אנחנו רוצים לדעת מחירים של בתים.

* השאלה: שימוש ב-GNN – האם הוא עוזר או לא?

יש לנו 6 דאטא סטס, שכל אחד מייצג מידע ממיקום מסוים. הדאטא שלנו גם די מגוון – מגיע מכל מיני מיקומים שונים, כלומר ניתן לסמוך עליו שלא יבצע אובר פיטינג.

הרעיון של מחירים של בתים שזה דאטא שהוא לא טבלאי – כלומר, הנתונים קשורים זה לזה. אם אני נמצאת בשכונה כלשהי, המחיר של שאר הבתים שם משפיע גם על המחיר שלי. דברים כמו בתי ספר, רעשים וכו' הם חשובים. המידע שלי **תלוי** בשאר המידע.

* הקדמה: GNN הוא ייצוג גרפי. זאת בניגוד לשאר השיטות שהשתמשנו בהן, שמביאות לנו נתונים מוזרים כשזה מגיע למידע גיאוגרפי. עצים לא תמיד עוזרים לי, כי הם חושבים על הדאטא בצורה בלתי תלויה, וכפי שאמרנו, הדאטא שלנו תלוי.

ב-GNN כל בית הוא צומת והקשתות מתארות את הקשרים בין הבתים, נניח המרחק ביניהם. ככה אנחנו בונים גרף שמבין את הקשרים בין הבתים. חוץ מזה, ככל שדברים קרובים יותר, הם דומים יותר, ולכן סביר שייצוג גרפי של הבתים ישפיע לטובה. אינטואיטיבית, הגיוני שהמחיר של בית יהיה קשור למחיר של בית אחר באותה שכונה מאשר בצד השני של העיר.

GNN עבור לעבוד בתיאוריה, כי הוא מבין את היחסים המציאותיים שמשפיעים על המחירים.

בצורה פורמלית:

GNN מייצגת בתים כצמתים ואת קשריהם המרחביים כקשתות בגרף, וכך מאפשרות למדל תלות מרחבית וללכוד דפוסים מרחביים ששיטות מסורתיות עלולות להחמיץ. עקרון ההומופיליה ב־GNNs תואם את חוק הגאוגרפיה של טובלר, המדגיש קרבה מרחבית כבסיס לקשר הדוק יותר בין תצפיות.

* איך בונים GNN? איך יוצרים את הקשרים?

כל בית הוא צומת. הצומת מכיל מידע על הבית, כגון בן כמה הוא, מספר חדרים, והמיקום הגיאוגרפי שלו וכו'.

מהן חמש הדרכים להגדיר קשר בין שני בתים?

1. KNN: תחבר כל בית ל-K השכנים הקרובים שלו. תשתמש במרחק האריסון, שטוב למרחקים על כדור הארץ.
2. פרמטרים: K – מספר השכנים שאני בוחרת.
3. Gaussian Kernel: גם עובד לפי שכנים, אבל עובד בצורה מעריכית, ככה שהוא שם דגש על השכנים הקרובים. כלומר, ככל שאתה מתרחק, יש לבית פחות השפעה על המחיר של הבית שלך. מתקשר לחוק הגאוגרפיה של טובלר.

פרמטרים:

1. מרחק Haversine בין צמתים משמש למדידה.
2. סיגמה: פרמטר זה שולט ברוחב גרעין הגאוס (Gaussian kernel) ומוגדר כסטיית התקן של הווריאציות.
3. **פרמטר חיתוך (k)** נקבע כך שיתקרב לרמות דומות של דרגת צומת כמו בשיטות אחרות.

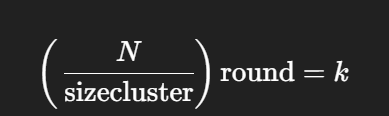
במחקר, נבחנו גרפים גאוסיים עם פרמטרים שנועדו להניב דרגות צומת ממוצעות של 35 ו־100.

1. K-means Clustering: ליצור קודם אשכולות (קלאסטרס), ואז לחבר את האשכולות האלה. יכול להיות שימושי נניח אם אני יוצרת אשכול של בתים כדי ליצור בניין, ואז מחברת בניינים כדי ליצור שכונה, נניח. מאוד עובד לפי עיקרון תכנון מונחה עצמים.

פרמטרים:

1. K, עבור כמות האשכולות.

על מנת להשיג דרגות צומת ממוצעות דומות לשיטות אחרות, מספר האשכולות k נגזר מפרמטר הנקרא *sizecluster* (שמתאים לדרגת הצומת הרצויה), לפי הנוסחה:



כאשר N הוא מספר הבתים.

1. שימוש במרחק Haversine.

במחקר נבחנו גרפי K-Means עם פרמטרים שנועדו להניב דרגות צומת ממוצעות של 35 ו־100.

1. Gabriel – אין פרמטרים. רף גבריאל הוא שיטה גאומטרית חסרת־פרמטרים לבניית גרף.

קיימת קשת בין שני צמתים אם ורק אם המעגל שקוטרו הוא הקטע המחבר את שני הצמתים בקשת אינו מכיל צמתים אחרים בתוכו.

אמפירית, נמצא שגרפי גבריאל מניבים דרגת צומת ממוצעת של כ־5 ברוב מערכי הנתונים (כאשר כוללים קשתות לא מכוונות וחיבורים עצמיים).

1. RNG – אין פרמטרים. גם כן מסתכל על מצבים גיאוגרפיים כדי לדעת לחבר בין שני בתים.

קיימת קשת אם המרחק בין שני הצמתים Vi,Vj קטן או שווה למקסימום של המרחקים בין צומת Vi לכל צומת אחר Vk ובין Vj לכל צומת אחר Vk.

גרף ה־RNG הוא תת־גרף של גרף גבריאל, וגם הוא גרף דליל.

אמפירית, נמצא שגרפי RNG מניבים דרגת צומת ממוצעת של כ־5 ברוב מערכי הנתונים (כאשר כוללים קשתות לא מכוונות וחיבורים עצמיים).

* תשובות: אין כמעט שיפור.

KNN באזור 100 שכנים הביא תוצאות טיפה טובות יותר מאשר עם פרמטר נמוך יותר. אבל בכללי, כל חמש השיטות האלו לא היו טובות בהרבה מהשיטות המקוריות שלנו (המבוססות עצים).

מנגד, בגרף לא מכוון, זה דווקא עבד לא רע. כלומר, יכול להיות שלהוסיף מידע מרחבי דווקא יוצר איזשהו רעש מאשר תורם. או שאולי המידע המרחבי היה זניח ביחס לשאר הדאטא. או בכללי, לפעמים פחות זה יותר. עדיף לשמור על פשטות.

פורמלית:

**גרפים לא מקושרים עשויים להצליח:** במערך VOV נמצא כי "גרף בלתי מקושר מניב ביצועים טובים יותר מכל גרף מבוסס־מיקום," מה שמעיד שלעתים מורכבות מרחבית מוסיפה רעש ולא מידע מועיל.

לכן, הגיוני להניח שאם נסיר חלק מהמידע, המודלים שלנו ישתפרו. נבחן את התיאוריה הזאת:

* כמה התוצאות שלנו משתפרות כשאנחנו נפטרים מהמידע של כל בית על המיקום שלו בצירים (קורדינטות X,Y,Z) וכמה התוצאות משתפרות כשאנחנו מסירים את הקשרים בין הבתים (קשר של מרחק בין שני בתים וכדומה)?

1. כשהורדנו קשרים, לא היה שיפור. כלומר, ה-GNN השתמש במידע הזה. במקומות עם מחירים גבוהים וצפופים, המידע על הקשרים היה מאוד רלוונטי, מה שגרם לאלגוריתם לנחש מחירים נמוכים מדי.
2. כשהסרנו את המידע על הצירים, זה פגע ביכולות לחזור מקומות עם מחירים נמוכים.

* האם ניתן להסיק מכך ששני סוגי המידע האלה משלימים זה את זה?

1. לא. התוצאות שקיבלנו מאוד לא עקביות. כמעט כאילו בכל דאטא סט שמתמקד במיקום אחר, היו השפעות שונות לחלוטין. כלומר, נוכל רק להסיק שלמיקום עצמו וכל מקרה פרטי יש את השיטה היעילה שלו.
2. בנוסף, ראינו באמת תוצאות שבהן שני סוגי המידע האלה הוכיחו את עצמם כלא משלימים זה את זה. לדוגמה, ראינו מקרים שבהם האלגוריתם בעצמו השתמש יותר במידע על הקשרים מאשר בקורדינטות, מה שמעיד שהאלגוריתם לא צריך "איזון" מסוים בשימוש שלו בשני סוגי הנתונים האלה, אלא יודע לכל מקרה פרטי מתי כדאי להשתמש במי יותר.

מי היה הכי טוב? איזו ארכיטקטורה של GNN?

1. GraphSAGE: ידוע בכך שהוא טוב בלאסוף מידע מהשכנים שלו.
2. SuperGAT: לומד איזה שכנים הכי חשובים.
3. GIN: טוב למידע טבלאי. משתמש בשתי רשתות קשירות בתור מערכת פנימית.
4. ארכיטקטורות טרנספורציה: כמו של למידת מכונה למחרוזות שמותאם לגרפים.

**Transformer architecture:** Fully parameterises the attention mechanism to assign different importance to neighbours, with attention coefficients defined by multi-head dot product attention. Convolution-specific parameters were left at their default settings as specified in PyTorch geometric

הפרמטרים שרצו במאמר הם על כל 4 הארכיטקטורות הם:

1. **Batch size:** 256
2. **Epochs:** 1000
3. **Early stopping patience:** 250 epochs based on validation loss
4. **Learning rate scheduler:** ReduceLROnPlateau with a reduction factor of 0.5
5. **Weight decay:** 5 × 10−4
6. **Loss function:** Mean Squared Error (MSE) for the regression task
7. **Neighborloader configuration:** Utilises n1 = 25 and n2 = 10 for scalability
8. **Hyperparameter Optimisation (HPO) Settings**
9. **Optimizer:** HPO (Adam or SGD(
10. **Learning rate:** HPO (range 10−4 to 10−1)
11. **Hidden dimension:** HPO (values: 32, 64, 128, 256)
12. **Dropout rate:** HPO (values: 0, 0.1, 0.2, 0.3, 0.4, 0.5)

מי היה הכי טוב? 1 ו-4. הם היו טובים יותר ופחות רגישים לאיך שהמפה הייתה מצוירת. כלומר, על אף שמקודם אמרנו שהתוצאות היו מאוד פרטניות ביחס לכל מקרה, אלה היו הכי טובים בליצור תוצאה כללית.

הערה: PEGCN היה די גרוע.

* מסקנה: GNN לא טוב מספיק. מודלים מבוססי עצים כמו CatBoost היו הכי טוב. LightGBM היה גם טוב. כל מה שקשור לעצים היה טוב בהרבה מ-GNN. אפילו דברים פשוטים כמו שכבות מרובות היו טובים יותר מ-GNN. כלומר, **GNN רע.**

למה? אולי GNN לא טוב מספיק עם כל המורכבויות האלו. המידע מסובך מידע. ואולי לתאר את הקשרים עם גרף פשוט לא באמת עוזר, או שהוא לא באמת משנה. או שאולי CatBoost ו- LightGBM פשוט טובים מדי. כל אלה תיאוריות שעולות במחקר.

פורמלית:

1. הממצא המרכזי הוא ש־ GNNs אינן מצליחות לעלות על ביצועי שיטות למידת מכונה ולמידה עמוקה. CatBoost נמצא כמודל המוביל, במיוחד כאשר האפקטים המרחביים בולטים, בעוד LightGBM עדיף במערכים שבהם האפקטים פחות חזקים (כגון VLI)
2. מודלים מבוססי־עצים מספקים חיזויים מדויקים יותר ובמקביל יעילים חישובית וניתנים ליישום כמעט ללא התאמות, בניגוד ל־GNNs ובמידה פחותה MLP הדורשים עיבוד מקדים נרחב, כוונון פרמטרים ויכולות חישוב יקרות.
3. GNNs מתקשות "לנצל מידע לא־מרחבי באותה מידה כמו מודלים מבוססי־עצים, המתאימים יותר לנתונים טבלאיים.
4. במערך VBR גרף גאוסי בדרגת צומת 100 עקף את האחרים, מה שמעיד שגישות המותאמות לצפיפות מקומית יכולות ללכוד קשרים מרחביים עדינים יותר.
5. כל שיטות בניית הגרפים מניבות ביצועים דומים יחסית," ולכן יש צורך בבחינה מותאמת־מערך.

**GNN רע.**

* האם היו מקרים שבהם הוא כן היה טוב?

1. כאשר הם יצרו מידע משלם עצמם בכך שהם הורידו את המידע של הקורדינטות מכולם, GNN היה טוב יותר מ- CatBoost ו- LightGBM. נראה ש-GNN עדיין הצליח להשתמש במידע הזה איכשהו, על אף שהסרנו אותה. כנראה יכולת הסקה כלשהי.

כלומר, כשאנחנו מעוורים את שאר המודלים, GNN טוב יותר. בעולם האמיתי, זה לא היה קורה, מן הסתם. אנחנו לא נסתיר מידע חשוב מהמכשיר שלנו.

בנוסף, קשרים מרובים שיפרו את התוצאות קצת, אבל המחיר של זה היה גרוע מדי ולא שווה את זה.

סיכום: להשתמש ב- CatBoost ו- CatBoost. בעיקר ב- CatBoost. הם גם מהירים יותר לאימון וקל יותר להשתמש בהם. וגם יש להם פשוט תוצאות טובות יותר.

אפשר להשתמש ב-GNN רק עבור מידע נורא מסובך, רק פה הוא די יעיל. אבל שוב, הוא יקר יותר, מסובך יותר, איטי יותר וכו'.

* מחקר עתידי: האם GNN יעיל עם גרפים יותר דינמיים? יש צורות ספציפיות של גרפים שבהן GNN יכול להיות יעיל.
* שימוש ב-MSE: המחקר מעריך את ביצועי שיטות בניית הגרפים השונות ומשווה את ה־GNNs מול קווי בסיס (baselines) באמצעות **MSE** על סט הבדיקה. שיטות הערכה נוספות במחקר היו:

1. MAE מוצג בטבלאות התוצאות המפורטות (נספח B) לצד MSE, עבור כל ארכיטקטורת GNN, שיטת בניית גרף וסט פרמטרים. הדבר מעיד כי נעשה בו שימוש כמדד נוסף לדיווח על ביצועי החיזוי של המודלים. היתרונות שלו הם שהוא מספק מבט משלים על שגיאת החיזוי, שכן זהו מדד נפוץ במשימות רגרסיה.
2. MSdiff משמש במיוחד למדידת הסבר גרפי (graph attribution) ע"י כימות ההבדל בתחזיות כאשר כל הקשרים בגרף מוסרים בזמן החיזוי. בנוסף, הוא משמש למדידת חשיבות מאפייני הצמתים ה־X-Y (קואורדינטות גיאוגרפיות), ע"י השוואת התחזיות כאשר הקואורדינטות עוברות מיצוע מול התחזיות המקוריות.

יתרונות במחקר זה: מאפשר כימות ישיר של ההשפעה של מאפיינים מרחביים ספציפיים (מבנה הגרף או קואורדינטות X-Y) על תחזיות המודל – דבר שמעניק תובנה על האופן שבו GNNs מנצלים מידע מרחבי. חסרונות: זהו מדד ייחודי לניתוח ייחוס (attribution) ואינו משמש כפונקציית הפסד כללית לאימון או כמדד ביצועים סטנדרטי כמו MSE או MAE.

* מבחני השוואה:

1. מבחן Holm ומבחן Iman–Davenport Friedman: המבחנים מבוצעים כדי להשוות באופן סטטיסטי את דירוגי הממוצע של האלגוריתמים השונים על פני מערכי נתונים, ולבדוק אם קיימים הבדלים מובהקים בביצועים שלהם.

יתרונות: מספקים מסגרת סטטיסטית קפדנית לקביעת מובהקות ההבדלים בין מודלים, ומאפשרים להסיק מסקנות יציבות יותר מאשר רק השוואת ערכי MSE גולמיים.

חסרונות: החוקרים מציינים כי לא מתקיימות במלואן ההנחות של מבחנים אלו, אשר דורשים לרוב יותר מ־10 מערכי נתונים ויותר מ־5 מודלים (במחקר זה השתמשו ב־6 מערכי נתונים ו־8 מודלים). לכן, על אף שהמבחנים מצביעים על הבדלים מובהקים, יש לפרש את התוצאות בזהירות.

* לסיכום:

1. GNNנחותות ביחס למודלים מבוססי־עצים (CatBoost, LightGBM).
2. הארכיטקטורות המובילות הן GraphSAGE ומודלים מבוססי Transformer.
3. KNN עם דרגת צומת גבוהה (כ־100) עוזר לפעמים, אם כי התרומה שולית.
4. קואורדינטות קריטיות בעיקר לבתים זולים; מבנה גרפי מועיל יותר בבתים יקרים.
5. GNN דורשות משאבים רבים יותר.
6. שישה מערכי נתונים King County בארה״ב וחמישה מחוזות בפלנדריה, בלגיה.
7. מדד הערכה: MSE.
8. לפי מבחן הולם, GNN מתפקדות גרוע מ־CatBoost ברמת מובהקות 0.05.