# דוח פרוייקט סיכום קורס ניתוח רשתות ביולגיות

לכתוב את השמות שלנו

# בהנחיית: פרופ' רודד שרן.

# מבוא

ניתוח רשתות ביולוגיות הוא תחום מתפתח שחוקר תהליכים ביולוגיים מורכבים בעזרת ייצוגם כרשתות של אינטראקציות בין מולקולות. ניתוח הרשתות המתקבלות מציב אתגרים המשלבים תורת הגרפים ולמידה חישובית עם יישומים חשובים בביולוגיה ורפואה. הקורס סוקר אלגוריתמים ושיטות ניתוח עדכניות בתחום כמו גם את יישומיהם למידע ביולוגי בקנה מידה גדול.

אחת מהבעיות שנחקרו בתחום מציאת המודולים ברשתות זה היא מציאת קומפלקסים חלבוניים. קומפלקס הינו קבוצה של שני חלבונים או יותר שביחד מהווים מכונה מולקולרית (אולי להסביר מה זה). את בעיה זו ניתן למדל ע"י מציאת תת-גרף צפוף מתוך רשת של אינטראקציות חלבון-חלבון. למרות שבעולם האמיתי חלבונים יכולים להשתתף במספר קומפלקסיים וכן קיימים חלבונים שלא משתתפים באף קומפלקס לרוב נשתמש בשיטות קליסטור שמתאימות לכל חלבון קלאסטר יחיד.

בעבודה זו נשווה בין מספר שיטות קימות למציאת קומפלקסים וכמו כן, נציע שיטה חדשה המבוססת על קליסטור נקודות במרחבים מטריים. לשם כך נמיר את רשת האינטרציות אל ווקטורים ב בעזרת כלים של למידה ועליהם נריץ אלגורמי קליסטור.

נשווה בין השיטות הבאות:

1. MCL
2. Clusterone
3. Our method-protein embedding

כאשר השיטה השלישית מבוססת על מרחק מטרי.

# Clustering

"קלסטרינג" הינו טכניקה בלמידה חישובית אשר כולל קיבוץ נקודות מידע. בהינתן קבוצה של נקודות מידע, אנחנו יכול להשתמש באלגוריתם קלסטרינג על מנת לתייג כל נקודת מידע נתונה לקבוצה ספציפית המתאימה לה.

בתאוריה, נקודות אשר נמצאות באותה קבוצה צריכות לחלוק תכונות משותפות ואייפיונים דומים, בעוד שנקודות מקבוצות שונות צריכות להיות שונות מאוד בתכונותיהן ובאיפיוניהן.

שיטת הקלסטרינג משויכת ללמידה לא מפוקחת, והינה טכניקה נפוצה מאוד בקרב זיהוי ניתוח מידע סטיסטי בהמון שדות מחקר.

ולכן בתחום הביולוגיה החישובית יש צורך ממשי לזיהוי תכונות דומות של גנים וחלבונים, ומציאת אינטרקציות. כמות המידע אדירה אשר נחשפים אליה הינה גדולה מאוד וברובה המוחלט ללא תיוג.

מכך אנחנו מגיעים לאלגוריתמים של למידה לא מפוקחת ובפרט לקלסטרים.

עם מעט המידע הקיים לנו, קלסטרינג יכול להעניק כח גדול בתיוג המידע החסר.

# תיאור האלגוריתמים

## אלגוריתם קליסטור מרקובי (MCL)

באלגוריתם זה נתייחס לגרף כעל רשת מרקובית כאשר הסתברויות המעברים הן אחידות מבין השכנים של כל חלבון (או לפי הגודל היחסי של משקלי הקשתות). נוכל להגדיר קלאסטר כקבוצת חלבונים שכשאר מבצעים הילוך מקרי מאחד מהם נקבל הסתברות גבוה לסיים באחד מהחלבונים בקבוצה ולכן נוכל לחלק את הגרף לקלאסטרים על ידי הדמייה של הילוך מקרי.

כדי למצוא זאת באופן יעיל נחזור על שני הפעולות הבאות על מטריצת המעברים עד להתכנסות:

1. Expansion- העאלת המטריצה בחזקת כאשר - פעולה זו מדמה הילוך מקרי של צעדים
2. Inflation- העאלת כל איבר בחזקה ונרמול העמודות – באופן דומה לsoftmax פעולה זו מגדילה בכל עמודה את הערכים הגבוהים על חשבון הערכים הנמוכים ולמעשה "מחזקת" את הקלאסטרים הקיימים.

Chart

Description automatically generated

נוסחא המתארת את התהליך הינה:

כאשר האלגוריתם מתכנס נקבל מטריצת מעברים כאשר הצמתים מחולקים לקבוצות (קלאסטרים) כך שמכל צומת ניתן להגיע רק לצמתים אחרים באותו הקלאסטר ונוכל להתחייס לכל קלאסטר כזה כעל קומפלקס.

## Clusterone

ClusterONE (Clustering with Overlapping Neighborhood Expansion)

הינו אלגוריתם קלסטרינג לגרפים, אשר מתמודד גם עם גרפים ממושקלים ומייצר קלסטרים בעלי חפיפה ללא קשיים.

שני תכונות אלו שימושיות מאוד בזיהוי קומפלקסים של רשתי אינטרקציות חלבון-חלבון.

ציטוט מן המאמר המתאר את שלבי האלגוריתם:

" Our algorithm consists of three major steps (Online Methods). First, starting from a single seed vertex, a greedy procedure adds or removes vertices to find groups with high cohesiveness. The growth process is repeated from different seeds to form multiple, possibly overlapping groups. Although some overlaps are likely to have biological importance, groups overlapping to a very high extent in comparison to their sizes should likely be merged. In the second step, we quantify the extent of overlap between each pair of groups and merge those for which the overlap score[4](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3543700/#R4) is above a specified threshold. In the third step, we discard complex candidates that contain less than three proteins or whose density is below a given threshold. Note that our method potentially can be used to recognize not only partial overlaps but also cases in which a complex is completely contained in another complex."

## Protein Embedding

השיטה המוצעת בעבודה זו היא לנצל אלגוריתמים ידועים לקליסטור. הקלט לאלגוריתמים אלו צריך להיות נקודות בתוך מרחב מטרי (כלומר שקיימת פונקציה שמודדת מרחקים בין האובייקטים\קבוצות של אובייקטים). בבעית מציאת המודולים הקלט הוא גרף אינטראקציות ללא מטריקה טבעית מלבד אורך המסלול הקצר ביותר. מטריקה זו מאבדת את מידע על הקישוריות בגרף ולטעמנו אינה מספיקה. לכן, על מנת שנוכל לנצל בצורה מיטבית את מלוא המידע שנתון ברשת הטמענו את הרשת ב בעזרת כלים של למידה עמוקה.

להוסיף ציור שמסביר למה זה לא מטריקה טובה

פעפוע זה לא מטריקה?

התנסינו בלמידה בעזרת רשת סיאמית אך זמן הריצה היה ארוך מידיי (ריבועי במספר החלבונים) ולכן השתמשנו בnode2vec. לאחר ההטמעה השתמשנו בk-means

### Node2vec

שיטה זו משמשת להטמעה של רשתות במרחב באופן שמשמר תכונות של הרשת המקורית. נרצה שהמרחק של הצמתים המוטמעים קטן אם הצמתים קרובים ברשת. מכיוון שזה לא פיזבלי לאמן על הסביבה של כל צומת נגריל הילוך מקרי באורך קבוע מכל צומת (כאשר את הסתברויות המעברים קובעים בעזרת היפר פרמטרים – פרמטרים אלו קובעים את אופי ההליכה ולכן אילו אספקטים של הרשת נלמד)

לשים פה את הציור מהמאמר

כדי ללמוד את ההטמעה- לכל צומת מאתחלים ווקטור רנדומלי ב, מתחייסים לכל הילוך מקרי מצומת כלשהו כדגימה מהסביבה שלו (בדומה למשפטים ב word2vec) ונמזער את המרחק בין הצומת לסביבה שלו ובסוף התהליך נפלוט את הווקטורים לכל הצמתים.

לשים פה את פונקצית הלוס מהמאמר

### k-means

אלגוריתם מבוסס prototype לקליסטור של נקודות.

האלגוריתם מחלק את הקבוצה ל-K קלסטרים זרים, כל קלסטר מיוצג ע"י "הטיפוס" שלו- כאשר שייכות של ווקטור לקלאסטר נקבע ע"פ הטיפוס הקרוב ביותר.

כדי למצוא את המרכזים מאותחלים הטיפוסים באופן רנדומי ואז באופן איטרטיבי עד להתכנסות:

* משייכים כל דגימה לטיפוס הקרוב ביותר אליה
* מעדכנים את הטיפוסים להיות המרכז של כל הדגימות המשויכות אליו.

האלגוריתם דורש שיספקו לו את מספר הקלסטרים "הרצויים". הוא מותאם היטב לעבודה עם כמות אדירה של מידע, ובשל כך יש בו שימוש נרחב.

אלגוריתם ה- K-means שקול לבחירה של הטיפוסים כך שנמזער את הפונקציה הבאה:

A picture containing text

Description automatically generated

# תהליך העבודה

על מנת להשוות בין השיטות השתמשנו במאגר המידע GO. כאשר סיננו outliers ע"י לקיחת 3-core של הרשת.

תהליך העבודה כלל הגדרת ציון לpartition על מנת שנוכל להשוות בין האלגוריתמים, מימוש של האלגוריתמים (כאשר לclusterone השתמשנו במימוש קיים) וכיוון של היפר-הפרמטרים כדי להשיג תוצאות מיטביות.

## הגדרת הציון

נשווה בין שלושת השיטות ע"י חישוב של העשרה פונקציונאלית בהתבסס על התפלגות היפר גאומטרית. נחשב לכל קלאסטר את הציון כאשר לקלאסטרים עם פחות משלושה חלבונים ניתן ציון 0. ע"מ לקבל מספר בודד נייחס לכל חלבון את הציון של הקלאסטר שלו ונמצע את הציונים.

## מימוש האלגוריתמים

ניתן למצוא את המימושים שלנו [בגיטהאב](https://github.com/haimsaw/YeastEmbedding)

כאשר המימושים לאגוריתמים נמצאים בקבצים עם השמות המתאימים (בנוסף למספר קבצי עזר)

את השיטה שלנו הרצנו על גבי [google collab](https://colab.research.google.com/drive/1bWiEjjkkv4leswz_wqLL2Ryxqg3AUn2U?usp=sharing) עם GPU

## כיוון היפר-פרמטרים

בשיטת ה Protein Embeddingישנם מספר רב של היפר-פרמטרים שנדרש לכוון בנינו תשתית שמאפשר להריץ מספר טסטים ולבחון את ההשפעה של הפרמטרים על הציון.

הרצנו ניסויים כאשר מכל צומת הגרלנו 10 מסלולים באורך 20.

הפרמטר הראשון שנדרש לכיוון הוא מספר מחזורי הלימוד. בגרף הבא ניתן לראות את הloss של ההטמעה עבור כל epoch:

Histogram

Description automatically generated

ניתן לראות כי לאחר 20 איטרציות הגרף מתיישר – כלומר השינויים בהטמעה הם זניחים ומספיק להריץ 20 איטרציות.

לאחר מכן בחנו את השפעה של (מימד ההטמעה והסתברויות ההילוך המקרי) על ההעשרה הפונקציונאלית:

A picture containing square

Description automatically generatedSquare

Description automatically generated with medium confidenceLogo

Description automatically generated with medium confidence

ניתן לראות כי למספר המימדים יש השפעה על הציון וכן יש נטייה ל נמוך ו גבוה (מה שתואם את האינטואיציה – נרצה לחקור את הסביבה של הצומת ולכן לא נרצה לבקר באותה הצומת פעמיים, כמו כן נרצה שההילוך המקרה יידמה bfs וישמר homophily)

## תוצאות

### עבור ה- MCL:

cluster num of clusters : 404

cluster partitionScore : 10.718850527266893

cluster meanSize : 12.779702970297029

cluster maxSize : 386

cluster minSize: 5

### עבור ClusterONE:

cluster num of clusters : 366

cluster partitionScore : 1.080335547273504

cluster meanSize : 3.8934426229508197

cluster maxSize : 48

cluster minSize: 3

במקביל ובאופן בלתי תלוי, מימשנו את האלגוריתם שלנו.

את האלגוריתם שלנו הרצנו עם היפר פרמטרים משתנים אשר ביצענו בעזרתם חקירה של איך המידע מסודר ואיך למנוע בעיות כמו overfit וכדומה.

בדוגמה הראשונה ניתן לראות ש-

ולכן אנחנו מסיקים כי XYZ

לאחר שלמדנו ש XYZ מתקיים אנחנו רוצים לשפר את היכולת הבאה ומכך ניתן לראות בדוגמה השניה שיפור בזה

ובנוסף ניתן לראות תופעה של Overfit עקב עודף נקודות בסיס לאלגוריתם ה-Kmeans.

לכן לאחר צמצום אנחנו מקבלים שיפור נוסף:

כעת על מנת להקטין במקביל את הכמות שכל קלסטר מקבל וכן להגדיל ככל הניתן את המינימום איברים בכל קלסטר אנו משנים את אורך ההילוך / ***משהו שחיים צריך להגיד***

ואנו מקבלים את התוצאה הבאה:

ולכן סהכ אנחנו מקבלים שהתוצאה הטובה ביותר שקיבלנו עבור k-mean הינה:

## שאלות פתוחות

השאלה הפתוחה העיקרית שעולה מעבודה זו היא כיצד ניתן להימנע מoverfit. ע"י שימוש בכח מחשוב חזק יותר ניתן היה לדגום יותר הילוכים מקריים מכל צומת וכן הילוכים מקריים ארוכים יותר שיספקו יותר מידע על מבנה הרשת.

כמו כן, מדד ההעשרה הפונקציונאלית ייצר הטייה עבור יצירת קלאסטרים מועשרים ללא התייחסות לקישוריות שלהם ברשת המקורית. שימוש במדדים אחרים לעמידה של איכות הקלאסטרים (לדוגמה ) יכולה לנטרל את ההטייה הזו ולאפשר לנו לכוון את הפרמטרים בצורה מיטבית.

לסיום, למרות שתנסינו באלגוריתמי קלאסטרינג נוספים (Affinity Propagation, DBSCAN) מכיוון שההטמעה הייתה מוטה הם הניבו תוצאות לא טובות. פתרון של בעיית ה overfitting עשוי לאפשר תוצאות טובות יותר.