社群媒體分析期末專案書面報告-Youtube 合作對象推薦系統

組別: 第一組
組員: 魏靖軒(R11722025)、江子涵(R12922082)、葉冠宏(R11943113)、許聖慧(R12H41006)

1. 研究動機

當我們看YouTube頻道的時候，演算法常常會基於使用者過去的瀏覽紀錄被推播各種影片。YouTube的創作者，即YouTuber，常常會藉由和其他頻道的創作者合作來去拉抬彼此的聲量，創作出不一樣的影片內容。然而，目前市面上似乎還未有一個演算法去針對創作者提供建議和誰合作，以及創作什麼樣類型影片的服務。

對一個訂閱量較少的YouTuber來說，和訂閱量較高的YouTuber合作並不一定意味著會提高瀏覽觀看數。相反地，對於較高訂閱量的YouTuber來說，和一些新嘗試的YouTuber合作也有可能擴展不同的影片類型或主題，提供不一樣的新鮮感，提高影片流量。種種的連結有可能和彼此合作的化學效應、個人特質、目標觀眾客群、年齡、影片主題等等有關。

我們的專案目標在於基於我們資料集節點的歷史合作關係去推薦一個新的合作組合，以提高觀看流量。

2. 文獻回顧

2-1 An ecommerce recommendation algorithm based on link prediction [1]
在這篇論文中，其是利用BGN(Bipartite Graph Network)以及SMN(Single-Mode Network)的網路去做link prediction，尋找還未建立，但未來可能會出現的連線。在模型中，其會根據不同節點特徵之間的相似度去計算彼此之間的距離函數，並依照此來做相關的預測。

2-2 Link recommendations: Their impact on network structure and minorities [2]
在這篇論文中，其嘗試去研究是否link prediction相關的演算法會比較偏好有較多in degree 或是比較有名的節點，而忽略掉了較少被連線或是弱勢的節點。結果發現，除了Node2Vec演算法之外，確實蠻多演算法的設計常常會忽略較弱勢的節點。

2-3 Network Link Connectivity Prediction Based on GCN and Differentiable Pooling Model [3]
這篇論文主要是利用整合Graph Convolutional Network (GCN) 和 improved differentiable pooling (IDF) 的技巧來加強連線預測的關聯性。在先前的模型中，Graph Neural Network (GNN)萃取較深度或是高度複雜的特徵上會較有困難。而IDF模型可以加強不同神經網路層上的特徵萃取，使其可以在相似度的歸納上更有效率。除此之外，其利用雙向的距離參數使節點之間的關係可以更加精確的被表達。最後，其也利用到common neighbour 的方法來萃取

subGraph，使模型的複雜度降低。

2-4 The Link Prediction Problem for Social Networks [4]
這篇論文的目的是研究學術合作網路中的聯結預測問題。通過討論多種基於社交網路中節點近似度的聯結預測方法，作者試圖預測未來可能出現的合作關係。主要討論的方法包括共同鄰居（Common Neighbors）、Jaccard係數、Adamic/Adar指數等。這些方法利用網路拓撲結構進行預測，而不考慮節點的屬性。實驗結果表明，這些方法的預測效果顯著優於隨機預測，顯示網路拓撲結構中確實包含了有用的信息。

2-5 Link prediction in complex networks: A survey [5]
這篇論文利用物理學觀點和方法來加強連線預測的關聯性，透過對複雜網絡的分析討論連線預測演算法。論文重點介紹了基於相似度的演算法，將其分為本地相似度指數、全局相似度指數和準本地相似度指數三類別。具體方法包括共同鄰居（Common Neighbors, CN）、資源分配（Resource Allocation, RA）和局部路徑（Local Path, LP）指數等。根據實驗結果，本地相似度指數（如共同鄰居）在計算效率和預測精度上都表現良好，而全局相似度指數（如資源分配）和準本地指數（如局部路徑）則能捕捉到更複雜的網路結構信息，提升預測的準確性。

3. 資料取得

3-1 爬蟲程式使用及資料集、特徵的取得

在期中提案的時候，我們曾經提出可能可以當作YouTuber特徵的幾項指標。例如:影片個人風格、影片類型的分布、觀看受眾的年齡分布、觀看次數、訂閱人數、性別、爭議度、共同評論觀眾的向量等等。

有關channelId、訂閱人數等，我們是利用Youtube Data Api來爬取該YouTuber的相關資訊。而該頻道的所有影片標題、影片發行日期、觀看人數等我們則是用api scrapetube。至於影片類別我們有利用到ChatGPT的api去分析判斷。然而，不管是哪一種api，其都有相當的限制。例如：Youtube Data Api每天都有針對不同爬蟲服務有一定Quota的總量限制，而ChatGPT api則是限制每天可以判斷回答多少字，至於scrapetube的api是有限制每個頻道最多只能爬取多少個有限的影片資訊。這些api的使用限制，再加上人力、時間的不足，某種程度限制了我們資料集可以擴增的節點數量。

對於影片個人風格的部分，由於可以成為成功YouTuber的個人特質不外乎樂觀、知性、搞笑、真性情、談吐等等，彼此不會差距太多，我們難以捕捉到更細微的個人特質，除非我們需要人工去瀏覽過該YouTuber頻道的每部影片。當然，我們也有可能去爬蟲抓取每部影片下方前十多按讚數的留言去判斷影片個人風格，然而我們發現大多數評論者僅僅是針對影片內容做出不同的感想分享，較少針對YouTuber個人做出特質方面的評論。因此，這部分也難以做到。最後，我們是採取該位YouTuber的星座來作為個人特質的劃分。如果是團體頻道的話，我們是

採取最活躍的那個成員來當作代表來去紀錄。

影片類型的部分，我們列出'travel', 'unbox', 'challenge', 'culture', 'game', 'personal', 'talk', 'education', 'music'等9項影片類別。每部影片我們使用ChatGPT來判斷影片標題，並從所列類別中選擇一個類別來去標記。得出影片類別後，我們就可以得出該頻道影片類型的分布等等。當然，用ChatGPT來標記影片類別的正確率有待商榷，因為有些影片的內容和影片標題不見得相符，有些標題甚至是使用聳動標題來吸引觀眾。然而，除非我們人工去審視每個頻道的每部影片，否則僅使用語言模型來做判斷也是人力時間限制下的權衡之計。

觀眾受眾年齡的部分，由於這部份的資訊在YouTube api中僅限於YouTuber創作者自己才能審視相關的資訊，因此我們無法取得類似的資訊。之於此，我們僅能取得該YouTuber的年齡公開資訊來加以運用。如果是團體頻道的話，我們是採取最活躍的那個成員來當作代表來去紀錄。然而，最後我們發現資料集中年齡大多介於20歲至30歲，相距不大，我們最後決定斟酌使用這個參數。

性別的部分，我們是人工去檢視標記。如果是團體頻道的話，我們就標記為第三類。即總共有男性、女性、團體這三類標籤。

有關爭議度的部分，我們是去抓取該位YouTuber相關的道歉影片標題，並從該道歉影片的發行時間判斷距今的影響，設計影響參數。然而，由於在某個時間點爭議度的影響力可以真實反應在觀看量上，所以我們不將爭議度融入建議值或是Graph的邊值中。此外，因為爭議度的特徵會隨著時間或是事件的發生而改變，且當前的爭議度也無法反應出過往的觀看量變化。除非我們選擇的模型是可以隨著時間的變化而被輸入不同的特徵向量或是產生不同的圖關係的，否則固定時間點的爭議度將無法反應出模型的輸出，因此，我們也不將爭議度納入模型。

在共同評論觀眾的部分，原本我們是想去抓取該頻道每部影片每位評論者的名稱來當作該YouTuber的受眾，最後，我們去交集兩位YouTuber的評論者名稱，就可以得出大概的共同評論觀眾。然而，由於api的爬蟲上限，再加上會評論的使用者畢竟是少數而且隨機，這部分我們實作上難以取得精準的相關資訊，因此最後我們決定放棄這一項參數來使用。

對於每一個被選入的YouTuber，我們都有給予一個獨特的id，而由於有些YouTuber有開設許多的子頻道或是相關的副頻道，我們也將那些頻道歸類入同一個id的資料中。因此，對於同一個頻道而言，其名稱的關鍵字可能有許多種。除此之外，有些頻道其組成是有許多的成員，也因此其id的關鍵字有可能有多個名稱。其他，例如:所屬組織、特定節目單元、暱稱的出現等等也都會擴充該id的關鍵字的可能性。我們在判斷某個影片是否和誰合作的時候是用所爬蟲下來的影片標題去看說其是否隱含每個id可能的關鍵字來判斷的。如果有抓到關鍵字，則標記該合作對象的id。如果有多個合作對象，則用逗號隔開區隔。如果都沒有合作對象相關的關鍵字，我們則標記為-1。值得一提的是，被我們標記合作對象為-1的影片，雖然大部分代表該影片沒有和任何其他YouTuber合作。但由於我們所建立的Graph 並非真實圖譜當中的

independent set, 因此被標記為-1有可能代表該影片還是有和其他YouTuber合作, 只是因為關鍵字沒有在所整理的set中, 因此未被抓出。我們可以簡單將有類似該種情況的影片利用程式去抓代表有合作對象的特殊關鍵字, 例如:feat, @等等, 並確認其沒有任何在資料集節點的合作對象, 最後將該影片標記為跳過, 即不將該影片納入資料集中進行運算。如此一來雖然可以部分避免真實有合作但卻被標記為沒有合作對象的情況, 但仍有些情況是如果該影片的合作對象是標記在影片中, 或是其並未標記在合作對象的特殊關鍵字, 例如:feat, @等等之後的話, 而是在之前的話, 這種情況仍然無法用程式去抓出並跳過, 僅能以人工方式確認。但礙於人力、時間的關係, 這部分我們無法完全避免。此外, 有些YouTuber目前也同時屬於某個團體的YouTube頻道, 我們會假設該團體的每部影片均出現其團體成員。即使這不一定為真, 然而, 更準確的標記需要人工檢視影片內容。最後, 有些YouTube頻道的名稱也是一般常用的單字, 例如:牛排、洋蔥等等。這有可能造成利用程式標記的時候產生誤標的情況, 必須用人工去檢視才可以去判斷真實是否有合作。

3-2 Youtuber篩選

基於YouTuber兩兩合作可能性的機率, 我們僅考慮活躍於台灣的YouTuber才會列於審視的對象。首先, 我們決定一個較常和別人合作的YouTuber當作第一個被選擇的YouTuber。接著我們去瀏覽這位YouTuber的影片中有和其他哪些YouTuber進行合作來當作之後我們選擇YouTuber的候選名單。接著, 我們再挑選第二個常和別人合作的YouTuber來當作第二個被選擇的YouTuber, 並也以同樣的方式, 再次擴增YouTuber的候選名單。

接下來, 我們從候選名單中一個個去看說那個YouTuber的影片所合作的對象中有多少是有在我們所列的候選名單或是已選擇的YouTuber中。如果太少(例如:只有1個)的話, 我們最後就決定不把那個YouTuber列入最後所選擇的YouTuber中, 否則就會列入資料集中所要探討的YouTuber。這樣可以確保最後所選取的YouTuber 節點之間的連線數不會太過於稀疏, 不加入該節點也不會對整個Graph的關係產生巨大的影響。此外, 我們也發現有些合作對象是藝人或是合作廠商, 而這些合作對象不見得自己本身就是自體活躍的YouTuber, 是否應該列入其中的節點也有待商榷。而有些YouTuber因為爭議事件的關係, 其以前的影片均已大部分刪除, 因此即使歷史上有合作過, 我們也考慮不將其加入資料集清單中。

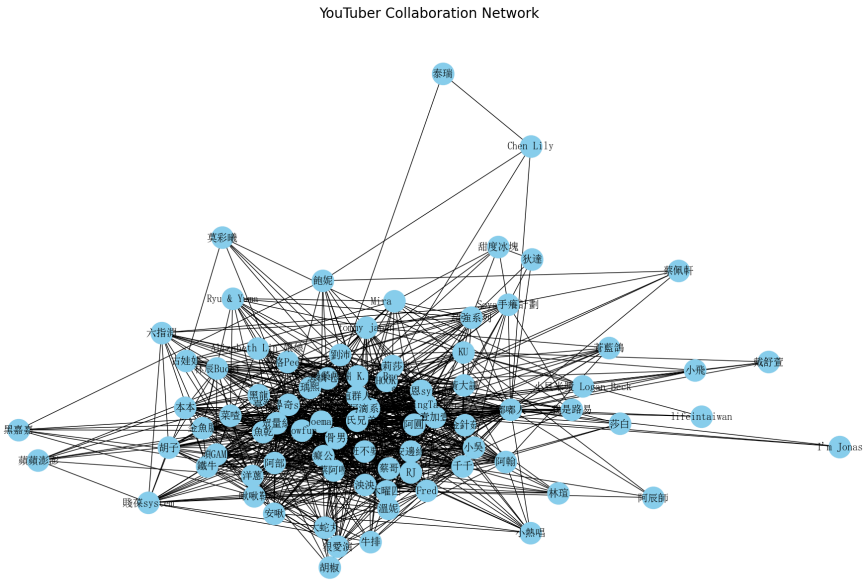
而當我們決定加入一個新的YouTuber列入資料集時, 其合作影片中勢必也有一些新的合作對象並不在我們的資料集或是可能加入的候選名單中, 但我們就不會再進一步把之加入候選名單中列為之後要去審視的對象了, 原因是因為這樣候選名單中的YouTuber審視對象有可能會一直去做擴增, 永無止盡。我們雖然可以試圖去用程式爬影片標題抓關鍵字, 例如:@、feat 後的頻道名稱等等來擴增候選名單, 期待其可以收斂, 使候選名單不再擴增, 即每一個YouTuber以及其所合作的對象均已在所列的清單中。然而, 每次抓出新的YouTuber之後, 我們都要人工去看並新增新的名單, 以及查詢其channelId, 列出YouTuber的別名等等, 這部份需要大量的時間、人力去完成。

即便如此，我們也發現到有些合作對象並不一定會列於關鍵字，例如：@、feat之後，而是直接列於影片描述的標題中。除非我們事前就知道候選名單外的YouTuber名字是什麼，否則即使影片標題有提及，我們也難以用程式抓出，擴增候選名單，除非用人工的方式一個個去審視影片標題。此外，也有許多影片當中的合作對象並不見得會列於影片標題中，而是列於影片內容中。這部份也無法去用程式去抓出，只能用人工去瀏覽影片判斷。由於不足的人力以及時間，我們並不能確實抓到所有的合作對象於我們的清單中。

基於上述原因，要把每個可能合作的YouTuber都列入考量的節點有其困難性。在有限的時間人力資源下，我們必需犧牲節點的數量及完整性。因此，我們把專案定義在如果只基於完整圖譜的某一群節點之間的關係來做兩個YouTuber是否應該合作的預測。這樣做的確會忽略、犧牲掉和我們資料集以外潛在節點的一些關係，進而影響到整個資料的預測準確率。然而，我們在建立資料集時已經盡量是將和目前已選入資料集較少交集的節點才將以排除，未做進一步審視擴增可能的候選名單。畢竟，實務上，除非我們將全台灣的YouTuber都納入考量的節點，否則我們很難確保我們資料集所列的節點剛好是一組independent set。這需要更多的人力、時間才能使我們所列入的節點不再擴增，達到收斂。

3-3 YouTuber關係圖

把各個YouTuber設為不同的節點，以視覺化圖形顯示YouTuber之間的關係。



4. 所提出之架構方法

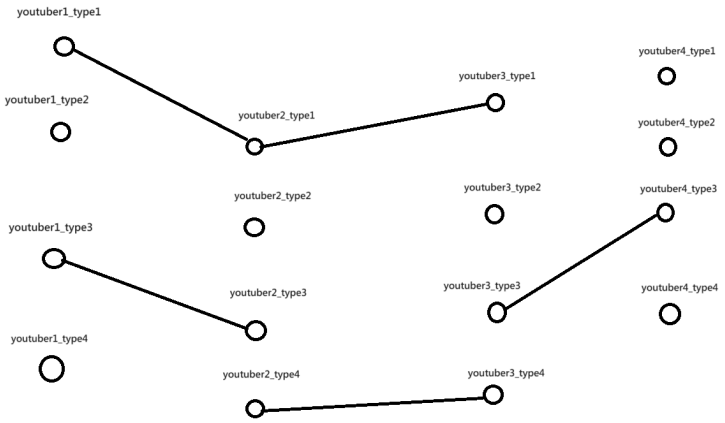
4-1 架構一

我們期中提案時所提出的架構是一種把每個節點設為某個YouTuber的某個影片類型所形成

的Graph。每個節點都會有基於那個YouTuber的某類影片類型的特徵，例如：該YouTuber的個人風格、性別等等。

在這樣的架構中，首先，同一個YouTuber的不同影片類型節點之間不會有任何連線。再者，不同YouTuber，且不同影片類型的節點之間也不會有連線，因為我們界定連線的值是基於合作影片的觀看量等等，然而該合作影片僅會有一種影片類別。因此，在這樣的架構下，僅有不同YouTuber的同一種影片類別會有互相連線的情況。

其示意圖如下:



然而，這樣的架構也造成整個Graph會有許多的independent set，使得有關link prediction的模型難以套入訓練，後續不再進行討論。

4-2 架構二

此種架構將各個不同的YouTuber設為一個個不同的節點，也把YouTuber各種可能的影片類型、性別、個人風格設為不同的節點。

每個YouTuber之間如果曾經有合作的話，彼此間就會建立有方向的連線，其值取決於頻道該影片類型的平均觀看數、影片數量、訂閱量等等。而每個YouTuber節點和影片類型節點的連線值則取決於其頻道該影片類型的平均觀看數等等。

權重設定如下：

1. YouTuber 與影片類型之間的權重：表示該影片類型的平均觀看數相對於該頻道的整體平均觀看數的比率，衡量該影片類型在該頻道中的受歡迎程度。若比率大於1，則說明該影片類型在這個頻道中比其他類型更受歡迎。
    a. 權重($W_{i,t}$)：$W_{i,t} = \frac{YouTuber_i\text{中}Type_t\text{的平均觀看數}}{YouTuber_i\text{的平均觀看數}}$
2. YouTuber 之間的權重：權重的設計考慮了訂閱量、合作影片數量、觀看表現等多個因

素，旨在全面評估兩個 YouTuber 之間的合作強度。

a. 訂閱指標($S_{i,j}$): $\frac{YouTuber_j\text{訂閱數}}{YouTuber_i\text{訂閱數}+YouTuber_j\text{訂閱數}}$

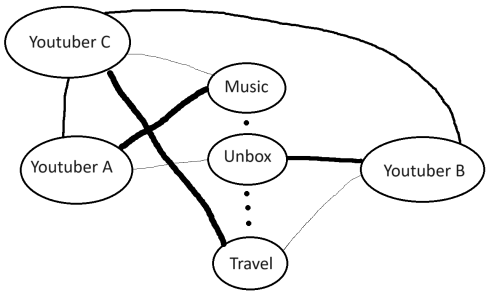
b. 合作次數($C_{i,j}$): $\frac{1}{YouTuber_i\text{頻道中}YouTuber_j\text{出現次數}}$

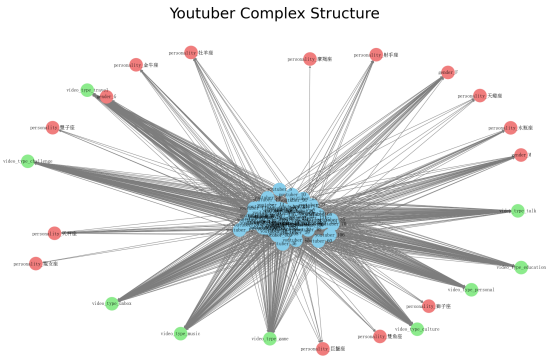
c. 觀看表現($L_{i,j,t,y}$): $\frac{YouTuber_i\text{頻道中}YouTuber_j\text{出現影片觀看量}}{YouTuber_i\text{頻道中影片}Type_t\text{於}Year_y\text{平均觀看量}}$

d. 權重($W_{i,j}$): $S_{i,j} * C_{i,j} * \sum_{YouTuber\ j\ Appear} L_{i,j,t,y} + S_{j,i}$

3. 性別、星座的節點連線：性別和星座的連線是二元的，如果屬於該類別，則連線值為1，否則不連線。用來標識 YouTuber 的基本屬性。

其示意圖如下：





其中，我們之所以只取該影片所屬年度的平均觀看量是因為YouTuber的觀看次數會隨著訂閱量的上升而有所成長。因此，我們只取和該影片時間軸較近的影片加以比較。

我們利用加權近似度演算法及圖神經網路，預測YouTuber之間的連線值：

1. 加權近似度演算法：本方法基於同質性假設，即具有相似特徵或行為的個體更傾向於互相聯繫或合作。使用加權共同鄰居、加權**Jaccard**係數、加權**Adamic/Adar**指數。這些指標在考慮節點間相似度時，不僅考慮共同鄰居的存在，也加入與這些鄰居相連的邊的權重進行考量。對於上述方法，除了Jaccard係數已經介於0至1外，我們將計算出的加權指數進行**Min-Max**標準化，將標準化後指標視作合作機率。
    a. 權重共同鄰居（Common Neighbors, CN）：此指標不僅統計兩個節點之間共同鄰居的數量，還考慮到與這些共同鄰居相連的邊的權重總和。這樣可以更準確地反映節點間的連接強度。
    b. 權重Jaccard係數：計算兩個節點共同鄰居的權重總和與所有鄰居的權重總和的比例。這種方法可以在考慮鄰居權重的同時，衡量共同鄰居在總鄰居中所占的比重。

c. 權重Adamic/Adar指數：為每個共同鄰居賦予一個與其連接的總權重成反比的權值，即越不常見的共同鄰居對相似度的貢獻越大。這反映了通過稀有連接發現的連結可能具有更高的資訊價值。

2. 圖卷積網絡（Graph Convolutional Networks, GCNs）：專門為處理圖結構數據而設計的一種深度學習模型，能夠有效地捕捉圖中節點之間的連接關係。
   a. 節點特徵的轉換：鑑於節點特徵（如YouTuber、影片類型、性別、星座）的多樣性，我們將這些特徵轉換為邊的權重，以此來捕捉各節點間的互動關係。
   b. 平衡訓練：是在進行鏈接預測的任務中，由於圖通常僅包含存在的邊（正樣本），因此需要合成不存在的邊（負樣本）來進行平衡訓練。
   c. 損失計算和優化：使用二元交叉熵損失（適用於二分類任務）計算預測輸出和真實標籤之間的損失。

4-3 架構三

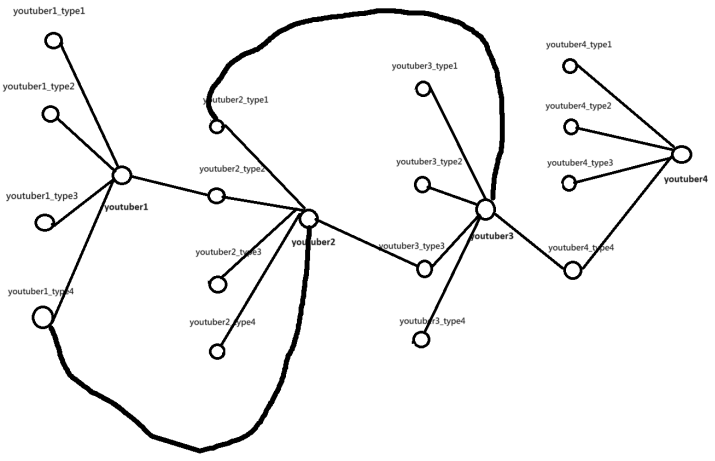
此種架構是基於架構一和架構二的改良方案。我們可以設置節點是有關某YouTuber不同影片類別的節點YouTuber_type，也可以設置單個YouTuber的節點YouTuber。每個YouTuber節點都會和自己所有的YouTuber_type有所連線。而如果你有出現在那個YouTuber的合作影片的某個類別的話，你這個YouTuber節點也會和那個YouTuber的某個影片類別有所連線。

權重公式如下：

$$W_{\text{類別 k 權重}} = \frac{\text{youtuber i 影片類型 k 的平均觀看數}}{\text{youtuber i 頻道的平均觀看數}}$$

$$W_{\text{合作權重}} = \frac{1}{\text{類別 k 中 youtuber j 出現的數量}} \times \frac{\text{youtuber j 在類別 k 中的平均觀看數}}{\text{類別 k 的平均觀看數}}$$

其示意圖如下：



每一個節點均有其特徵向量。維度中性別、影片個人風格、影片類型等均以1或是0去標記該節點是否有其對應的特徵。例如:在YouTuber這個節點中，屬於影片類型的維度裡，該頻道有幾種影片類型，那些維度就有幾個1。還有一個維度是訂閱量，為了使該維度正規化，我們定義特徵向量中屬於訂閱量的維度的值為(訂閱量/1000萬)。

我們利用近似度演算法，預測節點YouTuber i 和YouTuber j 的video type k的節點之間的連線值為何，並設定不同的標準衡量是否建立連線。基於此種架構，我們不但可以預測出兩兩YouTuber是否應該合作，也可以預測出應該合作什麼樣類型的影片。

4-4 架構四

在這個架構中，每個YouTuber會有自己的特徵向量。維度中的性別、個人風格等會依照自己有的屬性去標記。屬於為1，不屬於為0。此外，為了可以顯示和其他YouTuber之間的關係，在特徵向量中會有其他YouTuber關係的維度。曾經合作過則標記為1，不曾合作過則標記為0。最後，還有些維度是有關每種影片類型的平均觀看量、訂閱量等。為了正規化，我們會把讓平均觀看量、訂閱量去除以1000萬來當作特徵。

我們把兩兩一組的YouTuber特徵向量合併相接在一起，最後再去接上所要預測影片類別的向量。所要預測影片類別的向量總維度就是影片類別的種數，我們會在所要預測影片類別的那個維度值標記為1，其餘維度的值為0。最後，我們將合併後的向量餵進神經網路的模型中，預測出這一組YouTuber應該合作的影片類型為何。

訓練上，我們判定兩個YouTuber是否應該合作不只是看說兩個YouTuber是否曾經合作，也和其合作的觀看量是否有高於平均觀看量等等有關。因此在標記之前，我們會先去計算每個合作類別的建議值為何。如果建議值大於1，則標記為1，代表建議合作該影片類別，否則標記為0，代表不建議合作該影片類別。如果每個合作影片類別均標記為0，則表示這一組YouTuber之間不應該合作。因此，在預測結果中，我們有可能可以建議合作多種影片的類型。如果不曾合作過該種類型的影片，我們會先假定不建議合作該類型的影片，標記為0。雖然，還未曾合作過的影片類型並不絕對代表兩個YouTuber就不應該合作該影片類型，可能只是代表兩者還未找到適當的時機去合作。但是，為了方便模型評估成效，我們仍舊假設此計算方式。

如果兩個YouTuber曾經合作過某個影片類型k，則其建議值的計算公式定義如下:

影片類型的建議值(video type k) ={ [(YouTuber i的訂閱量)/(YouTuber i的訂閱量 + YouTuber j的訂閱量)] x [(1/在YouTuber j的頻道中影片類別為k的有出現YouTuber i的影片數量)] x [

$$\sum_{YouTuber\ i\ Appear\ in\ video\ type\ k\ of\ YouTuber\ j}$$

(在YouTuber j的頻道中該影片(屬於影片類別k)的觀看量/在YouTuber j 的頻道中屬於該影片年度影片類別k的平均觀看量)]}

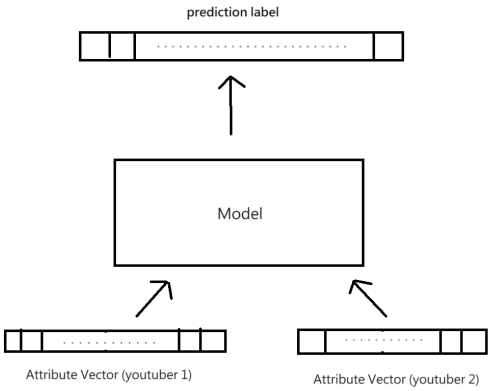
+ {[(YouTuber j的訂閱量)/(YouTuber i的訂閱量 + YouTuber j的訂閱量)] x [(1/在YouTuber i的頻道中影片類別為k的有出現YouTuber j的影片數量)] x [

$$\sum_{YouTuber\ j\ Appear\ in\ video\ type\ k\ of\ YouTuber\ i}$$

(在YouTuber i的頻道中該影片(屬於影片類別k)的觀看量/屬於該影片年度YouTuber i 頻道中該影片類別的平均觀看量)]}

註:如果兩者的合作影片僅在其中一個YouTuber的頻道中出現，則不需考慮訂閱量的加權權重。

其中，我們之所以只取該影片所屬年度的平均觀看量是因為YouTuber的觀看次數會隨著訂閱量的上升而有所成長。如果把整個頻道所有該影片類別來進行平均，那麼早期的影片觀看量和頻道所有影片的觀看平均比通常會小於1。因此，我們只取和該影片時間軸較近的影片加以比較。其示意圖如下:



基於此種架構，我們不但可以預測出兩兩YouTuber是否應該合作，也可以預測出應該合作什麼樣類型的影片。

在神經網路的使用上，我們是堆疊多層的線性模型，加上ReLU激化函數，最後用sigmoid激化函數做輸出。最後，由於資料集中存在著很大的資料不平衡，不合作的資料筆數遠遠大於合作的資料筆數，因此我們將此因素考量入損失函數的權重中。損失函數的設計如下:

$$l_{n,c} = -w_{n,c}\left[p_c y_{n,c} \cdot \log \sigma(x_{n,c}) + (1 - y_{n,c}) \cdot \log(1 - \sigma(x_{n,c}))\right]$$

其中，c為標籤類別，以架構四為例，我們僅有合作或是不合作兩種類別。yn,c是那個類別實際所代表的值，xn,c是針對實際的那個類別，模型所預測的值。而考量到資料不平衡的權重後，我們加上Wn,c作為參數。

4-5 架構五:

此架構將「預測 Youtuber 是否應該合作」的任務分成兩個子任務：

(1) Link prediction
若 Youtuber 間有合作過，即建立一條 link（沒有 **weight**）。訓練一個 link prediction 的模型，可以看成一個二分類任務（0: 沒有合作 / 1: 有合作）。

(2) Weight prediction
若 Youtuber 間有合作過，即建立一條 link（有 **weight**），並用合作影片之平均觀看量當作 weight。假設已知圖中所有 link 是否存在，訓練出一個能夠預測 weight 的模型，也就是一個數值預測的任務。

分成兩個子任務的好處是我們不僅能夠預測 Youtuber 是否應該合作，也能夠預測合作後的成功率，便可以提供預測之成功率當作最終是否要合作的考量。我們假設 Youtuber 最在意的是影片流量（即觀看數），因此選擇使用「合作影片之平均觀看量」當作 weight，也就是當作「合作成功率」的估算。

使用 **Node2Vec [6]** 取得節點嵌入（**node embeddings**）
除了使用 3-1 所提及的 YouTube API 取得的特徵，我們也使用 Node2Vec [6] 計算節點嵌入的方法來擴充我們的節點特徵維度，同時這種方式也更能有效的利用 Graph 本身的結構資訊。

Node2Vec 是一種在 Graph 中獲取節點嵌入的方法，其核心思想是透過隨機游走（random walk）來捕捉節點在圖中的結構特徵，並將這些特徵轉換為低維向量表示。Node2Vec 不考慮節點本身的標記或特徵，只考慮整張圖中節點的結構，因此可以將使用 Node2Vec 取得的節點嵌入用於各種下游任務中，而不用為每個任務重新設計與計算一次嵌入，進而節省了許多時間與運算資源。

Node2Vec 改良了原先從隨機遊走獲取節點嵌入的演算法，主要透過引入兩個參數 p 和 q 來控制隨機游走過程的靈活性：

隨機遊走策略：

- 為了平衡局部和全局結構特徵的捕捉，Node2Vec 使用 DFS 與 BFS 兩種方式來進行遊走。BFS 優先走較近的節點，而 DFS 則優先往更遠的節點移動。
- 參數 p 控制返回到前一節點的機率。如果 p 值大，遊走更傾向於不返回前一節點，因而更像 DFS；反之，若 p 值小，遊走則更傾向於返回前一節點。
- 參數 q 控制探索新節點的機率。如果 q 值大，遊走更傾向於進行局部探索（BFS）；反之，若 q 值小，遊走則更傾向於全局探索（DFS）。

演算法：

- Node2Vec 使用第二階近似隨機遊走(second-order random walk)來生成節點序列。這些節點序列類似於自然語言處理中的句子，而節點則類似於詞語。
- 每次選擇下一個節點時，根據當前節點、前一節點以及參數 p 和 q 來決定下一步的移動方向。
- 生成了大量的節點序列後，便開始進行嵌入學習(representation learning)：Node2Vec 使用類似於 Word2Vec 的 Skip-Gram 模型來學習節點嵌入。Skip-Gram 模型的目標是通過最大化相鄰節點在嵌入空間中的相似度，來學習每個節點的低維向量表示。因此在 Graph 中結構越相似的節點，其節點嵌入也會越相似。

<u>特徵向量</u>

此架構和架構四一樣賦予每個 Youtuber 一個特徵向量，但以不同的方式建構特徵向量。以下是組成特徵向量的各個元素：

```
personality_map = {
    "摩羯座": 0,
    "水瓶座": 1,
    "雙魚座": 2,
    "牡羊座": 3,
    "金牛座": 4,
    "雙子座": 5,
    "巨蟹座": 6,
    "獅子座": 7,
    "處女座": 8,
    "天秤座": 9,
    "天蠍座": 10,
    "射手座": 11
}
```

- Node2Vec 節點嵌入
  訓練 Node2Vec 模型，設定每個節點得到 100 維的節點嵌入。
- subscribers
  訂閱數 / 1000 萬
- sex
  男性 → 0, 女性 → 1, 團體 → 2
- personality
  以右圖 personality map 設定數字
- age
  因我們蒐集的 Youtuber 年齡中位數大約落在 30 上下，所以定義此特徵值為：
  (實際年齡 - 30) / 10
- 各種影片類別的平均觀看量(共 9 項)
  首先，需計算每部影片的正規化平均觀看量(**viewCountAvgInYear**)，考量到隨著時間增加，觀看量會隨著訂閱數的上升以及曝光時間長度增加而有所成長(與架構四提及的考量相同)，我們使用該 Youtuber 當年度的所有影片平均觀看量作為基準計算 viewCountAvgInYear：
  $viewCountAvgInYear$ ＝ 原始影片觀看量 / 影片所屬年度的所有影片平均觀看量

  因此若 viewCountAvgInYear > 1，則代表該影片的觀看量在所屬年度是高於平均的，大於 1 越多代表觀看數越高、影片成效越好；反之，若 viewCountAvgInYear < 1，則代表該影片的觀看量在所屬年度是低於平均的，小於 1 越多則觀看數越低、影片成效越差。

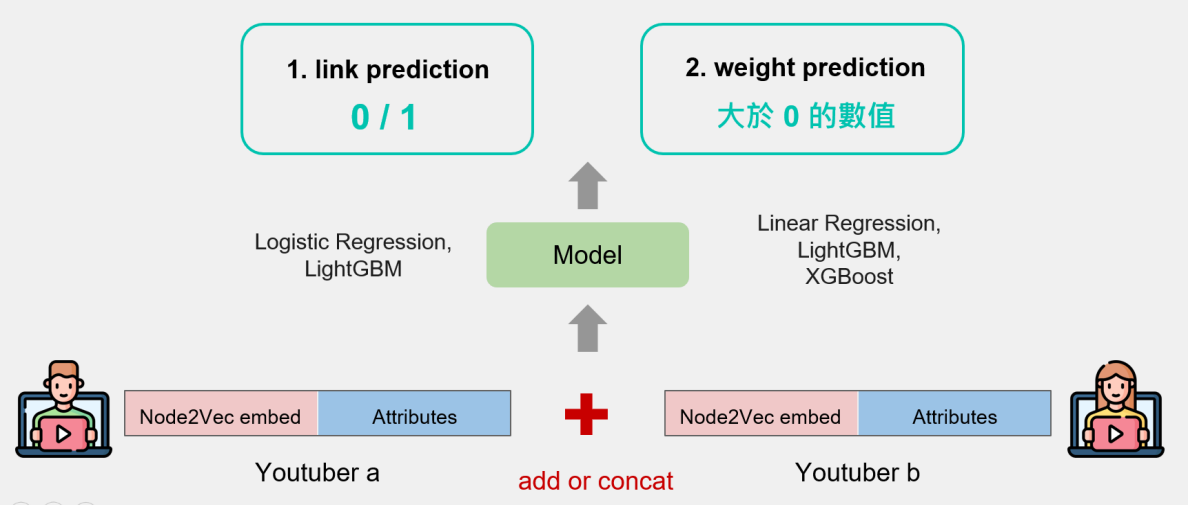
接著，為了將 Youtuber 在各種影片類別的成功度放進特徵向量中，我們對定義的 9 種影片類別（travel, unbox, challenge, culture, game, personal, talk, education, music）皆計算該類別的正規化平均觀看量：

$$\text{類別 } k \text{ 的正規化平均觀看量} = \frac{\sum_{\text{video } i \in \text{ type } k} viewCountAvgYear(i)}{|N|},\ |N| = number\ of\ videos\ belong\ to\ type\ k$$

因此，Node2Vec embedding 有 100 維，由我們爬取的資料獲得的特徵有 13 維，下段將會敘述我們如何使用這些特徵向量來訓練模型。

模型架構

不論是 (1) link prediction 或 (2) weight prediction 皆是對一組 Youtuber pair 做預測。因此我們將兩個 Youtuber 的特徵向量以相加或連接（**concat**）的方式結合，接著將融合後的向量丟進模型中做預測。各任務所使用之模型分別標示於下圖 "Model" 的左右兩邊。在實驗結果的段落中將會比較不同特徵融合方式以及不同模型的成效。



5. 實驗方式及結果呈現

5-1訓練資料與測試資料

我們首先設定過去某一個時間點為時間標準線，並找出那個時間標準線的時候兩兩已經有合作過的YouTuber組合，以及兩兩還沒合作過的YouTuber組合。這個時間點之後必須要有些合作組合是時間點之前沒有，之後才出現的。因此，這個時間點不能設定為目前資料集所蒐集到最新的時間點。原因是因為對於至今為止還沒合作過的組合，我們無從驗證模型是否有效。

我們以時間標準線以前各組合的YouTuber特徵、節點來當作訓練的資料集。其中，訓練的組合僅包括所設置時間點以前已經有合作過的YouTuber。然而，測試資料只針對所設置時間點之前都尚未合作過的YouTuber組合或是影片類型。從所設置的時間點到目前的時間點之間，

可能會有些新的合作組合是時間點之前還沒有出現的，但也有些組合是到目前為止都還沒合作過的。我們針對這些測試資料做預測，並比對所設置時間點之後到目前的時間點中間所預測和實際新的合作組合是否相符。

我們選取 2021/12/18 作為時間標準線，因為他能將訓練資料與測試資料以大約 8 : 2 的比例切分，此外，我們也測試過，若將這組測試資料中的 link 從圖中移除，也不會造成圖中出現孤立的點或是 connected_components > 1 的情形，意即，圖中所有節點仍然可互相連通，因此此時間點是較為合適的切分線。

5-2評估方式

我們針對時間點之前都尚未出現的合作組合做出預測，並繪出以下的實驗表格。

(時間點以後)真實中\預測	仍沒有合作	才開始有合作
仍沒有合作	A	B
才開始有合作	C	D

我們以模型的準確率(Accuracy)、召回率(Recall)來評估模型的好壞。公式如下:

Accuracy = (A+D) /(A+B+C+D)
Recall = D/(C+D)
F1 score = (2*D)/(2*D + B +C)

其中，recall的部分我們在意的是實際上從時間點到目前為止才出現的合作組合是否都有被偵測到。我們較不在意實際上仍沒有合作的組合被預測標記為有開始合作，原因是因為未來也許某些之前仍未合作的組合真的有可能合作，只是還沒發生。

然而，我們發現由於資料集中未合作的資料遠遠大於有合作的資料，因此在模型預測上，很容易造成模型訓練成全都預測為連線或是不連線，以取得似乎是好的accuracy或是recall。也就是說，如果模型全部預測為不連線，則可以取得較高的accuracy，卻不能取得很好的recall。反之，如果全部預測為連線，則可以取得較高的recall，卻不能取得較高的accuracy。因此，為了避免這種狀況，我們評估模型的好壞不偏重任何一個評估指標，而是希望整體來說兩項指標都能取得相對好的數值。

5-3實驗結果以及分析

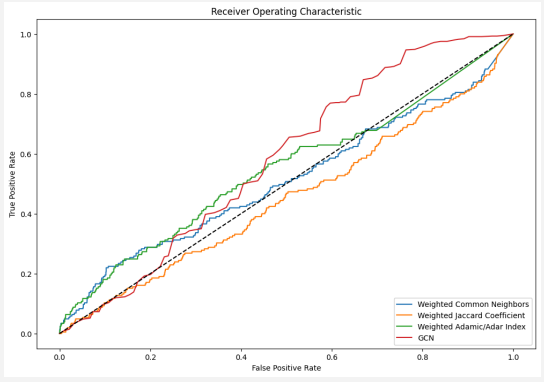
5-3-1 架構一

由於模型的架構會產生許多的 independent set，故未對此進行實驗。

5-3-2 架構二

針對架構二不同的指標與模型，其實驗結果數據如下：

Metric	Weighted Common Neighbors (Scaled)	Weighted Jaccard Coefficient	Weighted Jaccard Coefficient (Scaled)	Weighted Adamic/Adar Index (Scaled)	GCN
Accuracy	0.93	0.87	0.92	0.94	0.54
Precision	0.08	0.04	0.05	0.13	0.54
Recall	0.07	0.10	0.05	0.06	0.51
F1 Score	0.07	0.06	0.05	0.08	0.52
AUC	0.51	0.45	0.45	0.55	0.58
Predicted Positive	182	520	212	102	533



其中，我們可以看到在加權相似度指標中，準確度與其他指標(如精確度和召回率)之間存在顯著差異。這種差距表明在模型預測中存在大量的偽陽性和漏報情況，這通常表示模型未能有效預測到未來的連線。儘管準確度較高通常意味著大多數預測是正確的，但這可能是由於數據集中負例(無連接)遠多於正例(有連接)。因此，即使模型大部分時間預測“無連接”，它仍然可以達到高準確度。然而，當我們看到精確度和召回率較低時，這暗示模型在實際預測正例(即預測存在連接)時效果不佳。相對於加權相似度指標，GCN模型的實驗結果較為穩定，在各項指標中皆高於其它三個方法，代表GCN模型的平衡訓練能夠顯著降低漏報與偽陽性的發生。

不過，此種架構僅能預測出兩個YouTuber之間是否應該合作，並不能預測出兩者應該合作何種類型的影片。可能需要進一步的調整或結合其他策略，從而更有效地預測未來的連線。

5-3-3 架構三

針對架構三，由於我們是透過Graph，針對node間去預測是否會有連線，因此可能的連線結果會出現以下四種情形：

1. Youtuber_a 與 Youtuber_b 間的連線
2. Youtuber_a_topic 與 Youtuber_b 間的連線
3. Youtuber_a 與 Youtuber_b_topic 間的連線
4. Youtuber_a_topic 與 Youtuber_b_topic 間的連線

其中針對第1項，由於我們想知道Youtuber 間會不會針對某個主題去合作，因此第1項僅知道Youtuber間會有合作，但無法得知是哪些主題，所以在計算結果時，不考慮第1項是否有連線。

而第4項雖然是Youtuber的topic 間的連線，但我們仍可以視為有連線，並且因為是沒有方向性的Link，所以如果其中一方不考慮其topic，則可以將結果切分為（Youtuber_a_topic, Youtuber_b）與（Youtuber_a, Youtuber_b_topic），兩種連線及果，透過這種方法，可以增加我們所預測出來的連線數，並且也可以避免在2、3中，可能因為資料稀少導致預測出來的連線僅有少數幾條的情況，並根據以下情況判斷是否有連線：

衡量方法	判斷有連線	判斷無連線
Weighted Jaccard Coefficient	value >= 0.1	value < 0.1
Weighted Common Neighbors	value >= 5	value < 5
Weighted Adamic/Adar Index	value >= 1	value < 1

針對「**YouTuber**是否合作」結果如下：

	Weighted Jaccard coefficient	Weighted Common Neighbors	Weighted Adamic/Adar index
Precision	0.31	0.34	0.32
Recall	0.14	0.05	0.07
F1 Score	0.19	0.08	0.11
AUC	0.55	0.52	0.53

針對「類別是否合作結果」如下：

	Weighted Jaccard coefficient	Weighted Common Neighbors	Weighted Adamic/Adar index
Precision	0.13	0.15	0.13
Recall	0.26	0.09	0.12
F1 Score	0.17	0.11	0.12
AUC	0.61	0.54	0.55

三種模型的precision表現相近，但透過Jaccard Coefficicent可以得到較高的recall 與 AUC。

5-3-4 架構四

架構四所得出的數據結果如下：

	兩兩是否合作某影片類型	兩兩是否合作
Accuracy	0.67	0.15
Recall	0.65	0.88
AUC	0.66	0.49
F1 score	0.79	0.16

我們可以觀察到如果我們把目標設定在合作的影片類別是否預測正確的話，整體來說相對於隨機預測來講，accuracy和recall均可以達到相對不錯的結果。然而，如果把目標設定在僅預測兩兩是否合作的話，accuracy並沒有得到好的結果，而recall卻達到很好的結果。有這樣差別的原因可能在於基於類別的資料標記為1和0的分布和基於兩兩是否合作的資料標記為1和0的分布是不一樣的。因此在設定資料不平衡的權重時也應該要不一樣。

如果僅關注在recall的評估的話，即實際上有合作的關係中是否都有被預測到，我們可以發現架構四的模型有一個不錯的結果。針對類別的預測有64.9%，針對兩兩YouTuber是否合作的預測有88.2%的召回率。

5-3-5 架構五

針對架構五，以下是不同特徵融合方式以及不同模型在測試集所取得的分數。
1. 與 2. 只使用 Node2Vec 所得的 embedding（100 維），3. 與 4. 只使用我們自己爬取與標註的資料（13 維），5. 與 6. 則兩者皆使用。

藍色底標示出特徵向量融合的方式，綠色底標示出相同方法下（縱向比較）成效較好的模型。

(1) Link prediction

（表一）不分時間切分 train, test 的 **ROC-AUC score**

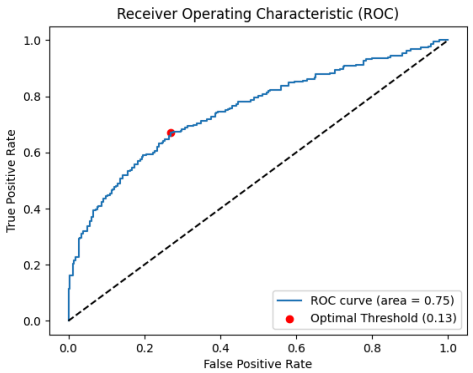
	1. Node2Vec (Add)	2. Node2Vec (Concat)	3. Our features (Add)	4. Our features (Concat)	5. Mix (Add)	6. Mix (Concat)
Logistic Regression	0.79	0.80	0.77	0.78	0.81	0.830
LightGBM	0.88	0.91	0.87	0.90	0.90	**0.91**

（表二）以時間標準線切分 train, test 的 **ROC-AUC score**

	1. Node2Vec (Add)	2. Node2Vec (Concat)	3. Our features (Add)	4. Our features (Concat)	5. Mix (Add)	6. Mix (Concat)
Logistic Regression	0.63	0.53	0.66	0.61	0.68	0.58
LightGBM	0.72	0.65	0.70	0.68	**0.75**	0.66

由上表的數據顯示，LightGBM 是最 robust 的模型。而只使用 Node2Vec embedding 和只使用我們爬取的特徵所獲得的成效差不多，若兩者都使用則可以提升預測的準確度。雖然在表一中 Mix (Concat) 的分數略高於 Mix (Add)，但在較困難且接近真實世界的以時間標準線切分的測試集中反而是 Mix (Add) 取得更好的分數，且高過 Mix (Concat) 的比例高出許多，因此我們認為在 link prediction 中最好的方法組合為 **Mix (Add) + LightGBM**。

下圖是使用最佳組合 **Mix (Add) + LightGBM** 測試在「以時間標準線切分」的測試集所得的 **ROC Curve**:



使用 **Optimal threshold** 的計算結果	
Accuracy	0.70
Recall	0.67

(2) Weight prediction

（表三）不分時間切分 train, test 的 **RMSE loss**

	1. Node2Vec (Add)	2. Node2Vec (Concat)	3. Our features (Add)	4. Our features (Concat)	5. Mix (Add)	6. Mix (Concat)

Linear Regression	0.79	0.89 (震盪很大, 也有出現過 18.多)	0.79	0.81	0.79	FAILED
XGBoost	0.79	0.77	0.76	0.76	0.77	0.79
LightGBM	0.78	0.76	0.78	0.76	**0.75**	0.77

（表四）以時間標準線切分 train, test 的 **RMSE loss**

	1. Node2Vec (Add)	2. Node2Vec (Concat)	3. Our features (Add)	4. Our features (Concat)	5. Mix (Add)	6. Mix (Concat)
Linear Regression	1.35	FAILED	1.10	1.09	FAILED	FAILED
XGBoost	1.13	1.09	1.13	1.10	1.12	1.11
LightGBM	1.11	1.08	1.09	1.06	1.09	**1.06**

由上表的數據顯示，雖然在表三中 XGBoost 有兩項贏過 LightGBM，但在較困難且接近真實世界的以時間標準線切分的測試集中 LightGBM 全部贏過其他兩個模型，因此 LightGBM 仍舊是最 robust 的模型。而 weight prediction 的最佳組合為 **Mix (Concat) + LightGBM**，次佳的則是 Our feature (Concat) + LightGBM。

另外，我們觀察到 Linear regression 在特徵向量維度比較大的時候（ex. 有加 Node2Vec embedding 且用 concat 時）會非常不穩定，RMSE loss 可能會到幾千萬，因此幾乎無法使用，我們在表中標示為 FAILED。

5-3-6 綜合比較（同任務間使用不同架構的結果分析）

各架構間著重比較 **Recall** 與 **ROC-AUC**

- Recall
  - 優點：可以判斷有連線的是否都有被找到
  - 缺點：只看這項數據可能導致模型直接全部預測有合作
- ROC-AUC
  - 優點：解決 recall 的問題，也不會受到 threshold 選擇的影響
  - 缺點：沒有合作不代表不適合合作，可能只是在目前資料蒐集範圍中還沒有合作，未來仍可能合作。此外，由於不合作和合作的資料量分布是不平均的，也因此如果模型均預測不合作，也有可能得到較好的數據。

任務：預測是否合作

	架構二	架構三	架構四	架構五
Recall	0.51	0.26	**0.88**	0.67

ROC-AUC	0.58	0.61	0.49	**0.75**

- 若考量兩種指標，架構五是最好的方法，因為它達到次高的 recall，而且有最高的 AUC，而各指標的分數都算是可以接受
- 若只考量 recall，代表我們希望能成功預測出越多合作組合越好，那就可以選擇架構四，但可能會有較多誤報，因其 AUC 不到 0.5
- 若只考量 AUC，架構五是最好的方法，次高為架構三，而架構四的 AUC 顯示其判別正負樣本的能力與隨機猜測差不多

任務：預測合作影片類型

	架構三	架構四
Recall	0.14	**0.65**
ROC-AUC	0.55	**0.66**

- 架構四是最好的選擇，不論在哪個指標都取得了最高的分數，而且 AUC = 0.66 顯示其有區分正負樣本的能力，不是隨機分類
- 由上表可以看出，預測合作影片類型相較於只預測合作對象難上許多，因此分數都沒辦法到太高，但架構四的預測結果仍可提供參考

6.總結以及未來展望

在特徵設計的部分，如果有更精細的資料權限，我們也許可以做出更好的預測。例如:由於未能取得使用者受眾年齡，因此最後我們並未採用相關特徵。

有關影片個人風格的部分，使用星座來當作代表也未盡理想。例如: 我們可以去設影片風格為可愛型、知識型、搞笑型、宅男型、挑戰型等，並依依用人工的方式去瀏覽影片，判斷每個YouTuber有哪些個人特質。而這些特質是可以同時擁有的。然而，這樣的標記需要人工去作業，由於學期間時間、人力的限制尚未能完成。

有關影片類型的部分，除了使用ChatGPT來判讀影片標題之外，是否有可以更精確判別的方式也是我們可以再加思考的。例如:有討論到的有些頻道名稱和一般常用的單詞是共用的，造成誤解的產生。此外，影片類型的決定是否可以擴大為更多的類型，使類型可以更加精細的反應影片標題也是可以著墨的地方。

除此之外，在YouTuber節點的選取上，由於時間、人力的關係，我們尚未能選擇足夠的節點使

得所選的節點集合在真實的Graph中屬於independent set。這使得有些合作對象明明在影片中，卻未被標記。或是團隊頻道中，其組員是否真實沒出現，卻被標記為有出現等等，也是未來有時間可以繼續努力的。而有關是否應當刪除掉連線數過少的節點、藝人或是爭議度極高的節點則可以再加思考。

在實驗模型的部分，我們針對不同的架構進行了詳細的測試與比較，在各任務下皆能找到一個較為合適的模型去使用。然而各個架構在不同評估指標下的表現差異顯著，這表示模型仍有很多進步空間。除了結合更多 Youtuber 特徵（例如：觀眾留言、影片逐字稿分析），我們也可以研究能力更強的圖神經網路模型以及更有效的特徵融合方法，同時，進行資料擴增（data augmentation）以解決資料標記不平衡的問題，都可以提升模型的泛化能力和預測效能。

7. 分工

葉冠宏:口頭簡報及製作、書面報告、架構四、資料集蒐集

魏靖軒:口頭簡報及製作、書面報告、架構三

江子涵:口頭簡報及製作、書面報告、架構五

許聖慧:口頭簡報及製作、書面報告、架構二

8. 參考資料

[1] Liu, G. (2021). An ecommerce recommendation algorithm based on link prediction. *Alexandria Engineering Journal*, 61(1), 905-910. https://doi.org/10.1016/j.aej.2021.04.081
[2] Antonio Ferrara, Lisette Espin-Noboa, Fariba Karimi, and Claudia Wagner. 2022. Link recommendations: Their impact on network structure and minorities. *Proceedings of the 14th ACM Web Science Conference 2022 (WebSci '22)*. Association for Computing Machinery, New York, NY, USA, 228–238. https://doi.org/10.1145/3501247.3531583
[3] N. Kuang, Y. Zuo, Y. Huo, L. Jiao, X. Gong and Y. Yang, "Network Link Connectivity Prediction Based on GCN and Differentiable Pooling Model," *2022 IEEE 14th International Conference on Advanced Infocomm Technology (ICAIT)*, Chongqing, China, 2022, pp. 1-6, doi: 10.1109/ICAIT56197.2022.9862715.
[4] Liben-Nowell, D., & Kleinberg, J. (2007). The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, 58(7), 1019-1031. https://doi.org/10.1002/asi.20591
[5] Lü, L., & Zhou, T. (2011). Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, 390(6), 1150-1170. https://doi.org/10.1016/j.physa.2010.11.027

[6] Grover, Aditya; Leskovec, Jure (2016). "Node2vec". *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* Vol. 2016. pp. 855–864.