САМООРГАНІЗОВНА ІНКРЕМЕНТНА НЕЙРОННА МЕРЕЖА ДЛЯ КЛАСТЕРУВАННЯ МАСИВІВ ДАНИХ

Іванова Є.В., студентка групи КН-14-5,
e-mail: yevheniia.ivanova@nure.ua.
Науковий керівник: к.т.н., ст. викл. ШІ Дейнеко А.О.
Харківський національний університет радіоелектроніки

In the paper self-organizing incremental neural network for multidimensional data sets. It is proposed to accomplish the task of clustering and represent the topological structure of the input data. Self-organizing incremental neural network find typical prototypes for large-scale data set and robust to noise. Automatically learn number of classes of input data, so clustering task solves with no prior knowledge. The proposed neural network separates clusters with high-density overlap. It adopts a two-layer neural network structure.

Однією з задач навчання без вчителя є кластерування масивів даних. У статті запропоновано самоорганізовну інкрементну нейронну мережу для вирішення цієї проблеми. Також вона застосовується для відображення топологічної структури вхідних даних. Існує декілька підходів для вирішення останньої задачі. Наприклад, алгоритм на основі самоорганізовної мапи Т. Когонена є методом проектування багатовимірного простору в простір з більш низькою розмірністю з визначеною структурою. Суттєвим недоліком є те, що зменшується розмірність вихідної задачі і остаточний результат роботи нейронних мереж залежить від початкових установок мережі. У зв’язку з цим, виникають дефекти проектування, аналіз яких є досить складною задачею.

Альтернативою цьому підходу є поєднання конкурентного навчання Хебба і нейронного газу. Це поєднання більш ефективно в побудові топологічної структурі, але практичному застосуванню цього підходу перешкоджає ряд проблем: необхідні апріорні знання про розмір мережі і складність застосування методів адаптації швидкості навчання мережі, зайва адаптація призводить до зниження ефективності при навчанні новими даними, а надто повільна швидкість адаптації викликає високу чутливість до

збурених даних. Для задач онлайн навчання, перераховані вище методи не підходять. Фундаментальна проблема для таких завдань – це адаптація мережі до нової інформації без пошкодження або знищення вже відомої. У доповіді розглядається алгоритм SOINN, який частково вирішує визначенні проблеми.

SOINN (self organizing incremental neural network) [1] використовує двошарову конкурентну мережу (рис. 1). Перший шар аналізує щільність розподілу вхідних даних і використовує вузли та ребра для представлення цього розподілу. Другий шар розділяє дані на кластери шляхом розташування області вхідних даних з низькою щільністю та використовує менше вузлів ніж перший шар. Коли навчання другого шару завершено, SOINN підлаштовує кількість кластерів і видає прототипи вузлів кожного кластера. Він також використовує аналогічний алгоритм навчання для першого та другого шарів.

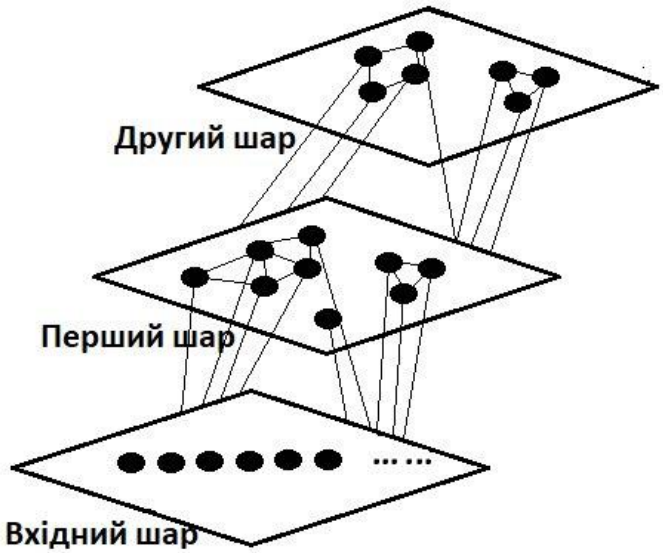

Рисунок 1 – Архітектура SOINN

Коли вхідний вектор подається на вхід SOINN [1, 2], він знаходить найближчий вузол (переможець) і другий найближчий вузол (другий переможець) вхідного вектору. Потім

використовуючи порогові критерії подібності мережа визначає, чи належить вхідний вектор кластеру першого переможця або другого.

На основі вищесказаного, ми можемо дати формальний опис алгоритму навчання для мережі SOINN. Цей алгоритм використовується для навчання і першого і другого шарів мережі. Різниця полягає лише в тому, що вхідні дані для навчання другого шару породжуються першим шаром і при навчанні другого шару, ми маємо знання про топологічної структурі першого шару, для обчислення постійного порога подібності. Опис алгоритму:

1. Ініціалізувати множину вузлів A двома вузлами, c1 і c2: A = {c1, c2}. Ініціалізувати множину ребер C порожньою множиною: C = {}.
2. Подати на вхід новий сигнал x з R^n.
3. Знайти в множині A вузли переможця s1 і другого переможця, як вузли з найближчим і наступним за ним векторами ваг (за деякою заданою метрикою). Якщо відстань між x, s1 і s2 більше порогів подібності Ts1 або Ts2, то створити новий вузол і перейти на крок 2.
4. Якщо ребро між s1 і s2 не існує, то створити його. Встановити вік ребра між ними рівним 0.
5. Збільшити вік всіх дуг, що виходять з s1 на 1.
6. Додати відстань між вхідним сигналом і переможцем до локальної сумарної помилки Es1.
7. Збільшити локальну кількість сигналів вузла s1: Ms1 = Ms1 +1.
8. Адаптувати вектора ваг переможця і його прямих топологічних сусідів: Ws1 = Ws1 + e1(t)(x – Ws1), Wsi = Wsi + e2(t)(x – Wsi), де e1(t) і e2(t) – коефіцієнти навчання переможця і його сусідів.
9. Видалити ребра з віком, вище заданого порогового значення.
10. Якщо число генеруються досі вхідних сигналів кратно параметру lambda, вставити новий вузол і видалити вузли в областях низької щільності.

11. Після довгого часу learning threshold кожному класу з вхідних даних буде відповідати компонента пов'язаності в побудованому графі. Оновити мітки класів.
12. Перейти до пункту 2 і продовжити навчання. Якщо це другий шар і кількість шагів дорівнює λ, то зупинитися.

Перший шар SOINN адаптивно оновлює поріг подібності кожного вузла, оскільки розподіл вхідних даних невідомий. Якщо вузол i має сусідні вузли, то поріг подібності T_i обчислюється з використанням максимальної відстані між вузлом i та сусідніми вузлами:

$$T_i = \max_{j \epsilon N_i} \|W_i - W_j\|.$$

У цьому випадку N_i – множина сусідніх вузлів, а W_i – векторні ваги вузла i. Поріг схожості T_i визначається як мінімальна відстань між вузлом та іншими вузлами в мережі, якщо вузел i не має сусідніх вузлів.

$$T_i = \min_{j \epsilon N \setminus \{i\}} \|W_i - W_j\|,$$

де N – множина всіх вузлів.

Вхідний вектор визначається в мережі як новий вузол для представлення першого вузла нового класу, якщо відстань між вхідним вектором і переможцем або другим переможцем більше, ніж поріг схожості переможця або другого переможця. Якщо вхідний вектор визначений як належний до одного кластеру як у переможця або другого переможця і якщо немає ребра, що з'єднує переможця і другого переможця. Тоді з'єднуємо переможця і другого переможця за допомогою ребра і встановлюємо вік ребра рівним нулю. Надалі збільшуємо вік всіх ребер, пов'язаних з переможцем, на одиницю.

Після цього оновлюється вектор ваги переможця та його сусідні вузли. Використаємо i для позначення вузла переможця і M_i, щоб показати час, коли він став переможцем. Зміна ваги переможця

ΔW_i і зміна ваги сусіднього вузла j $(\in N_i)$ для i ΔW_j визначаються як: $\Delta W_i = \frac{1}{M_i}(W_s - W_i)$ і $\Delta W_i = \frac{1}{100M_i}(W_s - W_i)$, де W_s – вага вхідного вектору. Якщо вік одного ребра більше заданого параметра age_{max}, то це ребро видаляється.

За фактом, тому що поріг подібності першого шару SOINN оновлюється адаптивно, помилка накопичення буде не висока. Тому вставка в межах класу є мало корисної. Вставка в межах класу для першого шару не потрібна.

Якщо результати вивчення першого шару були змінені, усі вивчені результати другого шару будуть зруйновані, таким чином потрібно перекласифікація другого шару. Другий шар SOINN є невідповідним для поступового онлайн навчання.

Після λ ітерацій (λ є таймером) SOINN [2] вставляє нові вузли в положення, де накопичена помилка є великою. Потім SOINN знаходить вузли, кількість сусідів яких менша або дорівнює 1 і видаляє ці вузли. Після того, як LT (learning threshold) [1, 3] аналізує ітерації першого шару, результати навчання використовуються як вхід для другого шару. Другий шар SOINN використовує той же алгоритм навчання, що і перший.

Запропоновано самоорганізовну інкрементну нейронну мережу для кластерування і аналізу топологічної структури даних.

Література

1. S. Furao An enhanced self-organizing incremental neural network for online unsupervised learning / / S. Furao, T. Ogura, O. Hasegawa – Neural Networks, 20 (2007), 893-903.

2. S. Furao A fast nearest neighbor classifier based on self-organizing incremental neural network / / S. Furao, O. Hasegawa – Neural Networks, 21 (2008), 1537-1547.

3. S. Furao An incremental network for on-line unsupervised classification and topology learning / / S. Furao, O. Hasegawa – Neural Networks 19 (2006) 90–106.