafo parcialmente dirigido. Pero, aquel algoritmo PC suele ser deficiente, ya que para casos sencillos no es concluyente acerca de como es la causalidad entre las variables, dejando unicamente una correlación entre las variables. Para solucionar este problema se propuso la implementación de las GAT. Se empleó el algoritmo PC y las GAT sobre unos datos sintéticos generados a partir de un contexto simple con cinco variables aleatorias relacionadas con las infecciones virales. Se obtuvo un resultado satisfactorio para este primer entrenamiento del modelo.

I. INTRODUCCIÓN

En la actualidad el análisis de datos y la aplicación de los métodos de estadística para la extracción de información constituye el eje principal para el estudio de nuevos fenómenos. No obstante, este análisis estadístico no entrega una alta interpretabilidad de los resultados. A modo de ejemplo considérese el coeficiente de correlación de Pearson. Empleando este coeficiente se puede determinar el grado de correlación[1] entre dos variables aleatorias, pero dicha correlación no indica la causalidad entre ellas o si existe una tercera variable que produce ambas. Tal es el caso del aumento de ataques de tiburón en una playa, dado que hace calor, asimismo el aumento en la venta de helados. En realidad no existe correlación alguna entre los ataques de tiburón y la venta de helados, pero como existe una tercera variable "oculta" (hace calor) el coeficiente de correlación de Pearson indicará correlación.

Esta falta de interpretabilidad de los datos da lugar a que se requiera herramientas adicionales para poder determinar adecuadamente la relación entre las variables aleatorias, la rama que se dedica a este estudio es la inferencia causal.

Por está razón el presente trabajo se enfoca en la programación de una red neuronal de atención sobre grafos (GAT-*Graph attention network*) para apoyar la funcionalidad del algoritmo PC que permite determinar la causalidad entre variables aleatorias, puesto que dicho algoritmo puede ser deficiente para la decisión de causalidad entre las variables(Ver subsección B).

A. Inferencia Causal

La inferencia causal permite a partir de los datos obtener tres niveles de conocimiento denominados como predicciones, intervenciones y contrafactuales. En el presente trabajo se abordará con detalle los niveles de conocimiento de predicción y intervención.

1. *Predicción*

En este nivel de conocimiento se busca establecer la causalidad entre las variables aleatorias. Dicha casualidad se establece mediante el uso de grafos, exactamente grafos acíclicos dirigidos, con los cuales se cuenta con una poderosa herramienta gráfica que representa las relaciones causales entre las variables.

Los vértices del grafo son las variables aleatorias, y en una primera instancia las aristas entre dos vertices dados indican una correlación entre dichas variables. Cuando a todas las aristas se le asignan una dirección se obtiene un grafo acíclico dirigido[2], y cuando existen aristas sin dirección establecida se obtiene un grafo parcialmente dirigido acíclico.

En el caso de los grafos acíclicos se dicen Markov relativo a una distribución de probabilidad P, cuando se satisface la siguiente ecuación:

$$P(x_1, ..., x_n) = \prod_i P(x_i|pa_i) \tag{1}$$

Está descomposición lo que quiere decir es que la probabilidad (modelo imbuido en los datos) representan las relaciones causales entre las variables observadas en el grafo, indicado por el autor Judea Pearl[3] como un isomorfismo donde se preservan las características del grafo.

Intuitivamente representa que el conocimiento de ciertas variables preferenciales es suficiente para predecir completamente el comportamiento de otras variables. Estas variables "preferenciales" pa_i serían los padres del nodo i-ésimo, esto es, la relación "parental" hace ilusión que el comportamiento de la variable i-ésima está determinada completamente por el conjunto de variables $x \in pa_i$. Entonces el conocimiento de cualquier variable adicional no aporta información a la variable i-ésima. Para cada variable aleatoria del conjunto se pueden determinar estos padres que indican la causalidad entre las variables.

* tomas.gomez@udea.edu.co

2. Intervención

En la anterior sección se consideró los grafos Markov relativos a una distribución de probabilidad P, lo cual es un modelo no determinista de los datos. Existe otra forma de determinar aquella distribución de probabilidad P a partir de un modelo determinista.

Estos modelos son las relaciones funcionales entre las variables aleatorias, donde estás relaciones son de la forma: $x_i = f_i(pa_i, u_i)$. La anterior ecuación denota las repercusiones que se tiene sobre los descendiente a partir de un cambio en sus padres. Luego, la aleatoriedad se introduce en aquel factor u_i.

La principal ventaja de este modelo determinista se encuentra en las posibles intervenciones que se pueden realizar en el modelo. Por intervención se comprende que alguna variable aleatoria se le asigna un valor fijo, denotado como $do(x)$, siendo $X = x$.

Dicho valor altera completamente la estructura del grafo, puesto que cada variable y sus relaciones con las otras variables funcionan como engranajes de una maquinaria, entonces aquella intervención produce un nuevo grafo y nuevas relaciones. Cabe señalar que una intervención es distinto a $P(x_1, ..., x_n|X_j = x_j)$, puesto que aquella probabilidad condicional simplemente particulariza un valor, no la construcción de un nuevo grafo.

Estas intervenciones permiten realizar predicciones de resultados sin necesidad de una experimentación directa. Esto representa una basta ventaja en ámbitos donde la experimentación es costosa o compleja. Esta ventaja se quiere explorar en el presente trabajo en el ámbito de la predicción de la estancia hospitalaria (LoS) para el hospital Alma Mater de la Universidad de Antioquia.

La finalidad del proyecto es aplicar un modelo causal al *dataset* LoS (por sus siglas en ingles *length of stay*), el cual se determina completamente a partir del Algoritmo PC y el suplemento de las GAT, pero por limitaciones de tiempo aquello no se realizó.

B. Algoritmo PC

Antes de enunciar el Algoritmo PC que se ha mencionado a lo largo del trabajo, veamos algunos aspectos claves que denotan la complejidad de la aplicación del Algoritmo PC.

El primero de estos aspectos es la unicidad de las modelos causales. Para hablar acerca de la unicidad veamos la definición formal de estructura causal, latente y modelo causal. Segun Judea Pearl[3]:

Definition I.1 (Estructura Causal) *Una estructura causal de un conjunto de variables V es un grafo acíclico dirigido (DAG) en el que cada nodo corresponde a un elemento distinto de V, y cada enlace representa una relación funcional directa entre las variables correspondientes.*

Definition I.2 (Modelo Causal) *Un modelo causal es un par $M = \langle D, \Theta_D \rangle$ que consiste en una estructura causal D y un conjunto de parámetros Θ_D compatibles con D. Los parámetros Θ_D asignan una función[4] $x_i = f_i(pa_i, u_i)$ a cada $X_i \in V$ y una medida de probabilidad $P(u_i)$ a cada u_i, donde PA_i son los padres de X_i en D y donde cada U_i es una perturbación aleatoria distribuida de acuerdo con $P(u_i)$, independientemente de todos los demás u.*

De la definición de estructura causal se desprende inmediatamente la definición de estructura latente L. Una estructura latente L es una estructura causal, pero definida sobre el conjunto de variables observables, es decir, existe un $O \subset V$, donde O es la cantidad de variables observables y se construye un modelo causal unicamente teniendo en cuenta O.

En nuestro caso siempre vamos a considerar la determinación de L, esto es, obviamos la existencia de alguna variable oculta que no se está teniendo en cuenta que pueda influir en el modelo. Consecuentemente las formalizaciones respecto al alcance de estas variables ocultas no será mencionado aquí.

Ahora, aquella estructura latente para un conjunto de datos dada no es única, en realidad existen un número muy grande (infinito si así se prefiere) de estructuras latentes que pueden hacer "mímica" de los datos. Cuando se dice mímica es que otras estructuras latentes pueden predecir las relaciones causales entre las variables sobre el mismo conjunto de datos O. Entonces estas dos estructuras latentes se dicen observacionalmente equivalentes.

Dicho lo anterior parece inútil la búsqueda de una estructura causal de un conjunto de datos, no obstante se debe indicar que existe un método para "preferir" una estructura latente sobre otra. Emplear iterativamente ese método sobre una clase de estructuras latentes dará lugar a la estructura latente mínima.

La estructura latente mínima cuenta con dos características fundamentales, la primera de ella es la estabilidad. Por estabilidad se entiende que para cualquier variación de los parámetros Θ_D no se pierdan relaciones causales previamente observadas entre las variables. Cuando desaparecen dichas relaciones causales indica que la relación causal observada anteriormente fue resultado de una combinación particular de parámetros Θ_D, no una relación causal verdadera entre las variables. La segunda característica como su nombre lo indica, la estructura latente debe ser mínima, es decir, para cualesquier otra estructura latente, la estructura latente mínima puede hacer mímica. Adicionalmente, no debe existir otra estructura latente que pueda hacer mímica de la estructura mínima y de cualquier otra, porque en ese caso sería la estructura mínima, es decir, son dos estructuras equivalente.

Por último cabe mencionar que una ventaja de trabajar con estructuras latentes mínimas es que un modelo es menos falsable entre más simple sea, indicando que siempre se buscará la mayor simpleza posible en un modelo para indicar causalidad entre las variables.

Teniendo en cuenta lo discutido a lo largo de está sección se observa el grado de dificultad que se tiene para poder inferir una relación causal a partir de un conjunto de datos. Esta problemática la intenta resolver el algoritmo PC[5], buscando siempre prevalecer las independencias condicionales que dan lugar a las ν-estructuras en el grafo. Una ν-estructura es cualquier arreglo de dos conexiones tal que su dirección apunta a un mismo vértice, pero que no son adyacentes.

Por último se debe indicar que dada la complejidad mencionada, el algoritmo PC suele ser inconcluso. Inclusive en situaciones relativamente sencillas. Esto es lo que se observó (ver Metodología) cuando se le entrego al algoritmo PC unos datos sintenticos con unas relaciones funcionales preestablecidas. Por lo mismo se implementaron las GAT para apoyar su funcionalidad (ver subsección C).

Algorithm 1: Algoritmo IC (*Inductive Causation*), determina el grafo parcialmente dirigido asociado a un conjunto de datos.

Input: P, distribución de probabilidad estable sobre un conjunto de variables V.
Output: Grafo parcialmente dirigido $H(P)$ compatible con P

1 **foreach** *par de variables* $a, b \in V$ **do**
2 Buscar el conjunto S_{ab} tal que a sea condicionalmente independiente de b dado S_{ab} en P;
3 Construir un grafo sin dirección con una conexión entre a y b, solo si no existe tal conjunto S_{ab};
4 **foreach** *par de variables no adyacentes* a, b *con un vecino común* c **do**
5 Verificar si c pertenece a S_{ab};
6 **if** $c \in S_{ab}$ **then**
7 No realice nada y continúe;
8 **else**
9 Agregar dirección a las flechas apuntando a c, es decir: $a \rightarrow c \leftarrow b$;
10 Orientar el resto de las conexiones en el grafo parcialmente dirigido teniendo en cuenta;
- Cualquier orientación alternativa crearía una nueva ν-estructura.
- Cualquier orientación alternativa resulta en un ciclo dirigido.

Al anterior algoritmo se le adicionan la siguientes reglas de elección para la direccionalidad de los vertices:

- Orientar b-c como b→c cuando existe la orientación a→b, tal que a y c son no adyacentes.
- Orientar a-b como a→b cuando existe una cadena: a→c→b
- Orientar a-b como a→b cuando existen dos cadenas: a-c→b y a-d→b tal que c y d son no adyacentes.
- Orientar a-b en a→b cuando hay dos cadenas: a-c→d y c→d→b tal que c y b son no adyacentes y, a y d son adyacentes.

C. Redes neuronales de atención sobre grafos (GAT)

En el presente trabajo se empleará para entrenamiento del modelo una red neuronal, pero está no puede ser una clásica red neuronal densa, dado el tipo de datos empleados (grafos) se empleará una red neuronal gráfica (GNNs - *graph neuronal network*).

Existen diversos tipos de redes neuronales gráficas, entre ellas son de particular interés las GCNs por sus siglas en ingles (*graph convultional network*) y las GAT. Las primeras son el análogo de las redes neuronales convulsionales tradicionales, porque el concepto de aprendizaje en ambas redes neuronales es el mismo, su diferencia radica en el tipo de datos que cada una considera, donde las CNNs consideran pixeles y las GCNs consideran grafos.

Una red neuronal convulsional aprende de considerar un pixel central y combinar la información de los vecinos aplicando la operación de convolución. En el caso de una red neuronal gráfica convulsional la idea es la misma, aquella red neuronal aprende a partir de considerar un nodo y combinar la información de sus nodos vecinos, teniendo en cuenta las características de cada nodo y la conexión entre las diversas variables (estructura de los datos).

Como se podría esperar las GNCs dependen de la estructura de los datos, lo cual representa una desventaja al momento de generalizar el entrenamiento de un red neuronal[6]. Consecuentemente las GNCs suelen ser útil unicamente para la clasificación de los nodos.

Cuando se desea clasificar la direccionalidad entre las conexiones de los diversos nodos se debe emplear otro tipo de agregación de los datos que no sea altamente dependiente de la estructura de los mismo, lo cual es el fundamento de las GAT. En este caso el conjunto de datos se agrupa considerando el peso de las características de los vecinos (no todos los vecinos aportan de igual forma entre las variables aleatorias) y normalización libre de estructura de los datos.

Este procedimiento de agregación de los datos en el entrenamiento se pueden observar en las siguientes cuatro ecuaciones:

$$z_i^{(l)} = W^{(l)} h_i^{(l)} \tag{2}$$

$$e_{ij}^{(l)} = LeakyReLU(\vec{a}^{(l)^T}(z_i^{(l)} || z_j^{(l)})) \tag{3}$$

$$\alpha_{ij}^{(l)} = \frac{exp(e_{ij}^{(l)})}{\sum_{k \in \mathcal{N}(i)} exp(e_{ik}^{(l)})} \tag{4}$$

$$h_i^{(l+1)} = \sigma\left(\sum_{j\in\mathcal{N}(i)} \alpha_{ij}^{(l)} z_j^{(l)}\right) \tag{5}$$

La ecuación (2) toma el vector de características $h_i^{(l)}$ del nodo i-ésimo en la capa l-ésima y le aplica una transformación lineal, mediante la matriz de pesos $W^{(l)}$ entrenable, la cual es compartida por todos los nodos de la misma capa.

Posteriormente, la segunda ecuación nos define un puntaje de la atención de un nodo respecto a otro. Es decir, está ecuación está indicando la relevancia que tiene el nodo j-ésimo respecto a su nodo vecino i-ésimo en la l-ésima capa de la red. Esto es, la segunda ecuación toma los vectores transformados $z_i^{(l)}$ y $z_j^{(l)}$, luego los concatena (aquellas dos barras en la ecuación significan concatenación) en un vector de dimensión $dim(z_i^{(l)}||z_j^{(l)}) = dim(z_i^{(l)}) + dim(z_j^{(l)})$. Posteriormente realiza el producto interno con un vector $\vec{a}^{(l)}$ de pesos entrenable, el cual "aprende" la relevancia de combinar las características de los nodos i, j. Entonces su producto interno valida la relevancia que supone el envío de información desde el nodo j al i[7]. Por ultimo el resultado obtenido pasa por la función de activación no lineal *LeakyReLU* definida como:

$$LeakyReLU(x) = \begin{cases} x & : x > 0 \\ \alpha x & : x \leq 0 \end{cases} \tag{6}$$

En la definición de la función de activación *LeakyReLU* α suele tener un valor de 0.01. Está función de activación permite tener en cuenta la contribución de valores negativos de x, pero con un factor de reducción. Esta propiedad brinda una mayor "expresividad" en comparación de la función de activación *ReLU* que aniquila completamente estos valores.

La ecuación (4) es simplemente una normalización de los puntajes de atención $e_{ij}^{(l)}$ de los vecinos $j \in \mathcal{N}(i)$. Aquel conjunto $\mathcal{N}(i)$ es el conjunto de nodos que están conectados con el nodo i-ésimo mediante una única arista.

Finalmente la ecuación (5) es donde ocurre la agregación de los datos. La característica del i-ésimo nodo en la capa $l+1$ viene dado por la agregación ponderada de la información que aporta cada vecino, esto es, la suma aritmetica de la multiplicación del peso de atención de la j-esima neurona por su respectivo vector transformado de característica: $\sum_{j\in\mathcal{N}(i)} \alpha_{ij}^{(l)} z_j^{(l)}$. Finalmente aquel resultado pasa por una función de activación σ, la cual suele ser *ReLU*. Así se observa que la característica de la siguiente capa tiene en cuenta la información de todos los vecinos inmediatos dado su puntaje de atención.

Al igual que las redes neuronales convulsionales podemos tener diversos filtros, en nuestro contexto de las redes gráficas de atención tendríamos "varias cabezas de atención". Esto es, con las cuatro primeras ecuaciones obtenidas podemos tener una única atención, entonces si aplicamos iterativamente aquel algoritmo obtendríamos "varias cabezas", donde estás se pueden combinar en un único resultado. Claro está, estas "cabezas" no se pueden combinar de cualquier forma, existen dos formas para hacerlo dependiendo de la capa en cuestión. Para las capas ocultas las distintas cabezas se combinan mediante la siguiente ecuación:

$$h_i^{(l+1)} = ||_{k=1}^{K} \sigma\left(\sum_{j\in\mathcal{N}(i)} \alpha_{ij}^{k} W^k h_j^{(l)}\right) \tag{7}$$

La ecuación (7) se dice concatenación, porque aquella básicamente lo que estaría haciendo es concatenar en un vector de dimensión $K \cdot d'$ (en este caso K es el número de cabezas empleadas y d' es la dimensión del vector resultante de cada cabeza).

Luego, para la capa de salida se combinan las distintas cabezas en un promedio mediante la siguiente ecuación:

$$h_i^{(l+1)} = \sigma\left(\frac{1}{K}\sum_{k=1}^{N}\sum_{j\in\mathcal{N}(i)} \alpha_{ij}^{k} W^k h_j^{(l)}\right) \tag{8}$$

En este caso la dimensión es la misma, de modo que simplemente se estaría teniendo en cuenta las "variaciones" que observa cada una de las "cabezas".

Este entrenamiento de "varias cabezas" se realiza para darle una mayor capacidad y estabilidad al aprendizaje del modelo. Puesto que, intuitivamente se puede entender la ecuación (7) como que el algoritmo está aprendiendo cada una de las perspectivas que brinda cada cabeza, es decir, como si una persona estuviera viendo la misma fotografía desde distintos ángulos. La ecuación (8) se puede entender también intuitivamente como que la salida final ya tiene en cuenta todos los ángulos posibles para la misma fotografía (grafo) y se promedian todos estas observaciones para reconocer las diferencias entre las distintas observaciones y así indicar cual es la representación más plausible de la fotografía (grafo) en cuestión.

II. METODOLOGÍA

La metodología se encuentra dividida en dos procedimientos: (1) *Datos sintéticos:* se crea un modelo causal para emplear y evaluar la funcionalidad del algoritmo PC, (2) *Entrenamiento:* se entra una GAT para predecir la orientación más probable en las aristas entre dos variables.

A. Datos sintéticos

Para evaluar la funcionalidad del algoritmo PC se propuso la creación de datos sintéticos con relaciones causales impuestas a través de modelos funcionales entre las variables, esto es, con unos datos sencillos se quiere verificar si el algoritmo es capaz de reconocer una relación causal conocida a priori.

Estos datos sintéticos se crearon con el siguiente contexto: Escenario clínico donde se analizan distintas variables asociadas a la presencia de una infección viral. Las variables a considerar son:

- *InfecciónViral*: Variable binaria que indica la presencia de la infección viral (1 = sí, 0 = no).
- *Estrés*: El nivel de estrés percibido por el paciente. Esta es una variable aleatoria entera que toma valores en una escala de 0 a 10.
- *TemperaturaCorporal*: Medida de la temperatura corporal en $C°$. Esta es una variable aleatoria tipo continua.
- *DolorCabeza*: Intensidad del dolor de cabeza percibido por el paciente en una escala del 0 a 10.
- *ResultadoPCR*: Variable tipo binaria que indica el resultado de la prueba PCR (1 = Positivo, 0 = negativo)

Dado el contexto anterior se pueden inferir que las variables *InfecciónViral* y *Estrés* causan el resto de variables. Para producir los datos se impuso que la variable aleatoria *InfecciónViral* se distribuye de forma binomial: $InfecciónViral \sim \mathbb{B}(0.4)$. Indicando que hay una probabilidad de 0.4 que la persona tenga la infección viral. Asimismo se impuso que el estrés se distribuye de forma uniforme: $Estres \sim Uniforme(0, 10)$. Esto básicamente me da un valor entre 0-10 aleatorio, que indica el dolor de cabeza percibido por la persona. Luego, las relaciones funcionales son:

$$TempCorporal = 36.5° + 1.5(Infección) + N(0, 0.3)$$

Está relación funcional indica que el aumento en la temperatura de la persona es en 1.5° si está infectada, además que hay un ligero aumento por una distribución normal con media en 0 y desviación estándar 0.3, siendo una especie de aleatoriedad en los datos.

$$DolorCabeza = 2 + 1(Infección) + 0.5(estres) + N(0, 1)$$

Está relación funcional indica que el dolor cabeza normal de una persona es de 2, algo leve y imperceptible. Luego, aquel dolor de cabeza aumenta en una unidad si tiene la infección viral, adicionalemente aumenta 0.5 veces el nivel de estrés percibido y hay una aleatoriedad inducida por una distribución normal de media 0 y desviación estándar 1.

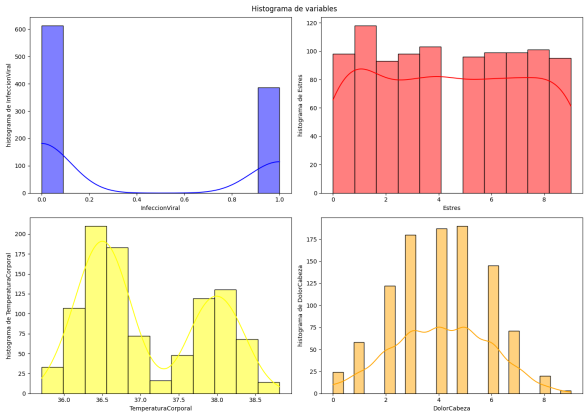

FIG. 1. Histogramas de las variables Infección Viral, Estres, Temperatura corporal y Dolor de cabeza

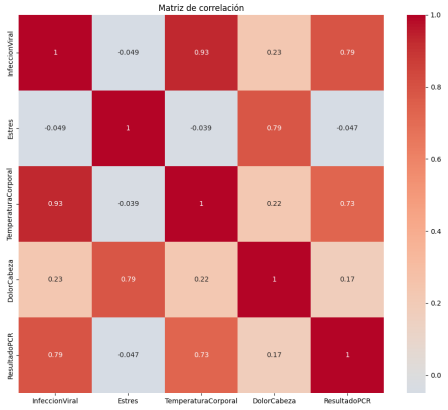

FIG. 2. Matriz de correlación de los datos sinteticos

$$ResultadoPCR = \mathbb{B}(0.9(Infección) + 0.1(1 - Infección))$$

Finalmente esta relación funcional indica que la probabilidad de éxito para la distribución binomial que me indica los resultados de la prueba PCR en un 90% se debe a que efectivamente la persona tiene el virus, y hay un 10% asociado a un error donde se indica que la persona está infectada, pero en realidad no cuenta con el virus.

En la figura 1 se observan los histogramas que verifican todas estás distribuciones de probabilidad y relaciones funcionales impuestas entre las variables.

Posteriormente en la figura 2 se puede observar como es la correlación entre los datos en una matriz de correlación. Aquella matriz indica que efectivamente las relaciones funcionales impuestas se reconocen mediante las herramientas estadísticas como correlación entre las diversas variables.

De la relaciones funcionales impuestas entonces se deriva el grafo causal que se puede observar en la figura 3.

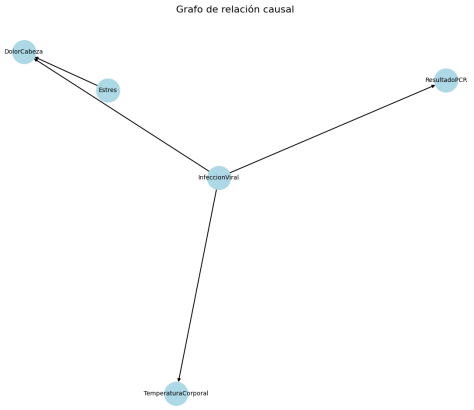

FIG. 3. Grafo causal que denota las relaciones funcionales impuestas entre las variables.

B. Entrenamiento

En esta sección se abordará lo que fue el entrenamiento de las GAT. Aquella implementación se realizo basada en el código provisto por el articulo [6]. El primer problema para poder implementar este código fue la inconsistencia observada entre las librerías `dgl` y `pytorch`. Esta inconsistencia nace del hecho que la librería `dgl` empleada fue instaurada en 2018 cuando se propuso por primera vez el algoritmo de las GAT, siendo una librería algo "antigua".

Por el contrario la librería `pytorch` ha ganado popularidad siendo actualizada varias veces, al punto que actualmente la maquina de `google coolab` cuenta con la versión 2.8 de `pytorch`, que tuvo ser bajada hasta la versión 2.2 que si es compatible con la librería `dgl`.

Otro problema apareció con los tipo de objetos que manejan aquellas librerías `dgl` y `pytorch`, estos objetos son tensores. Esto presentó otra inconsistencia respecto a la versión moderna de `numpy`, de modo que se tuvo que instalar la versión 1.26.4 de `numpy`.

La cantidad de datos empleados para el entrenamiento fue un *dataset* de $n = 3000$ datos, lo que quiere decir que se tuvo 3000 datos para cada variable aleatoria. Luego a este conjunto de datos se realizo *bootstraping* para obtener un *dataset* con 300 datos. A cada uno de estos 300 datos se obtuvo la predicción del algoritmo PC, posteriormente se convirtió cada resultado a un grafo de la librería `dgl`. Estos 300 grafos obtenidos fueron los datos de entrenamiento.

Los detalles de la programación se encuentran en el archivo `Algoritmo PC.ipynb` (ver apéndice)

III. RESULTADOS Y ANÁLISIS

Se obtuvieron dos resultados de la aplicación del modelo, el primero de ellos se encuentra en la figura 4. Este resultado denota que efectivamente las GAT pudieron predecir las direcciones impuestas por el modelo. El segundo de los resultados es la métrica de la matriz de confusión. En esa matriz tendríamos una medida de como clasifica el algoritmo los verdaderos positivos, falsos negativos, falsos positivos, etc. En nuestro caso particular el algoritmo está evaluando cuantas veces se predijo se forma adecuada o inadecuada la dirección de la arista que une dos variables aleatorias.

Esta matriz de confusión se observa en la figura 5. En dicha matriz de confusión se observa que el algoritmo es "perfecto" y siempre predice adecuadamente los resultados.

Aquí se debe mencionar varias cosas respecto a los resultados obtenidos anteriormente. No debe ser sorprendente que los resultados obtenidos sean perfectos, esto se debe a que básicamente el modelo tiene todo para aprender, es decir, se entreno ya sabiendo cual era la relación causal. Como se mencionó anteriormente saber esto no es posible para problemas más complejos que involucre un alto número de variables. Todo esto se realizó teniendo la ilusión que este modelo entrenado fuera generalizable para el *dataset* LoS.

Aquel *dataset* LoS es uno con 79 variables, de las cuales por el método del peso de la evidencia se redujo aquel conjunto solo a 9 variables relevantes para predecir la estancia hospitalaria. Se creía cuando se planteó el proyecto que se podría generalizar la anterior programación para que algoritmo aprendiera y predijera la causalidad entre las 79 variables teniendo en cuenta como que el modelo establecido para un caso sencillo podría aprender en pequeños bloques para el ejemplo más complejo.

Estudiando a fondo la estructura de los grafos se aprendió a lo largo del trabajo que los grafos son estructura de datos complejas y no lineales, lo que quiere decir que aquella conexión que se dice sencilla (una simple flecha señalando causalidad) en realidad significa que los datos cuenta con una estructura bien definida en el espacio. Por lo tanto, estos código de las GNNs son difícilmente generalizables, siendo muy particularizados.

Pero, no todo está perdido, con lo que se aprendió a lo largo del documento se espera en una próxima entrega del proyecto aplicar la algoritmia de las GAT al problema LoS, teniendo en cuenta que las GAT pueden aprender relaciones complejas y el algoritmo tiene una alta expresividad. Entonces, en vez de esperar obtener un grafo con todas las relaciones causales se entrenaría un modelo desde cero.

IV. CONCLUSIONES

Se concluye que las GAT son más efectivas que las CGNs para la aplicación de problemas más complejos, donde no se busque simplemente clasificar nodos.

Se concluye que los grafos cuentan con una estructura en el espacio, haciéndolos una estructura de datos no lineal y compleja, por ende, difícilmente se obtiene una generalización de algún modelo de aprendizaje estadístico.

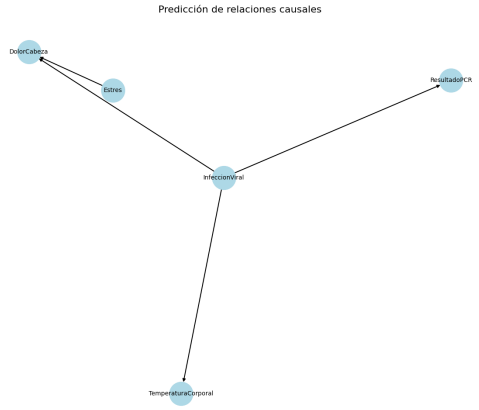

FIG. 4. Grafo causal predicho por las GAT. Este grafo fue construido a partir de la predicción de los 3000 datos creados sintéticamente.

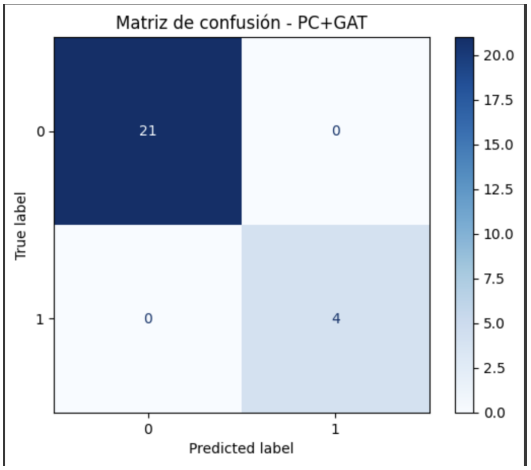

FIG. 5. Métrica del rendimiento del algoritmo de las GAT: Matriz de confusión. En este caso vemos que la predicción es perfecta, puesto que el modelo es sencillo.

Por ultimo se concluye que las GAT pueden aprender relaciones complejas y tener una alta expresividad como modelo gracias a su aprendizaje de atención, donde se le asigna un peso a la relevancia de la información que aportan los vecinos de un nodo particular.

V. APÉNDICE

Aquí se encuentra un *link* al archivo donde se realizó la programación: `https://colab.research.google.com/drive/1owL9l0qT4lCQONjQXgzk6JgBnW5Y5Dfg?usp=sharing`

[1] S. Sirca, *Probability for Physicists* (Springer International Publishing Switzerland, 2016).

[2] El grafo no debe ser cíclico, porque en ese caso carece de sentido alguno hablar de causalidad entre las distintas variables, puesto que sería un sistema autocontenido. Por ejemplo, las causas del movimiento de un sistema mecánico se debe a la interacción con otro cuerpo. Si se emplea un grafo que relacione las causas del movimiento este debe ser acíclico, donde las causas son las fuerzas (padres Markovianos) y el efecto es el movimiento del sistema. Sí se aceptará la posibilidad de que el grafo fuera cíclico este indicaría que un cuerpo puede cambiar su estado de momento al interactuar consigo mismo, lo cual se ha demostrado que no es posible.

[3] J. Pearl, *Causality, models, reasoning, and inference* (Cambrigde, 2009).

[4] Cuando se indica que existen unos parámetros en el modelo causal, estos hacen referencia a la proporcionalidad entre las distintas variables en la relación funcional. Suena complejo, pero considérese un ejemplo sencillo, donde una variable X_i tiene como padres $PA_i = \{X_{i-1}, X_{i-2}\}$, con una relación lineal entre los datos: $x_i = \alpha x_{i-1} + \beta x_{i-2}$, entonces los parámetros serían $\alpha, \beta \in \Theta_D$.

[5] Cabe indicar que el algoritmo presentado en está sección es el algoritmo IC, el cual es una variante del algoritmo PC que optimiza uno de los pasos seguidos. Pero como el algoritmo original es el PC, entonces se menciona este a lo largo del articulo.

[6] Hao Zhang, Mufei Li, and Minjie Wang Zheng Zhang, Understand graph attention network, Paper Study with DGL (2018).

[7] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio, Graph attention networks, ICLR (2018).