Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра інформатики факультету інформатики

Дослідження методів навчання структури баєсівської мережі

Текстова частина до курсової роботи
за спеціальністю „Комп'ютерні науки та інформаційні технології" 122

Керівник курсової роботи
старший викладач, к.т.н.
Ющенко Юрій Олексійович
"____" _____ 2020 р.

Виконала студентка ФІ-4
Салій Анна Сергіївна
"____" _____ 2020 р.

Київ 2020

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра інформатики факультету інформатики

ЗАТВЕРДЖУЮ
Викладач кафедри інформатики ,
старший викладач, к.т.н. _____ Ющенко Ю.О.
(підпис)
„____”_____2020р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на курсову роботу
студенту Салій Анні Сергіївні
факультету інформатики 4 курсу бакалаврської програми
ТЕМА: **Дослідження методів навчання структури баєсівської мережі**

Зміст ТЧ до курсової роботи:
Індивідуальне завдання
Вступ
Огляд теоретичного матеріалу і здійснення дослідження
Висновки
Список літератури
Додатки (за необхідністю)

Дата видачі „___” _____ 2019 р. Керівник_____
(підпис)
Завдання отримав _____
(підпис)

Календарний план виконання курсової роботи

Тема: Дослідження методів навчання структури баєсівської мережі

Календарний план виконання роботи:

№ п/п	Назва етапу	Термін виконання етапу
1.	Отримання завдання на курсову роботу.	20.11.2019
2.	Огляд технічної літератури за темою роботи.	05.12.2019
3.	Вивчення методів аналізу даних.	19.12.2019
4.	Написання умов до текстової частини роботи.	19.01.2020
5.	Написання умов до практичної частини проекту.	21.01.2020
6.	Написання першої частини курсової роботи.	22.02.2020
7.	Перегляд змісту роботи з керівником.	23.02.2020
8.	Написання другої частини курсової роботи.	24.03.2020
9.	Перегляд змісту роботи з керівником.	25.03.2020
10.	Написання висновків курсової роботи.	28.03.2020
11.	Перегляд змісту роботи з керівником.	30.03.2020
12.	Створення слайдів для доповіді та написання доповіді.	02.04.2020
13.	Аналіз отриманих результатів з керівником.	04.04.2020
14.	Корегування роботи .	09.04.2020
15.	Остаточне оформлення роботи та слайдів.	11.04.2020

Студент _____

Керівник _____

“_____” _____

Анотація

У даній роботі розглядаються важливі проблеми навчання структури баєвської мережі. Почнемо з короткого ознайомлення з баєсівськими мережами. Надається огляд навчання структури баєсівських мереж з даних та не тільки. Потім зосередимо увагу на конкретному завданні навчити структуру мережі Баєса із повністю спостережуваних даних, використовуючи підхід пошуку та оцінки. Обговорюємо також баєсівську оцінку та її наслідки. Внесок цієї роботи включає: дослідження літератури про існуючі алгоритми навчання структури; генетичний алгоритм структури навчання, який здійснює пошук по простору графа і використовує оператори мутації та кросовера.

.

Зміст

ВСТУП

Іноді на практиці потрібно обчислити ймовірність невизначеної причини, враховуючи деякі спостережувані свідчення. Наприклад, ми хотіли б знати ймовірність конкретного захворювання, коли спостерігаємо симптоми у пацієнта. Такі проблеми часто складні з багатьма взаємопов'язаними змінними. Можливо, існує багато симптомів і навіть більше потенційних причин. На практиці зазвичай можна отримати лише зворотну умовну ймовірність, тобто ймовірність свідчень, що дають причину, ймовірність спостереження за симптомами, якщо у пацієнта є захворювання.

Інтелектуальні системи повинні міркувати про своє оточення. Наприклад, роботові необхідно знати про можливі результати своїх дій, а система медичних експертів повинна знати, які причини призводять до яких наслідків. Інтелектуальні системи почали використовувати імовірнісні методи, щоб впоратися з невизначеністю реального світу. Замість того, щоб будувати спеціальну систему імовірнісних міркувань для кожної нової програми, хотілось б загальну основу, яка б дозволила імовірнісно міркувати в будь-якій новій програмі, не відновлюючи все з нуля. Саме цим і обґрунтована **актуальність** обраної теми роботи. Баєсівські мережі, що вперше з’явилися в роботі Йуди Перла та його колег наприкінці 1980-х, пропонують саме таку незалежну основу для вірогідних міркувань.

У даній роботі досліджується **проблема** навчання баєсівської структури мережі. **Мотивація** написання та **новизна** даної роботи виходить із проведеного дослідження та розробленого алгоритму, представлених в розділі третьому та четвертому відповідно.

У першому розділі знайомимося з баєвською мережею та кількома іншими поняттями, які використовуються в даній роботі, знайомимся з історією терміна баєсівської мережі, розглядаємо деякі приклади та методи побудови

баєвської мережі, після чого детальніше розглядаємо ручну побудову та автоматичну побудову баєсівської мережі.

У другому розділі досліджуємо проблему навчання баєсівських мереж з даних та обговорюємо різні варіанти вирішення цього завдання.

У третьому розділі досліджуємо літературу про навчання структури, акцентуючи увагу на навчанні з повних даних та навчанні за допомогою еволюційних алгоритмів. Для стислості пропускаємо безліч інших способів навчання структури, таких як тести на умовну незалежність та пошук класів еквівалентності.

У четвертому розділі представляємо генетичний алгоритм для навчання структури баєсівської мережі з повністю спостережуваного набору даних, робимо експеременти та аналізуємо їх.

РОЗДІЛ 1. Теоретична частина

У першому розділі детально описуємо певні аспекти баєсівських мереж. Розділ структурован наступним чином. У розділі 1.1 знайомимся з історією терміна баєсівської мережі. Поняття баєсівської мережі розкрито у розділі 1.2 разом із деякими прикладами. Методи побудови баєвської мережі наведено в розділі 1.3, після чого детальніше розглядаємо ручну побудову в розділі 1.3.1 та автоматичну побудову в розділі 1.3.2. У розділі 1.4 ознайомлюємося з структурою баєсівської мережі.

1.1. Історія виникнення баєсівської мережі

Термін Баєсова мережа був введений Йудою Перлом у 1985 році, щоб підкреслити:

- Часто суб'єктивної природи вхідної інформації.
- Покладання на баєсове обумовлювання як основу для уточнення інформації.
- Відмінність між причинними та доказовими способами міркування.

В кінці 1980-х років праці Йуди Перла «Імовірнісне міркування в інтелектуальних системах» та Річарда Неаполітана «Імовірнісне міркування в експертних системах» підсумували властивості баєсових мереж та утвердили баєсові мережі як область дослідження.

1.2. Поняття баєсівської мережі

У данній роботі вирішили зосередитися на спеціальному класі моделей під назвою Баєсові мережі. Вони є графічними моделями, це означає, що вони містять частину, яку можна зобразити як граф.

Баєсівська мережа - це ймовірнісна графова модель, яка може бути використана для побудови моделей з даних. Їх також називають мережами Баєса, мережами переконань та іноді причинними мережами. Баєсівські мережі можуть бути використані для широкого спектру завдань, включаючи прогнозування,

виявлення аномалії, діагностику, автоматизоване розуміння, міркування, прогнозування часових рядів та прийняття рішень в умовах невизначеності. Баєсівські мережі є ймовірнісними, оскільки вони побудовані з розподілу ймовірностей, а також використовують закони ймовірності для прогнозування та виявлення аномалії, для міркувань та діагностики, прийняття рішень в умовах невизначеності та прогнозування часових рядів.

Баєсівська мережа - це граф, який складається з вузлів і спрямованих посилань між ними. У багатьох баєсівських мережах кожен вузол являє собою імовірнісну змінну, таку як чийсь зріст, вік чи стать. Змінна може бути дискретна, наприклад, гендер = {жінка, чоловік} або може бути безперервною, наприклад, якась відстань. Ми називаємо вузли з більш ніж одною змінною багатозмінними вузлами.

Вузли та зв'язки утворюють **структуру баєсівської мережі**, і ми називаємо це структурною специфікацією.

Сервер Баєса підтримує безперервні змінні з умовними лінійними розподілами Гаусса. Це означає, що безперервні розподіли можуть залежати один від одного (є багатоваріантними), а також можуть залежати від однієї або декількох дискретних змінних.

Ребра додаються між вузлами, щоб вказати, що один вузол умовно залежить від іншого. Якщо зв’язку між двома вузлами не існує, це не означає, що вони умовно незалежні, оскільки вони можуть бути з'єднані через інші вузли.

Баєсівська мережа представляє причинно-ймовірнісну залежність між набором випадкових величин, їх умовними залежностями, і вона забезпечує компактне **зображення спільного розподілу ймовірностей**. БМ складається з двох основних частин: **спрямованого ациклічного графа** та **наборів умовних розподілів ймовірностей.** Спрямований ациклічний граф - це граф, в якому відсутні орієнтовані цикли, тобто шляхи, що починаються і закінчуються в одній і тій самій вершині.

Якщо на графу існує причинно-ймовірнісна залежність між двома випадковими змінними, відповідні два вузли з'єднані спрямованим ребром, тоді

як спрямований край від вузла A до вузла B вказує, що випадкова величина A викликає випадкову величину B. Оскільки спрямовані ребра являють собою статичну причинно-імовірнісну залежність, цикли у графі не дозволені. Для кожного вузла графа визначається умовний розподіл ймовірностей. Іншими словами, умовний розподіл ймовірності вузла (випадкова величина) визначається для кожного можливого результату попереднього причинного вузла.

Для ілюстрації розглянемо наступний тривіальний приклад. Припустимо, ми намагаємось увімкнути наш комп’ютер, але комп'ютер не запускається (спостереження / свідчення). Ми хотіли б дізнатися, яка з можливих причин комп'ютерних збоїв є більш імовірною. У цій спрощеній ілюстрації ми припускаємо лише дві можливі причини цього нещастя: збій електрики та несправність комп’ютера. Відповідний спрямований ациклічний граф зображений на рисунку 1.

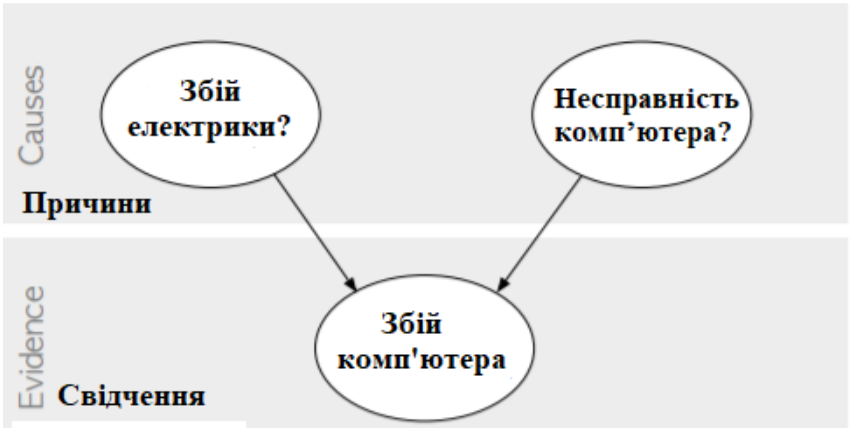

Рисунок 1: Спрямований ациклічний граф, що представляє дві можливі незалежні причини відмови комп'ютера.

Дві причини в цьому банальному прикладі вважаються незалежними (між двома причинними вузлами немає ребра. Якщо в графі немає циклу, баєсівські мережі здатні зафіксувати стільки причинно-наслідкових зв’язків, скільки їх потрібно для того, щоб достовірно описати ситуацію тієї частки зовнішнього середовища, для якої розв’язується задача.

Вся концепція баєсівських мереж побудована на **теоремі Баєса**, яка допомагає нам виразити умовний розподіл ймовірності причини з огляду на

свідчення, використовуючи зворотну умовну ймовірність огляду свідчень з урахуванням причини:

$$\mathrm{P}\left[\mathrm{Cause} \mid \mathrm{Evidence}\right] = \mathrm{P}\left[\mathrm{Evidence} \mid \mathrm{Cause}\right] \cdot \frac{\mathrm{P}\left[\mathrm{Cause}\right]}{\mathrm{P}\left[\mathrm{Evidence}\right]}$$

Ми можемо побудувати повну ймовірнісну модель, лише вказавши умовний розподіл ймовірностей у кожному вузлі.

Повертаючись до нашого прикладу, ми припускаємо, що відмова електроенергії, позначена E, відбувається з вірогідністю 0,1, P [E = так] = 0,1, а несправність комп'ютера, позначена M, виникає з ймовірністю 0,2, P [M = так] = 0,2. Доцільно вважати збій електрики та несправність комп'ютера незалежними. Крім того, ми припускаємо, що якщо немає проблем з електрикою, а комп'ютер не працює, комп'ютер працює нормально. Іншими словами, якщо C позначає збій комп'ютера, то P [C = так | E = ні, M = ні] = 0. Якщо немає проблеми з електрикою, але комп'ютер має несправність, ймовірність виходу з ладу комп'ютера становить 0,5, P [C = так | E = ні, M = так] = 0,5. Нарешті, якщо електроенергія вимкнеться, комп'ютер не запуститься незалежно від його потенційної несправності, P [C = так | E = так, M = ні] = 1 і P [C = так | E = так, M = так] = 1. Ймовірність збою комп'ютера P [C = так] можна обчислити як

$$\begin{aligned}
\mathrm{P}\left[C = \text{так}\right] &= \sum_{E,M} \mathrm{P}\left[C = \text{так}, E, M\right] \\
&= \sum_{E,M} \left(\mathrm{P}\left[C = \text{так} \mid E, M\right] \cdot \mathrm{P}\left[E\right] \cdot \mathrm{P}\left[M\right]\right) \\
&= 0.19
\end{aligned}$$

Ми можемо вважати ймовірність P [C = так] = 0,19 як апріорну ймовірність збою комп'ютера перед врахуванням свідчень. Графічна модель з апріорним розподілом ймовірностей, тобто перед спостереженням за будь-якими свідченнями, зображена на рисунку 2.

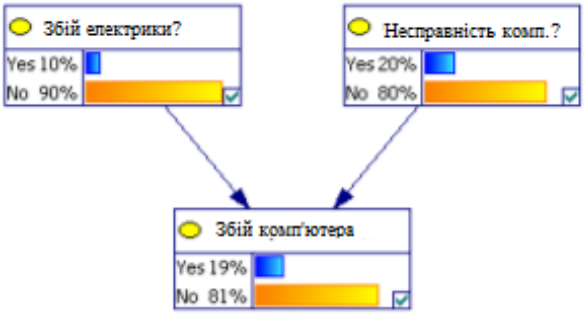

Рисунок 2: Спрямована графова модель.

Припустимо, що ми намагались увімкнути комп’ютер, але не вийшло. Іншими словами, ми спостерігаємо C = ні з вірогідністю 1 і задаємося питанням, як змінився розподіл ймовірностей відмови електроенергії E та несправності комп'ютера M з огляду на спостережувані свідчення. Використовуючи формулу Баєса, знаходимо

$$
\begin{aligned}
\mathrm{P}\left[E = \text{так} \mid C = \text{так}\right] &= \sum_{M} \mathrm{P}\left[E = \text{так}\,,\, M \mid C = \text{так}\right] \\
&= \sum_{M} \frac{\mathrm{P}\left[C = \text{так} \mid E = \text{так}\,,\, M\right] \cdot \mathrm{P}\left[E = \text{так}\right] \cdot \mathrm{P}\left[M\right]}{\mathrm{P}\left[C = \text{так}\right]} = 0.53 \\
\mathrm{P}\left[M = \text{так} \mid C = \text{так}\right] &= \sum_{E} \mathrm{P}\left[E,\, M = \text{так} \mid C = \text{так}\right] \\
&= \sum_{E} \frac{\mathrm{P}\left[C = \text{так} \mid E,\, M = \text{так}\right] \cdot \mathrm{P}\left[E\right] \cdot \mathrm{P}\left[M = \text{так}\right]}{\mathrm{P}\left[C = \text{так}\right]} = 0.58
\end{aligned}
$$

Графічна модель із апостеріорним розподілом ймовірностей, тобто після спостереження свідчення (відмова комп'ютера), зображена на рисунку 3.

Рисунок 3: Спрямована графова модель.

Зауважте, що спостережуваний збій викликав сильну залежність між спочатку незалежними можливими причинами; наприклад, якщо одна причина могла бути виключена, інша повинна мати місце. Тим не менш, наведені вище результати все ще не дуже корисні.

Припустимо розширення прикладу, включивши в модель ще один доказ, зокрема світловий збій L. Ми припускаємо, що світловий збій не залежить від несправності комп'ютера. Як і раніше, якщо електроенергію вимкнено, світло не буде світити ні за яких обставин, P [L = так | E = так] = 1. Якщо немає проблем з електрикою, ми припускаємо, що все-таки є 0,2 шансу, що світло згасне (розбита лампочка), P [L = так | E = ні] = 0,2. Використовуючи той же алгоритм, що і раніше, отримуємо, що попередня ймовірність P [L = так] = 0,28. Розширена графічна модель з апріорним розподілом ймовірностей, тобто перед спостереженням за будь-яким свідченням, зображена на рисунку 4.

Рисунок 4: Спрямована графова модель.

На рисунку 5 показані зміни в апостеріорному розподілі ймовірностей після спостереження свідчень для всіх чотирьох комбінацій результатів відмови від світла та комп’ютера. Наприклад, якщо ми спостерігаємо як збій комп’ютера, так і світловий збій, тобто спостерігаємо як C = так, так і L = так з вірогідністю 1 (правий верхній граф на рисунку 5), отримуємо P [E = так | C = так, L = так] = 0,85 і P [M = так | C = так, L = так] = 0,33. Спостереження, що і світло, і комп’ютер не працюють, значно збільшили ймовірність збою електроенергії. Збій комп'ютера, таким чином, можна пояснити. У трьох інших випадках принаймні один з приладів (світло та комп'ютер) працює, і тому ми можемо стверджувати, що з електроенергією точно немає нічого поганого. Якщо світло працює, але

комп'ютер не запускається (нижній лівий граф на рисунку 5), ми точно знаємо, що з електрикою немає нічого поганого, тому несправність комп’ютера є єдиним можливим поясненням збою комп'ютера.

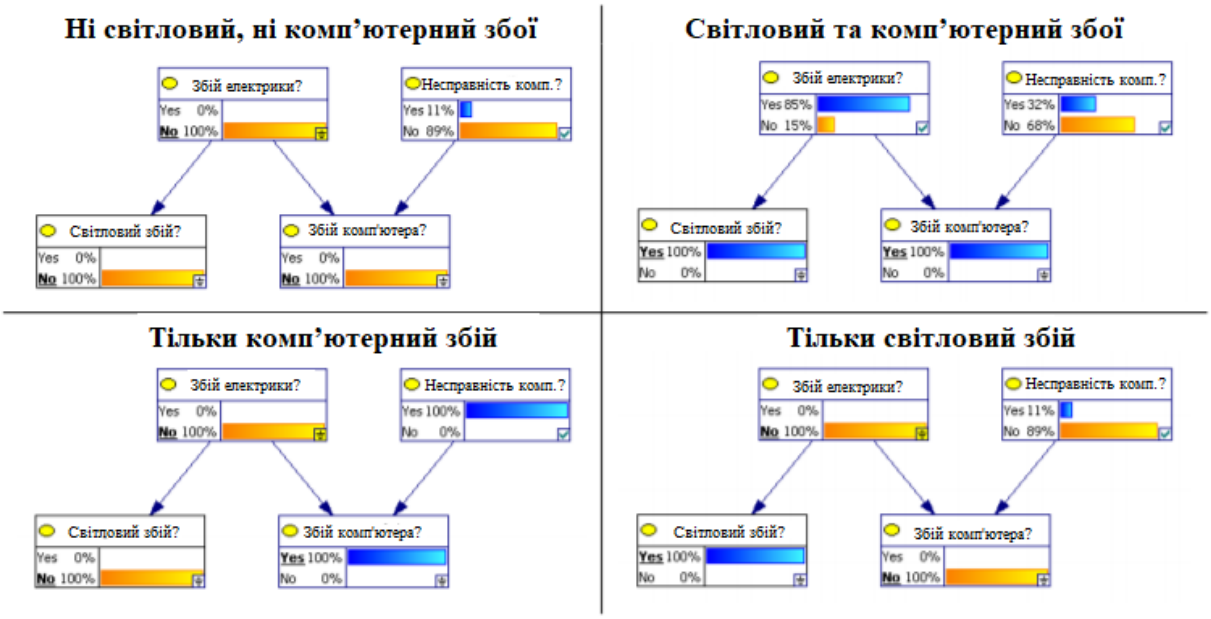

Рисунок 5: Спрямована графова модель.

На практиці баєсівські мережі значно складніші, використовують десятки чи навіть сотні вузлів. Важливо також зазначити, що кожен вузол у графі повинен бути пов'язаний принаймні з одним ребром до іншого вузла. В іншому випадку відокремлений вузол не залежить від усіх інших вузлів, і тому немає необхідності враховувати цей вузол.

Через їх візуальний вигляд, баєсівські мережі можуть плутати з марківськими моделями. Однак між цими двома поняттями є принципова різниця. Модель Маркова - приклад графу, який представляє лише одну випадкову змінну, і вузли представляють можливі реалізації випадкової змінної у різних часових точках. На противагу цьому, кожен вузол в баєсівської мережі являє собою одну випадкову змінну в одну мить. Іншими словами, моделі Маркова фіксують динаміку однієї випадкової величини, тоді як мережі Баєса фіксують статичний причинно-наслідковий зв’язок серед набору випадкових величин.

Баєсівські мережі можна використовувати як для якісного, так і кількісного моделювання, оскільки вони можуть поєднувати об'єктивні емпіричні

ймовірності (частоти) з суб'єктивними оцінками. Баєсівські мережі також можуть бути використані як діаграми впливу замість дерев рішень. У порівнянні з деревами рішень, баєсівські мережі, як правило, більш компактні, простіші в побудові та їх зміні. Кожен параметр з’являється лише один раз у баєсівській мережі, і в разі потреби мережа може перетворитися на дерево рішень, тоді як зворотне не завжди можливо.

Основна слабкість полягає в тому, що баєсівські мережі вимагають апріорного розподілу ймовірностей, які можуть мати оманливий вплив на результати.

1.3. Аналіз методів побудови баєсівської мережі

Існує два способи побудови баєсівської мережі: ручна побудова або автоматична побудова з баз даних. Обидва способи мають переваги та недоліки.

1.3.1. Ручна побудова

Ручна побудова баєсівської мережі передбачає попередні знання у цій галузі. Перший крок - побудова спрямованого ациклічного графа, після чого другий крок - оцінка умовного розподілу ймовірностей у кожному вузлі.

Спрямований ациклічний граф: побудова спрямованого ациклічного графа починається з ідентифікації відповідних вузлів (випадкових змінних) та структурної залежності між ними. Не всі змінні потрібно розглядати; насправді деякі випадкові величини можуть визначати непомічені величини, які, як вважають, впливають на результати спостереження. Дані, приховані змінні та параметри всі однаково розглядаються як вузли на графові. Баєсівський підхід заснований на тому, що всі невідомі величини вважаються випадковими змінними, а отже, природно включати параметри як вузли у графі, так само як і приховані змінні та потенційно спостережувані величини. Наступний крок - накреслити мережу з урахуванням зв’язків між випадковими змінними. Структура графу, як правило, ґрунтується на предметних знаннях, хоча критика та перегляд моделі часто є важливими. Незважаючи на свою назву, баєсівські

мережі не обов'язково передбачають вплив баєсівської статистики. Справді, для оцінки параметрів умовного розподілу ймовірностей зазвичай застосовують методи частот. Звичайно, можна реалізувати й баєсівський підхід використовуючи гіперпараметри замість параметрів умовного розподілу ймовірностей, що лежать в основі графа, які можна розглядати як вузли в моделі.

Умовний розподіл ймовірностей: побудований спрямований ациклічний граф повинен містити умовні розподіли ймовірностей для кожного вузла в графі. Якщо змінні дискретні, це може бути представлено у вигляді таблиці (мультиноміального розподілу), в якій перерахована ймовірність того, що дочірній вузол приймає кожне з різних його значень для кожної комбінації значень батьків. Якщо умовний розподіл ймовірності недоступний, для отримання цього умовного розподілу з даних (наприклад, емпіричного умовного розподілу ймовірності / частоти) можуть застосовуватися інші статистичні методи. На даний момент, Баєсова мережа повністю визначена. Однак перед використанням мережі в реальному застосуванні необхідно провести аналіз чутливості. Аналіз чутливості може виконуватися або як односторонній детермінований аналіз чутливості (тобто, змінюючи один параметр відразу у визначеному діапазоні), або як імовірнісний аналіз чутливості (тобто, змінюючи всі параметри мережі відразу за визначеним розподілом ймовірностей) .

1.3.2. Автоматична побудова

На відміну від ручної побудови, автоматична не вимагає експертних знань в даній області. Баєсівські мережі можуть автоматично навчатися безпосередньо з баз даних, використовуючи алгоритми, засновані на досвіді, часто вбудовані у відповідне програмне забезпечення. Однак недоліком є те, що автоматична конструкція ставить більше вимог до даних. Крім того, повинно бути достатньо даних, щоб задовольнити алгоритм щодо надійних оцінок умовного розподілу ймовірностей. Для ручної побудови умовні розподіли ймовірностей вважаються завідома відомими. В той час як автоматична побудова передбачає як створення структури мережі, так і оцінку умовних розподілів ймовірностей.

1.4. Структура баєсівської мережі

Основна мета структури баєсівської мережі - графічно узагальнити ряд умовних незалежних зв’язків. Простими словами, структура передбачає набір умовних незалежних зв’язків серед залучених змінних. Крім незалежностей, структура графів баєсівської мережі може також використовуватися в певних областях для представлення взаємозв'язку причинних ефектів через ребра та їх напрямки. У цих випадках батьки вузла вважаються "прямими причинами" кількості, представленої цим вузлом. Однак це справедливо лише за певних припущень, найбільш важливими є наступні:

- чи є, чи немає загальних неспостережуваних причин (змінних) двох або більше спостережуваних змінних. Якщо таких змінних немає, вважається, що властивість причинної достатності відповідає дійсності. Неспостережувані змінні називають також латентними або прихованими змінними.
- враховуючи причинну достатність, чи можливо для більш ніж однієї мережевої структури відповідати обмеженням. Ці обмеження є статистичними залежностями, які спостерігаються в даних.

Баєсівські мережі можуть використовуватися як причинно-наслідкові моделі у звичайній “причинно-наслідковій” інтерпретації, коли ми можемо сміливо припускати наведені вище припущення (що рідко, особливо для причинної достатності). В даний час існує великий інтерес та дослідження щодо виявлення причинної недостатності, наприклад, з використанням інструментальних змінних.

РОЗДІЛ 2. Навчання структури Баєсової мережі з даних

У цьому розділі досліджуємо проблему навчання баєсівських мереж з даних та обговорюємо різні варіанти вирішення цього завдання.

У попередньому розділі дізналися, що можливо створити баєсівські мережі вручну. Але цей процес трудомісткий і схильний до помилок, і отримана модель успадкує будь-які недоліки в міркуваннях експертів. У великих галузях запитувати експертів про кожну деталь може бути недоцільно. У деяких предметних областях просто немає експерта з усіма необхідними знаннями, або відповідний розподіл змінюється занадто часто, щоб моделювати вручну кожен раз. Альтернативний спосіб отримати баєсівську мережу - це дізнатися її безпосередньо з набору прикладів даних. Припустимо, що проблемна область має деякий невідомий розподіл P^* і у нас є набір даних D = {d [1], ..., d [M]} з M зразків з P^*. Припускаємо, що зразки є незалежними та однаково розподіленими. Також можемо мати деякі попередні знання або обмеження щодо моделі M˜, яку намагаємося навчити. Наприклад, можемо знати, що змінна Y інстанціюється після змінної X, і тому Y не могла бути причиною X. Враховуючи ці два входи (набір даних D та будь-які обмеження на M˜), хочемо знайти модель M˜, яка найкраща відповідно до деякого критерію.

Критерій, який використовується для визначення якості моделі, залежить від мети, для якої навчаємо баєсівську мережу. Є три основні цілі навчання баєсівських мереж:

- **Оцінка щільності.** Хочемо побудувати модель M˜, що представляє розподіл P, максимально наближений до справжнього розподілу P^*. Будемо використовувати цю модель для відповіді на загальні запити висновування. Тут не маємо відношення до відновлення основної структури P^*, яка, як правило, невідома. Натомість абсолютно задоволені більш простою моделлю, яка фіксує відповідні аспекти розподілу.
- **Конкретні завдання прогнозування.** Заздалегідь знаємо, що будемо використовувати навчену модель лише для конкретних видів запитів. Наприклад, є проблема класифікації, де цікаво передбачити

значення однієї змінної, враховуючи всі інші. У таких випадках можемо оптимізувати модель для цієї конкретної задачі передбачення, а не оптимізувати її для відповіді на загальні запити висновування, як при оцінці щільності.

- **Виявлення знань.** Хочемо дізнатися модель, а потім навчити її, щоб отримати знання про предметну область. Наприклад, можемо переглянути залежність зв'язків між змінними. Перше питання - ідентифікація: кілька моделей можуть бути хорошими для одного набору даних. Друге питання - це вимірювання впевненості: однієї навченої моделі недостатньо для прийняття висновків про предметну область, оскільки часто кілька різних моделей можуть однаково добре пояснити набір даних. При навчанні баєсівських мереж для виявлення знань критерій оптимізації сильно залежить від того, як плануємо використовувати навчену модель.

Завдання навчання мережі баєсівської з даних варіюється в двох вимірах: скільки обмежень маємо в моделі, яку ми намагаємося дізнатися, і скільки даних спостерігається.

2.1. Навчання параметрів з повністю спостережуваних даних

Коли дані спостережуються повністю і знаємо структуру G моделі, навчання параметрів θ - це легке завдання. (Це стосується дискретних чи гауссових змінних; оцінка параметрів може бути складнішою для інших локальних імовірнісних моделей.) Припустимо, є граф G над вузлами X і набір даних D = {d [1], ..., d [M]} з M екземплярів. Існує два підходи до навчання параметрів θ з набору даних D: Метод максимальної правдоподібності та Баєсова оцінка. Обидва ці методи обчислюють параметри у закритому вигляді, використовуючи статистику з набору навчальних даних D.

Метод максимальної правдоподібності: імовірність даних визначається як ймовірність того, що заданий набір параметрів виробляє дані:

$$L(\boldsymbol{\theta} : \mathcal{D}) \;=\; P(\mathcal{D} \mid \boldsymbol{\theta}) \;=\; \prod_m P(\boldsymbol{d}[m] \mid \boldsymbol{\theta}).$$

Через факторизацію баєсівських мереж ймовірність розкладається наступним чином:

$$L(\boldsymbol{\theta} : \mathcal{D}) \;=\; \prod_i L_i(\boldsymbol{\theta}_{X_i|\mathrm{Pa}_{X_i}} : \mathcal{D})$$

де локальна ймовірність для вузла X_i:

$$L_i(\boldsymbol{\theta}_{X_i|\mathrm{Pa}_{X_i}} : \mathcal{D}) \;=\; \prod_m P(x_i[m]|\mathrm{pa}_{X_i}[m], \boldsymbol{\theta}_{X_i|\mathrm{Pa}_{X_i}}).$$

Ми можемо максимізувати кожен локальний термін вірогідності незалежно та об'єднати їх для отримання рішення методу максимальної правдоподібності для всієї мережі.

Оцінка параметрів Баєса. Ставимо апріорний розподіл P (θ) за параметрами, який представляє думку про параметри θ, перш ніж побачимо будь-які дані. Враховуючи набір навчальних даних D, використовуємо правило Байєса для отримання апостеріорного розподілу за параметрами:

$$P(\boldsymbol{\theta} \mid \mathcal{D}) \;=\; \frac{P(\mathcal{D} \mid \boldsymbol{\theta})\; P(\boldsymbol{\theta})}{P(\mathcal{D})}.$$

Терм у знаменнику - це константа нормалізації, яка не залежить від θ. Чисельник - це добуток апріорних і вірогідних даних. Щоб апостеріорний розкладався аналогічним чином, потрібно, щоб апріорний задовільнив дві умови. По-перше, незалежність глобальних параметрів говорить про те, що апріорні для окремих об’єднань незалежні:

$$P(\boldsymbol{\theta}) \;=\; \prod_i P(\boldsymbol{\theta}_{X_i|\mathrm{Pa}_{X_i}}).$$

По-друге, незалежність локальних параметрів говорить про те, що апріорні для вузла є незалежними, даючи різні призначення u його батькам U:

$$P(\boldsymbol{\theta}_{X|U}) \;=\; \prod_u P(\boldsymbol{\theta}_{X|u}).$$

Якщо апріорний задовольняє цим двом умовам, то апостеріарний розкладається аналогічно ймовірності:

$$P(\boldsymbol{\theta} \mid \mathcal{D}) \;=\; \prod_i \prod_{\mathrm{pa}_{X_i}} P(\boldsymbol{\theta}_{X_i|\mathrm{pa}_{X_i}} \mid \mathcal{D}).$$

2.2. Навчання структури з повністю спостережуваних даних

Тут намагаємося навчити як структуру G, так і параметри θ для деякого невідомого розподілу P^*, з якого маємо повністю спостережуваний набір даних D незалежних та однаково розподілених зразків. Методи структури навчання приблизно поділяються на три категорії: підходи, засновані на обмеженнях, методи пошуку та оцінювання та Баєсова модель усереднення.

Підходи на основі обмежень використовують статистичні тести на умовну незалежність для пошуку залежностей та незалежностей між змінними. Потім будують баєсівську мережу для представлення набору виявлених незалежностей. Тести на незалежність часто недостовірні для більш ніж декількох змінних, а обмежені методи чутливі до збоїв у цих тестах. Це тому, що ребро може бути неправильно виключено з мережі на основі результату одного невдалого тесту.

Методи пошуку та оцінювання трактують навчання структури як проблему вибору моделі. Визначаємо набір можливих структур і функцію оцінювання, яка оцінює, наскільки добре структура відповідає навчальним даним. Потім використовуємо процедуру пошуку, щоб знайти структуру.Простір баєсівських мереж з n вузлами має $2^{O(n^2)}$ можливі структури, і проблема пошуку найкращої структури в цілому є NP важкою. Отже, метод пошуку (такий як сходження на гірку або еволюційний алгоритм) не буде враховувати всі можливі структури і не гарантовано знайде оптимальну структуру.

Методи **Баєсової моделі усереднення** створюють ансамбль можливих структур, а не навчають єдину структуру. Ансамблі можуть бути створені за допомогою будь-якого методу навчання структури, наприклад методів на основі обмежень та методів пошуку та оцінки, викладених вище. Ансамблеві методи намагаються порівняти прогнозування всіх можливих структур, замість того, щоб покладатися на прогнозування єдиної структури.

2.3. Навчання параметрів з частково спостережуваних даних

Коли набір даних містить загублені значення, спершу потрібно розглянути питання про відсутність даних випадково, це означає, що факт відсутності змінної не дає додаткової інформації про будь-яку (іншу) відсутню змінну. Це дозволяє ігнорувати модель спостереження при навчанні параметрів. Нагадаємо, вірогідність розкладається зручно у випадку повністю спостережуваних даних. Що стосується відсутніх даних, це вже не так, оскільки загублені значення вводять зв'язок між змінними. Отримана функція ймовірності є мультимодальною (має кілька локальних максимумів) і не має простого представлення закритої форми.

Все ще можемо розкласти ймовірність як продукт над кожним екземпляром у наборі даних D. Потім можемо виразити кожен фактор у добутку як суму над усіма можливими присвоєннями загублених змінних H [m] у цьому випадку:

$$L(\boldsymbol{\theta} : \mathcal{D}) \;=\; \prod_{m} P(\boldsymbol{o}[m] \mid \boldsymbol{\theta}) \;=\; \prod_{m} \sum_{\boldsymbol{h}[m]} P(\boldsymbol{o}[m], \boldsymbol{h}[m] \mid \boldsymbol{\theta}).$$

Кількість можливих присвоєнь H [m] експоненціальна в кількості загублених змінних. Для обчислення функції ймовірності потрібно виконати висновування в мережі для кожного примірника D. Це значно ускладнює проблему навчання параметрів, оскільки висновування у кращому випадку повільне, а в гіршому - нерозв'язне.

Gradient Ascent підтримує поточний набір параметрів і в кожній ітерації обчислює градієнт у просторі параметрів. Градієнт дає напрямок, в якому слід змінювати параметри для максимального збільшення ймовірності. Потім алгоритм на невелику кількість змінює параметри в напрямку градієнта і продовжує наступну ітерацію.

EM-алгоритм варіюється між двома кроками. На E-кроці (expectation) вираховується очікуване значення функції правдоподібності, при цьому приховані змінні розглядаються як спостережувані. На M-кроці (maximization) вираховується оцінка максимальної схожості, таким чином збільшується очікувана схожість, вирахувана на E-кроці. Потім EM продовжує переходити до наступного кроку з використанням оновленого набору параметрів.

Підкреслюємо, що обидва ці методи потребують висновування по мережі, і вони значно повільніші за навчання параметрів із повністю спостережуваними даними. Більше того, обидва алгоритми намагаються оптимізувати функцію мультимодальної вірогідності, і жоден з них не гарантує, що глобальний оптимум буде знайдений. На практиці зазвичай потрібно кілька перезавантажень, щоб зменшити ризик застрягти в локальних оптимах.

2.4. Навчання структури з частково спостережуваних даних

У випадку повних даних використовували функцію оцінювання (наприклад, баєсівська оцінка) для оцінки кандидатів. Функція оцінювання була ефективною для обчислення, оскільки вона розкладалася на суму термів для кожної сім'ї (вузол та його батьки). Але якщо маємо загублені дані, то це вже не працюватиме.

Приблизні методи оцінки намагаються наблизити Баєсівську оцінку, незважаючи на загублені дані. Обчислення такого наближення Баєсівської оцінки займає значно більше часу, ніж обчислення Баєсівської оцінки для повних даних. Більше того, не можемо кешувати та повторно використовувати оцінки для різних сімей, оскільки приблизна оцінка не розкладається. З цієї причини, прямий метод пошуку та оцінки для загублених даних був би надто повільним.

Структурна ЕМ - це ітеративний алгоритм, подібний до ЕМ, який включає кроки для зміни структури. Починаємо з початкової структури і чергуємо два етапи. На Е-кроці Е використовуємо поточну структуру для обчислення очікуваного значення функції правдоподібності для частково спостережуваного набору даних. Це вимагає виконання логічного висновування для кожного екземпляра в наборі даних. На М-кроці використовуємо цю статистику для вдосконалення структури та параметрів нашої моделі. М-крок може використовувати будь-який алгоритм навчання структури для повних даних.

2.5. Навчання із прихованими змінними

Якщо знаємо про існування прихованої змінної, але вона ніколи не спостерігається у нашому навчальному наборі даних, то це лише надзвичайний екземпляр навчання з частково спостережуваними даними. Більш цікавий випадок, коли ми навіть не знаємо, що існує прихована змінна. Це може статися, якщо є якась відповідна змінна, яку нехтували включити до моделі, або якщо є якесь природне явище, про яке не усвідомлювали, створюючи модель. Було б корисно, якби алгоритми навчання могли ввести приховані змінні при необхідності.

Структурна ЕМ іноді може включити одну або дві приховані змінні, але лише якщо заздалегідь скажемо, скільки прихованих змінних потрібно шукати. Загалом проблема навчання із прихованими змінними залишається невирішеною.

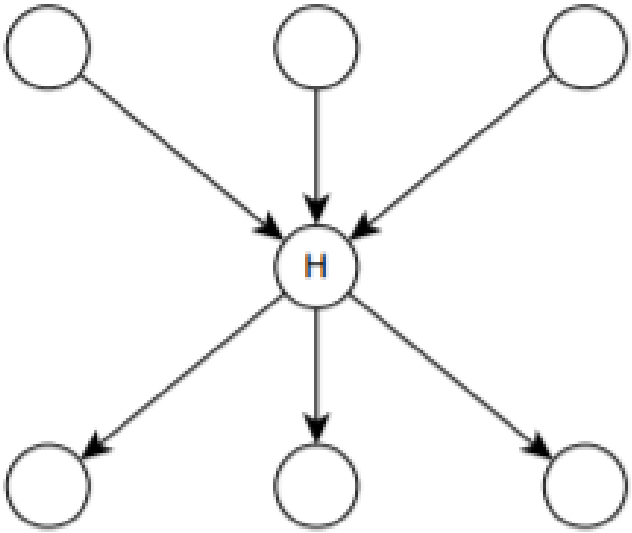

Рисунок 6: Модель із прихованою змінною H.

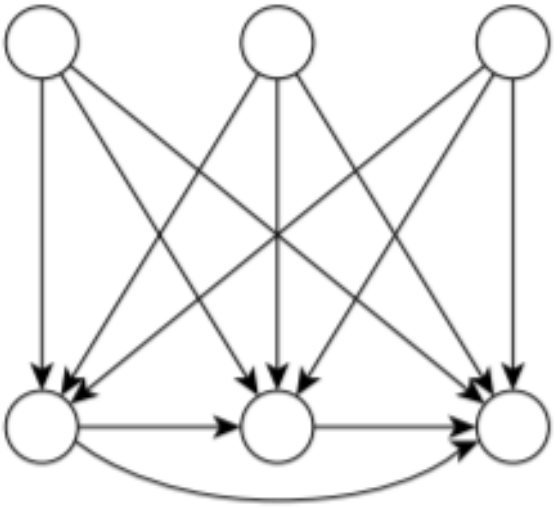

Рисунок 7: Модель без використання прихованої змінної.

РОЗДІЛ 3. Дослідження пов'язаних робіт

У цьому розділі досліджуємо літературу про навчання структури, акцентуючи увагу на навчанні з повних даних та навчанні за допомогою еволюційних алгоритмів. Для стислості пропускаємо безліч інших способів навчання структури, таких як тести на умовну незалежність та пошук класів еквівалентності.

3.1. Навчання структури БМ

K2 - один з найбільш ранніх алгоритмів для навчання структури Баєсівської мережі, був представлений Купером і Герсковіцом 1992 році. Їх алгоритм вимагає впорядкування вузлів. Для кожного вузла алгоритм додає батьківський вузол, який дає найбільше поліпшення в оцінці сім'ї. Коли вже не має батька, який би збільшував оцінку, або коли досягнуто максимальної кількості батьків, алгоритм переходить до вибору батьків для наступного вузла.

Локальний пошук (пошук зі сходженням до вершини - hill climbing) над структурами мережі Баєсів запропонували Chickering, Geiger та Heckerman у 1995 році.

Алгоритм вдосконалює мережу ітеративно, виконуючи локальні модифікації: додавання ребра, видалення ребра або перевертання ребра (реверсування його напрямку). У мережі з n вузлами та e краями є $n\ (n - 1) - e$ ребра, які ми могли б додати, e ребра, які ми могли б видалити, і e ребра, які ми могли перевернути. Деякі з цих модифікацій є незаконними, оскільки вводять цикли. Дельта-оцінка модифікації - це зміна оцінки, викликана цією модифікацією в поточній мережі. У кожній ітерації алгоритм обчислює дельту-оцінку для кожної легальної модифікації та застосовує одну модифікацію з найвищою позитивною дельтовою оцінкою. Якщо оцінку можна розкласти, багато дельтових оцінок залишаються незмінними від ітерації до ітерації, та їх не потрібно перераховувати.

На практиці алгоритм пошуку зі сходженням до вершини для навчання структур БМ працює дуже добре і широко використовується.

Також у 1995 році Сінгха та Вальторта представили ітеративний алгоритм (iterative algorithm) для навчання структур БМ. Їх алгоритм складається з декількох фаз. Перша: використовуються тести на умовну незалежність, щоб створити неорієнтований граф змінних. Потім використовується евристика для орієнтування ребер цього графа і, таким чином, отримання впорядкування вузлів. Нарешті, він запускає K2, використовуючи отриманий вузол впорядкування для навчання структури БМ. Цей процес повторюється, збільшуючи порядок випробувань на умовну незалежність на першій фазі, поки оцінка, дана K2 на останній фазі, не перестане покращуватися. На відміну від K2, алгоритм Сінгха та Вальторта не потребує впорядкування вузлів як вхідних даних.

У 2003 р. Мур та Вонг представили алгоритм оптимального повторного введення (the Optimal-Reinsertion algorithm) для навчання структури БМ. Цей алгоритм особливо корисний для великих наборів даних (десятки тисяч записів). Алгоритм кілька разів видаляє вузол з мережі та повторно вставляє його з оптимальним набором батьків та дітей. Мур і Вонг встановили, що їх алгоритм був швидшим, ніж алгоритм пошуку зі сходженням до вершини, і створили структури з кращою оцінкою (і кращим узагальненням), особливо коли в пошуковому просторі було багато локальних мінімумів.

У 2005 році Тейссьє та Коллер представили ще один алгоритм навчання структур БМ. Замість пошуку по простору структур їх алгоритм здійснює пошук по простору порядків вузлів. Алгоритм використовує той факт, що найкраща оцінка БМ, що відповідає певному порядку, може бути знайдена у поліноміальний час. Для обчислення необхідної великої кількості достатньої статистики вони використовували дерево AD. Вони також обрізали набір можливих батьків для кожного вузла, щоб ще більше скоротити час роботи.

3.2. Навчання структури БМ за допомогою еволюційних алгоритмів

У 1996 році Larrañaga та ін. представили генетичний алгоритм (genetic algorithm) навчання структури БМ. Вони представляють мережеві структури як матриці суміжності. При заданому упорядкуванні вузлів, мутація та кросовер завжди генерують дійсну структуру (без циклів). Без заданого впорядкування вузлів алгоритм використовує оператор відновлення, який випадковим чином видаляє ребра з циклів, поки не буде задоволено властивість спрямованого ациклічного графа. Вони обмежують максимальну кількість батьків для будь-якого вузла до чотирьох. Якщо кросовер або мутація призводять до структури, що порушує це обмеження, вони використовують локальний оптимізатор для вибору чотирьох батьків, які максимізують логарифмічну правдоподібність.

Також у 1996 р. Larrañaga та ін. представили ще один генетичний алгоритм для навчання БМ структури, де вони еволюціонують порядок вузлів, а потім використовують алгоритм K2 для навчання структури за цим порядком. Вони порівнюють численні оператори кросовера та мутації. Серед кросовер-операторів найкраще працює циклічний кросовер. Кілька операторів мутації працюють порівняно добре. Larrañaga та ін. оцінюють їх алгоритм, намагаючись відновити структуру мережі сигналізації з набору даних 3000 випадків. Вони порівнюють свої оцінки мережі з базовою лінією, отриманою за допомогою запуску K2, з правильним упорядкуванням вузлів, і їх генетичний алгоритм не перевершує цю базову лінію.

У 2003 році Бланко та ін. порівняли три еволюційні алгоритми навчання структур БМ за двох умов. Вони представляють мережу як матрицю суміжності. У першій умові відомий порядок вузлів, а матриця суміжності верхньотрикутна. У другій умові порядок вузлів невідомий, і алгоритми можуть видавати недійсних екземплярів, вимагаючи використання оператора відновлення. Blanco та ін. порівняли традиційний генетичний алгоритм з двома алгоритмами оцінки розподілу (Estimation of distribution algorithms). Алгоритм універсального

граничного розподілу (UMDA) передбачає, що кожен біт розподіляється незалежно, і оцінює розподіл кожного біта. Алгоритм інкрементального навчання на основі населення (PBIL) підтримує вектор вірогідності та оновлює його за допомогою геббського правила.

У 2004 році Вонг та ін. представив гібридний алгоритм кооперативного коефіцієнта (hybrid cooperative-coevolution algorithm) для навчання структур БМ. На першій фазі вони використовують умовні тести на незалежність, щоб виключити існування певних ребер, тим самим зменшивши простір пошуку. На другому етапі вони використовують алгоритм спільного коефіцієнта, який розбиває проблему навчання структури на підпроблеми навчання набору батьків для кожного вузла. Ці підпроблеми стають співпрацюючими "видами", які розвиваються незалежно, використовуючи біт-рядкове подання (один рядок матриці суміжності мережі). Вони використовують додаткові хитрощі, такі як модифікований кросовер-оператор (оскільки кількість батьків обмежується невеликим значенням) та приблизне впорядкування по вузлах (щоб зменшити шанс створення циклічних структур).

У 2011 році Карвальо запропонував ще один кооперативно-коеволюційний (cooperative-coevolution) генетичний алгоритм для навчання структур БМ. Він використовує дві субпопуляції ("види"): субпопуляція перестановки, що представляє порядок вузлів, і двійкову субпопуляцію, яка представляє верхньо-трикутну матрицю суміжності графа БМ, задану впорядкуванням вузла. Він використовує циклічний кросовер та мутацію зміни для субпопуляції перестановки, а двоточковий кросовер та мутацію біт-фліпу для двійкової субпопуляції. Зокрема, ці оператори закриті, це означає, що вони не можуть створити недійсну структуру. Тому оператор з відновлення не потрібен. Більше того, це представлення може кодувати будь-яку структуру БМ, без обмежень щодо кількості батьків у вузла.

3.3. Навчання структури БМ із загубленими даними або прихованими змінними

У 1997 році Сінгх запропонував ітеративний алгоритм для навчання структури та параметрів баєсівської мережі за наявності загублених даних. Його алгоритм працює, чергуючи чотири кроки. Спочатку, він створює кілька повних наборів даних шляхом вибірки загублених значень з мережі, знайдених у попередній ітерації. (У першій ітерації алгоритм відбирає вибірки з апріорного розподілу кожної змінної.) Далі, алгоритм навчає мережу для кожного набору даних, запускаючи K2 з істинним упорядкуванням вузла. По-тім, алгоритм знову приховує загублені дані і знову вивчає параметри кожної мережі за допомогою ЕМ. Нарешті, алгоритм поєднує мережі в єдину мережу, використовуючи користувацький метод синтезу. Потім алгоритм використовує отриману мережу для вибірки загублених значень на етапі першому наступної ітерації. Сінгх оцінює свій алгоритм на наборі даних із 10000 екземплярів, відібраних з мережі сигналізації, при цьому 20% та 40% втрачені дані. Він повідомляє значення для перехресної ентропії між навченою мережею та справжньою мережею.

Також у 1997 році Ramoni та Sebastiani представили ще один алгоритм навчання структури БМ із загубленими даними. Замість того, щоб використовувати вибірки ЕМ або Gibbs для "завершення" даних і отримати очікувану достатню статистику для кожної ітерації, вони використовують детермінований алгоритм під назвою " Bound-and-Collapse ". Вони стверджують, що цей метод набагато швидший, але вони оцінювали їх метод лише у невеликих мережах від 3 до 5 вузлів.

У 1999 р. Myers та ін. порівняли декілька стохастичних алгоритмів (stochastic search algorithms) пошуку навчання структури та параметрів мережі Баєса за наявності загублених даних. Вони порівнюють еволюційний алгоритм - MCMC та новий алгоритм під назвою EAMCMC, який поєднує два підходи. В ЕА вони паралельно еволюціонують загублені значення та структуру мережі. Вони представляють структуру мережі у вигляді списку суміжності та

визначають відповідні оператори - кросовер та мутація. У МСМС вони мають два окремі ланцюги, один на просторі загублених значень, а другий на просторі мережевих структур. Вони оцінюють свої алгоритми в кількох невеликих мережах від 7 до 9 вузлів.

У 2000 році Peña та ін. вдосконалили етап навчання параметрів алгоритму максимізації структурних очікувань (the Structural Expectation Maximization algorithm). Замість того, щоб використовувати ЕМ для оптимізації параметрів, вони використовують метод, який поєднує Bound-and-Collapse з ЕМ. Вони показують підвищену ефективність та точність проблеми кластеризації, де загублені дані є в одному стовпчику (змінна кластер).

3.4. Ансамблі БМ

Ансамблеві методи навчають декілька моделей та поєднують їхні прогнози, сподіваючись пом'якшити обмеження будь-якої конкретної моделі.

У 2002 році Россет і Сегал використовували підвищення оцінки щільності, а Баєсові мережі були основними слабкими класифікаторами. Таким чином, вони ефективно вивчають ансамблі БМ шляхом повторного зважування екземплярів у навчальному наборі даних.

У 2003 році Фрідман та Коллер представили методику оцінки ймовірності ознак (наприклад, чи вузол X є батьком Y) в БМ, з метою здобути знання про структуру предметної області. Їх мотивація випливає з того, що, маючи невеликий набір даних, може бути багато мережевих структур з високим апостеріорним числом. Обчислення точної ймовірності ознаки шляхом перерахування всіх можливих структур є незмінним, оскільки кількість можливих структур є надекспоненціальною за кількістю вузлів. Фрідман і Коллер ділять проблему на два етапи. Спочатку вони використовують МСМС для вибірки з простору порядку вузлів для мережі. (Вузол X може бути батьком Y, якщо X в цьому порядку передує Y.) Потім, вони обчислюють ймовірність ознаки з урахуванням вибраного порядку. Для простих ознак (таких як вибір

батьків, ребер) вони показують спосіб обчислити цю ймовірність у закритому вигляді (заданий порядок). Для більш складних особливостей (таких як шляхи) вони можуть наблизити ймовірність шляхом вибірки набору структур за відомим порядком та емпірично оцінивши ймовірність, використовуючи ці структури. Цей двоступеневий метод повільний, і для їх прискорення доводиться використовувати декілька хитрощів та наближень. Для більш складних ознак (таких як шляхи) вони можуть наблизити ймовірність шляхом вибірки набору структур за відомим порядком та емпірично оцінивши ймовірність, використовуючи ці структури. Цей двоступеневий метод повільний, і для його прискорення доводиться використовувати деякі хитрощі. Фрідман і Коллер демонструють великі результати і повідомляють про успіх, з яким їх підхід реконструює ознаки з мережі сигналізації з набором даних у 100, 500 або 1000 екземплярів.

У 2007 році Liu та ін. навчили ансамблі БМ, спершу намагаючись знайти кореневі вузли, а потім вибірку із заміною з навчального набору, виходячи із значень, які приймають кореневі вузли. Вони використовують отримані набори даних для навчання БМ, використовуючи алгоритм оптимального відновлення (the optimal-reinsertion algorithm) Мура та Вонга. Нарешті, вони поєднують отримані БМ за допомогою методу синтезу спеціального призначення (the special-purpose fusion method) та отримують єдиний БМ як результат їх алгоритму. Їх алгоритм працює краще, ніж простий метод розфасовки. Простий бегінг, в свою чергу, працює краще, ніж БМ, отримана за один цикл алгоритму оптимального відновлення.

РОЗДІЛ 4. Генетичний алгоритм над графами

У цьому розділі представляємо генетичний алгоритм для навчання структури баєсівської мережі з повністю спостережуваного набору даних. Цей алгоритм здійснює пошук по простору спрямованих ациклічних графів.

4.1. Представлення

Представлення розробленого алгоритму сильно натхнене представленням Карвальо. Індивід представлений за допомогою кортежу {R, A}, де R - перестановка, що вказує на порядок вузлів, а A - верхня трикутна матриця суміжності, рядки та стовпці якої представляють вузли в лексикографічному порядку. Рисунок 8 ілюструє це подання. Кортеж {R, A} кодує унікальний спрямований ациклічний граф (DAG) таким чином: матриця суміжності A описує неорієнтований граф над вузлами. Тоді кожне ребро S - T спрямовується у порядку R: S → T, якщо $S \overset{R}{<} T$ (S передує T в R) , або S ← T, якщо $T \overset{R}{<} S$. Зауважте, що представлення конкретної DAG не є унікальним, оскільки може бути декілька порядків, що відповідають DAG. Це називається надмірністю кодування. Наприклад, альтернативні порядки V, W, X, Y, Z і V, X, W, Z, Y обидва відповідають мережі на рисунку 8.

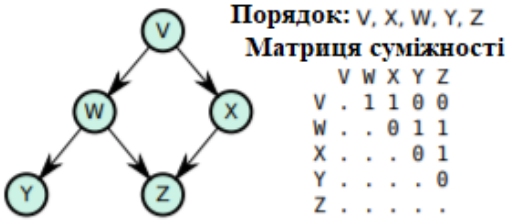

Рисунок 8: Невелика баєсівська мережа (зліва) та її представлення в генетичному алгоритмі (справа).

Існує тонка, але важлива різниця між розробленим представленням та представленням Карвальо. У даному поданні рядки та стовпці матриці суміжності A завжди однакові - вони є вузлами в лексикографічному порядку. Отже, дана матриця суміжності кодує неорієнтований граф, ребрам якого надається напрямок, використовуючи порядок R. На відміну від цього, Карвальо дозволяє порядку R визначати значення рядків і стовпців матриці суміжності A.

Його матриця суміжності являє собою спрямований граф. Представлення Карвальо було б непридатним у традиційному генетичному алгоритмі, оскільки воно має погану локальність: обмін місцями двох змінних у порядку R призводить до великих змін у мережі. Наприклад, припустимо, що порядок V, X, W, Y, Z з рисунку 8 змінилося на V, Z, W, Y, X. За даним представленням це призведе до перегортання напрямку ребер W → Z і X → Z, як показано на рисунку 9(а). З представленням Карвальо ті ж мутації призведуть до перегортання ребер X → Z, але також видалення трьох ребер та додавання трьох нових ребер, як показано на рисунку 9(б). Наше представлення уникає цієї слабкої локальності (poor locality), що перетворила б генетичний алгоритм у випадковий пошук.

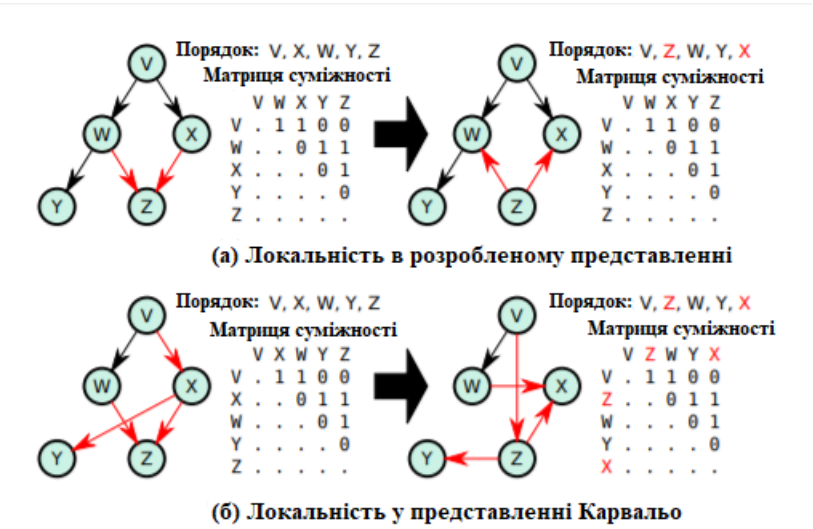

Рисунок 9: Локальність представлення генетичних алгоритмів.

4.2. Оператори

В розробленому алгоритмі ініціалізуємо індивід, надаючи йому довільну перестановку R для порядку та верхньо-трикутну матрицю суміжності A, що має по одному значенню «1» у кожному рядку. Використовуємо процедуру турнірного відбору (a tournament selection procedure) з розміром турніру 2. Це означає, що два індивіда вибираються випадковим чином, а той, який має більш високу пристосованість, вибирається для репродукції. Ця процедура повторюється, поки ми не відберемо достатньо екземплярів для репродукції.

Для кожної пари батьків {R1, A1] та {R2, A2}, вибраних для репродукції, випадково застосовуємо один з наступних операторів:

- Циклічний кросовер на порядках R1 і R2. Раніше було показано, що цей оператор має хороші показники для структурного навчання орієнтованих ациклічних графів, які здійснюють пошук за порядками вузлів. Циклічний кросовер працює, поділяючи дві перестановки на цикли. Цикл визначається як мінімальний набір значень, які разом займають однакові позиції у двох перестановках. Рисунок 10 ілюструє циклічний кросовер.
- Двоточковий кросовер на матрицях суміжності A1 і A2. Цей оператор вирівнює дві матриці суміжності в бітові рядки. Потім він обрізає обидва рядки у двох випадкових місцях та обмінюється їхніми середніми фрагментами. Рисунок 11 ілюструє двоточковий кросовер.
- Мутація зміни місцями (swap mutation) за порядками R1 і R2. Для кожного з батьків цей оператор вибирає два випадкових вузла та обмінює їх у порядку.
- Біт-фліп мутація (bit-flip mutation) на матрицях суміжності A1 і A2. Для кожного з батьків цей оператор перевертає кожен біт у матриці суміжності, який має невелику ймовірність.

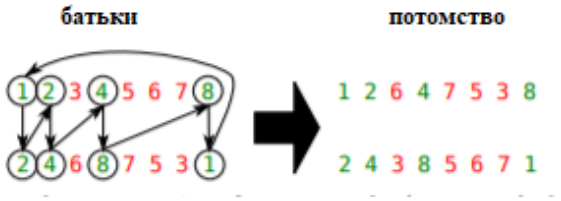

Рисунок 10: Циклічний кросовер на порядках. 1, 2, 4, 8 утворюють цикл (зелений колір), оскільки разом вони займають однакові позиції у двох батьків. Колами та стрілками показують, як знайдено цей цикл. 3, 5, 6, 7 утворюють ще один цикл (червоним колір).

Рисунок 11: Двоточковий кросовер на матрицях суміжності.

Оператори кросовера поєднують інформацію обох батьків, тоді як оператори мутації цього не роблять. Два оператори на порядок перевертають ребра в мережі: кросовер перевертає багато, а мутація перевертає декілька. Два оператори на матрицях суміжності додають, видаляють або обмінюються ребрами між мережами.

Біт-фліп мутація Карвальо перевертає біти випадковим чином, незалежно від їх початкових значень. Оскільки більшість орієнтованих ациклічних графів, розглянутих у процесі пошуку, є рідкісними, то матриця суміжності містить більше нулів, ніж одиниць, тому цей оператор мутації швидше переверне 0 у 1, ніж зробить протилежне. Це означає, що оператор частіше додає ребра, ніж видаляє ребра. В розробленому алгоритмі зробили мутацію біт-фліпу симетричною, враховуючи початкове значення біта. Це означає, що оператор мутації з однаковою ймовірністю як додасть ребро, так і видалить ребро.

4.3. Кешування

Використовуємо два рівні кешування для прискорення обчислення байєсівської оцінки мережі. Перший рівень кешує достатню статистику (рахує) набір даних. Кеші другого рівня встановлюють набір оцінок, які будуть визначені незабаром.

Перший рівень кешування використовує структуру даних AD-дерева Мура та Лі для прискорення обчислення таблиць спряженості. AD-дерево економить час за рахунок використання більшої кількості пам'яті. Це структура даних дерева, де кожен вузол "розбиває" набір даних відповідно до всіх можливих присвоєнь змінній.

Таблиці спряженості можна обчислити з дерева AD в часі, що не залежить від загального розміру набору даних. Таким чином, нам більше не потрібен один повний прохід через набір даних для обчислення таблиці спряженості.

Традиційні методи пошуке зі сходженням до вершини кешують оцінки кожної сім'ї, щоб їх можна було отримувати знову і знову, не починаючи перерахунки. Вдосконалюємо це потрохи, помічаючи, що з апріорною еквівалентною формою Баєса Діріхле (BDeu) сімейна оцінка розкладається на два множинні показники:

$$\text{FamScore}(X_i, \text{Pa}_{X_i}) = \text{SetScore}(\text{Pa}_{X_i} \cup \{X_i\}) - \text{SetScore}(\text{Pa}_{X_i}),$$

де $$\text{SetScore}(\boldsymbol{X}) = \sum_{\boldsymbol{v}_i \in Val(\boldsymbol{X})} \ln \frac{\Gamma(\alpha_{\boldsymbol{X}})}{\Gamma(\alpha_{\boldsymbol{X}} + M[\boldsymbol{v}_i])}$$

для будь-якого набору змінних X, а $\alpha_X = \alpha \cdot \frac{1}{|Val(X)|}$. Для обчислення сімейної оцінки, схоже, потрібні таблиці спряженості на випадок $Pa_{X_i} \cup \{X_i\}$ та Pa_{X_i}. Але насправді останню таблицю спряженості можна отримати з попередньої, маргіналізуючи стовпчик X_i. Це означає, що для обчислення оцінки сім'ї потрібен лише один запит AD-дерева.

Другий рівень зберігає набір оцінок. Це призводить до більшої кількості кешу, оскільки кілька сімей протягом життя алгоритму можуть мати один і той же набір батьків. Наприклад, припустимо, що хочемо обчислити FamScore (X, {A, B, C}) і пізніше FamScore (Y, {A, B, C}). У другому обчисленні SetScore ({A, B, C}) буде відновлено з кеша.

4.4. Результати експериментів

Запускаємо генетичний алгоритм з такими параметрами :

Параметр	Значення
prior strength α	1
population size	200
elite size	20
stopping condition	1000 generations
probability of order crossover	0.1
probability of order mutation	0.4
probability of adjacency-matrix crossover	0.1
probability of adjacency-matrix mutation	0.4
tournament size for selection	2

Використали чисельність в 200 і автоматично скопіювали кращих 20 екземплярів («еліту»). Для кожної пари батьків, обраної для репродукції, застосували лише один з чотирьох операторів, відповідно до вищезгаданих ймовірностей. Якщо вибрана мутація матриці суміжності, кожен біт перевертається з невеликою ймовірністю. Налаштували ці параметри вручну, грунтуючись на інтуїції стрибків в просторі рішення, які виконує кожен оператор. Можливо, можна й покращити вибір цих параметрів.

На рисунку 12 показані результати десяти запусків генетичного алгоритму в мережі сигналізації, з навчальним вибіркою розміром 1000. У перші 200 поколінь алгоритм додає багато ребер і оцінка швидко покращується. У наступні 200 поколінь алгоритм видаляє деякі ребра і оцінка продовжує швидко зростати. Оцінка не змінюється приблизно ще 600 поколінь. Стандартне відхилення все ще далеко від нуля навіть після 1000 поколінь, що свідчить про те, що різні прогони виявляють різні локальні оптимуми. У середньому найкраща знайдена мережа має трохи меншу оцінку, ніж справжня мережа, і має трохи більше ребер, ніж справжня мережа. Тим не менше, найкраща мережа, знайдена в декількох запусках, перевищує оцінку справжньої мережі.

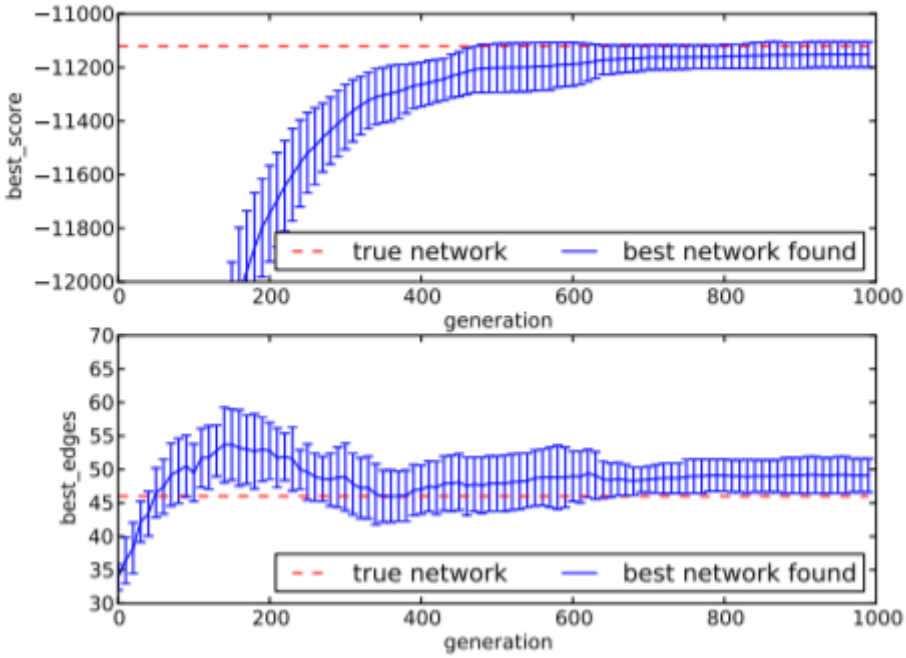

Рисунок 12: Десять запусків генетичного алгоритму.

Результат: показуємо оцінку (вгорі) та кількість країв (внизу) найкращого індивіда в кожному поколінні. Результати усереднюються протягом 10 запусків, при цьому рядки помилок показують стандартні відхилення. Найкраща мережа, знайдена за 10 запусків, мала оцінку -11090.56 та 46 ребер. Справжня мережа мала оцінку -11120,34 та 46 ребер.

На рисунку 13 показано десять запусків того ж алгоритму, якщо біт-фліп мутація на матриці суміжності не враховує поточне значення біта. Як вже згадували раніше, ця версія оператора набагато частіше може додати ребро, ніж видалити ребро. Діаграма підтверджує, що без симетричної корекції алгоритм додає занадто багато ребер на початку, а потім витрачає час, видаляючи їх повільно. Тим не менш, алгоритм знаходить досить хороші мережі після 1000 ітерацій. Здається, що додавання ребер цінніше для ранніх поколінь, а видалення ребер цінніше для наступних поколінь. Це говорить про те, що варто спробувати адаптувати ймовірності оператора в міру збільшення покоління.

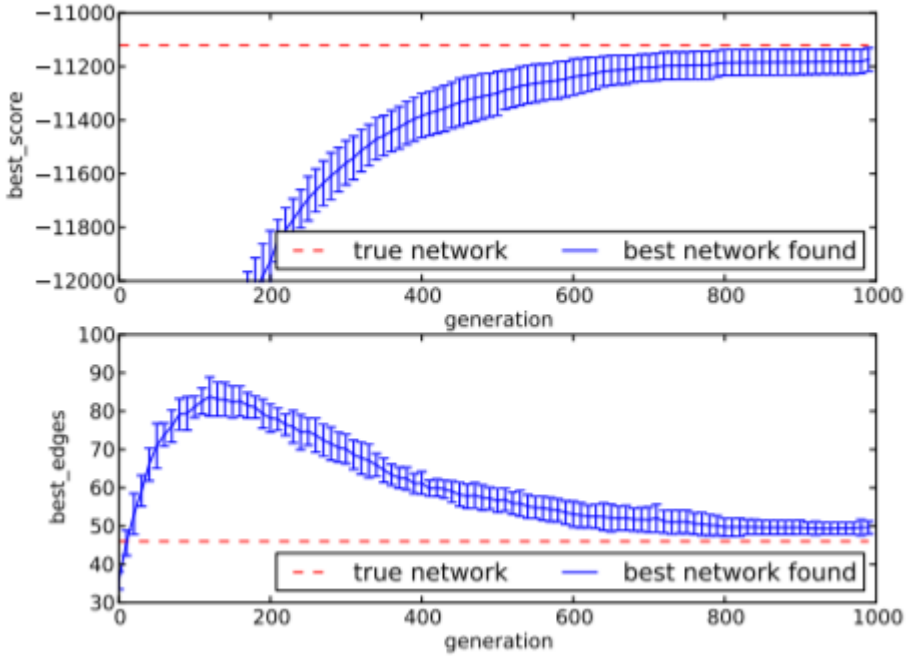

Рисунок 13: Десять запусків без симетричної корекції.

Результат: верхня ділянка (оцінка) має ту саму шкалу, що і на попередньому малюнку, але нижня ділянка (кількість ребер) вимагає більш широкого діапазону. Найкраща знайдена мережа мала оцінку -11128,59 та 47 ребер. Справжня мережа мала оцінку -11120,34 та 46 ребер.

Під час експерименту також спробували змістити оператор мутації, щоб з більшою ймовірністю видаляти ребра ніж додавати їх. Ця варіація алгоритму надто агресивно видаляла ребра, що призводило до гірших результатів мережі.

Висновки

Як перший внесок, надали ретельний огляд та опис літератури навчання баєсівської структури мережі (розділи другий та третій). Другий внесок - розробка генетичного алгоритма (GA) для навчання структури Баєсової мережі. Алгоритм (розділ четвертий) здійснює пошук по простору графа, використовує оператори мутації та кросовера. Його можна використовувати як швидкий спосіб навчити структуру БМ з якомога меншими обмеженнями.

Підсумовуємо висновки з даної роботи нижче:

- До проблеми вивчення структури з повністю спостережуваних даних можна підходити двома способами: шляхом пошуку DAG безпосередньо або шляхом пошуку лише порядку вузла, а потім обчислити найкращий DAG для цього конкретного порядку. Якщо заздалегідь відомо про гарний порядок для набору даних, то знайти найкращий DAG для цього порядку набагато простіше, ніж знайти найкращий DAG загалом. Незалежно від того, шукаємо ми загальну DAG або DAG з обмеженим порядком, кешування та повторне використання результатів є вирішальними для забезпечення високої ефективності.
- Однією з найбільших проблем, з якою стикалися під час роботи, була оцінка ефективності алгоритмів навчання структури. З дослідження літератури, схоже, інші дослідники також натрапляли на цю проблему дуже часто. Існує чимало програмних пакетів, які стверджують, що вони навчаються структури, але не ясно, наскільки добре вони працюють. Багато алгоритмів, які заявляють, що є сучасними, не мають загальнодоступного вихідного коду. З цих причин багато робіт не мають цілком переконливої оцінки. Робота Мура та Вонга про оптимальне відновлення має найбільш ретельну оцінку, яку зустрічали досліджуючи літературу.

Список використаної літератури

1. Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, San Mateo, California.
2. David Heckerman's tutorial on learning with Bayesian networks.
3. Ф. Дженсен. "Вступ до баєсівських мереж". UCL Press. 1996.
4. Ф. В. Дженсен. "Графи баєсівських мереж та рішень". Спрингер. 2001.
5. Doctor J. N., Strylewicz G. (2010): Detecting ’wrong blood in tube’ errors: Evaluation of a Bayesian network approach. Artificial Intelligence in Medicine 50, p. 75–82.
6. G. F. Cooper and E. H. Herskovits. A Bayesian method for the induction of probabilistic networks from data. Machine Learning, 9:309–347, 1992.
7. Murphy K. (1998): A Brief Introduction to Graphical Models and Bayesian Networks.
8. R. Bowden and D. Turkington. Instrumental Variables. Cambridge University Press, New York, 1984.
9. RG Cowell, AP Dawid, SL Lauritzen та DJ Spiegelhalter. "Імовірнісні мережі та експертні системи". Спрингер-Верлаг. 1999.
10. D. Koller and N. Friedman. Probabilistic Graphical Models: Principles and Techniques. MIT Press, 2009.
11. Andrew Moore and Weng keen Wong. Optimal reinsertion: A new search operator for accelerated and more accurate bayesian network structure learning. In In Proceedings of the 20th International Conference on Machine Learning (ICML ’03, pages 552–559. AAAI Press, 2003.
12. С. Лауріцен. "Графічні моделі", Оксфорд. 1996. Остаточне математичне виклад теорії графічних моделей.
13. Pedro Larrañaga, Mikel Poza, Yosu Yurramendi, Roberto H. Murga, and Cindy M.H. Kuijpers. Structure learning of bayesian networks by genetic algorithms: A performance analysis of control parameters. IEEE Transactions on Pattern Analysis and Machine Intelligence, 18:912–926, 1996.

14. J. M. Peña, J. A. Lozano, and P. Larra˜naga. An improved bayesian structural em algorithm for learning bayesian networks for clustering. Pattern Recogn. Lett., 21(9):779–786, July 2000.
15. Marco Ramoni and Paola Sebastiani. Learning bayesian networks from incomplete databases. In Proceedings of the Thirteenth conference on Uncertainty in artificial intelligence, UAI'97, pages 401–408, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.
16. uk.wikipedia.org/wiki/Баєсова_мережа – 09.11.2019.
17. Moninder Singh and Marco Valtorta. Construction of bayesian network structures from data: a brief survey and an efficient algorithm. In International Journal of Approximate Reasoning, pages 259–265. Morgan Kaufmann, 1995.
18. Kohsuke Yanai and Hitoshi Iba. Estimation of distribution programming based on Bayesian network. In Ruhul Sarker, Robert Reynolds, Hussein Abbass, Kay Chen Tan, Bob McKay, Daryl Essam, and Tom Gedeon, editors, Proceedings of the 2003 Congress on Evolutionary Computation CEC2003, pages 1618–1625. IEEE Press, 8-12 December 2003.
19. http://www.opensourceinitiative.net/edu/bnintro/ - 19.01.2020.
20. M. Teyssier and D. Koller. Ordering-based search: A simple and effective algorithm for learning bayesian networks. In Proceedings of the Twenty-first Conference on Uncertainty in AI (UAI), pages 584–590, Edinburgh, Scottland, UK, July 2005