МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра математического обеспечения и применения ЭВМ

ОТЧЕТ

по научно-исследовательской работе

ТЕМА: ПРИМЕНЕНИЕ НЕЙРОСЕТЕВОГО ПОДХОДА К КЛАСТЕРНОМУ АНАЛИЗУ РАСПРЕДЕЛЁННЫХ ДАННЫХ

Студент гр. 3304 _____ Итальянцев Я.В.

Руководитель _____ Борисенко К.А.

Санкт-Петербург
2018

ЗАДАНИЕ
НА НАУЧНО-ИССЛЕДОВАТЕЛЬСКУЮ РАБОТУ

Студент Итальянцев Я.В.

Группа 3304

Тема НИР: Применение нейросетевого подхода к кластерному анализу распределённых данных

Задание на НИР:

Анализ алгоритмов кластеризации: их описание, проблемы, связанные с анализом распределенных данных. Обзор релевантных работ по проблематике применения нейросетей к распределенным данным.

Сроки выполнения НИР: 25.10.2018 – 20.12.2018

Дата сдачи отчета: 20.12.2018

Дата защиты отчета: 20.12.2018

Студент(ка) _____ Итальянцев Я.В.

Руководитель _____ Борисенко К.А.

АННОТАЦИЯ

Произвести анализ существующих алгоритмов и подходов в решении задачи. Описание алгоритмов и проблемы возникающие при работе с распределёнными данными. С приведением списка решений на основе нейросетевой кластеризации и схожих работ. Выводы по проделанной работе представляют краткий анализ того, что было достигнуто в результате выполнения НИР.

SUMMARY

Perform analysis of existing algorithms and approaches in solving the problem. The description of the algorithms and problems arising when working with distributed data are given. The following is a list of solutions based on neural network clustering and related work. The findings of the work done provide a brief analysis of what has been achieved as a result of the implementation of research and development.

СОДЕРЖАНИЕ

Оглавление

ВВЕДЕНИЕ

Проблема кластеризации уже давно имеет множества решений, десятки подходов и их модификации могут быть применены для данной задачи, но только единицы (Self-Organizing Map, Adaptive Resonance Theory, Learning Vector Quantization) [1] могут применятся в обучение нейронных сетей без учителя. И всё это в совокупности работы с распределёнными данными, представляет из себя сложную задачу, в которой нужно обеспечить следующие аспекты: общий формат данных для кластеризации; конфиденциальность данных, передаваемых с локальных сайтов центральному.

В ходе исследования произведены обзоры на разную литературу и аналоги, описанные в статьях, на общие основы нейросетевой кластеризации и посвящённые проблематике распределённых данных. Было обращено особое внимание на уже функционирующие проекты такие как: partSOM, MDP, которые также являются средствами для обучения и эксплуатации нейронных сетей.

Исследование позволило выделить важные моменты необходимые при разработке и выполнении цели ВКР, с определением части инструментария для будущей работы.

1. ОПИСАНИЕ АЛГОРИТМОВ КЛАСТЕРИЗАЦИИ

В данной главе описываются алгоритмы нейросетевой кластеризации которые рассматривались для возможного решения поставленной задачи. Будет произведён анализ и выделены достоинства и недостатки при их работе с распределёнными данными.

1.1. Описание работы алгоритмов.

Основная идея работы алгоритма с описанием работы.

Представлена таблица с кратким описанием достоинств и недостатков алгоритмов, с примечанием о их взаимодействии с распределёнными данными.

Таблица. 1. Анализ методов кластеризации

Название метода	Достоинства	Недостатки	Примечание
Самоорганизующаяся карта Кохонена	Устойчивость к зашумленным данным, неуправляемое обучение, быстрое обучение, возможность визуализации, возможность упрощения многомерной структуры	Эвристичность алгоритма обучения, предопределенность числа кластеров	Если новые распределённые данные для кластеризации производятся постоянно, то метод не сможет это обработать.
Адаптивная резонансная теория	Решает дилемму стабильности-пластичности	Невозможность кластеризации зашумленных данных, невозможность определения кластеров если они наложены друг на друга, необходимость управления ростом сети	Распределенные нейронные сети ART сочетают в себе стабильные возможности быстрого обучения универсальных систем ART с возможностями

			помехоустойчивости и сжатия кода многослойных персептронов
Расширяющийся нейронный газ (GNG)	Позволяет определить топологию входных данных без какой-либо априорной информации об этих данных	Большое количество гиперпараметров, которые в некоторых случаях тяжело подобрать, неспособность отслеживать нестационарные ряды с быстро меняющимися характеристиками	Конкретной проблематики, связанной с распределёнными данными найдено не было.

1.1.1. Self-Organizing Maps of Kohonen

SOM – это искусственная нейронная сеть, основанная на обучении без учителя. В картах самоорганизации нейроны помещены в узлах решетки, обычно одно- или двумерной. Все нейроны этой решетки связаны со всеми узлами входного слоя. SOM преобразует непрерывное исходное пространство X в дискретное выходное пространство A. $Ф:X \rightarrow A$

Сеть распознает кластеры в обучающих данных и распределяет данные по соответствующих кластерах. Если дальше сеть встречается с набором данных, непохожим ни на одним из известных образцов, она относит его к нового кластеру. Если в данных содержатся метки классов, то сеть способна решать задачи классификации.

Сеть Кохонена имеет всего два слоя: входной и выходной, ее называют самоорганизованной картой. Элементы карты располагаются в некотором пространстве - как правило двумерном.

Сеть Кохонена учится методом последовательных приближений. Начиная со случайным образом выбранного выходного расположения

центров, алгоритм постепенно улучшается для кластеризации обучающих данных.

Основной итерационный алгоритм Кохонена последовательно проходит ряд эпох, на каждой эпохе обрабатывается один обучающий пример. Входные сигналы (векторы действительных чисел) последовательно предъявляются сети, желаемые выходные сигналы не определяются. После предъявления достаточного числа входных векторов, синаптические весы сети определяют кластеры. Весы организуются так, что топологически близкие узлы чувствительны к похожим входным сигналам.
Для реализации алгоритма необходимо определить меру соседства нейронов (окрестность нейрона-победителя).

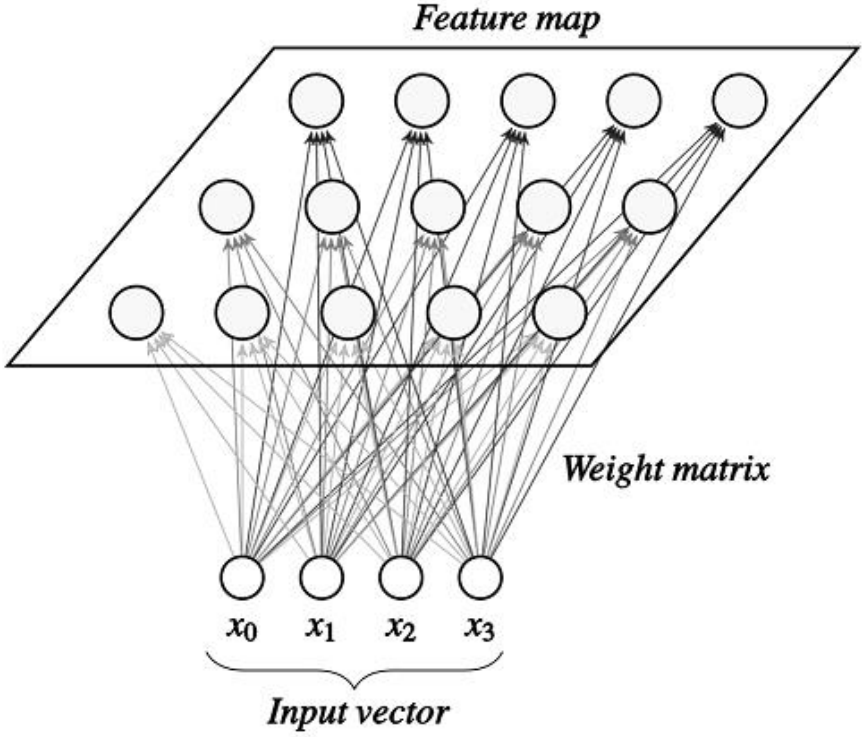

Рис. 1. Представление двухслойной нейронной сети Кохоненна.

Обучение сети состоит из трех основных процессов: конкуренция, кооперация и адаптация.

Алгоритм 1 (стандартный алгоритм обучения SOM):
Вход: X, η, σ, γ, α, E

1. Инициализировать W
2. E раз:
    1. Перемешать список индексов случайным образом order: = shuffle($[0, 1, \ldots, N-2, N-1]$)
    2. Для каждого idx из order:
        1. $\vec{x} := X[idx]$ // Берём случайный вектор из X
        2. $i := \arg\min_j \left\| \vec{x} - \overrightarrow{w_j} \right\|$ // Находим номер ближайшего к нему нейрона
        3. $\forall j : h_{ij} := e^{D_{ij}/\sigma^2}$ // Находим коэффициент близости нейронов по сетке. Раметьте, что h_{ii} всегда 1
        4. $\forall j : \overrightarrow{w_j} := \overrightarrow{w_j} + \eta h_{ij}(\vec{x} - \overrightarrow{w_j})$
    3. $\eta := \gamma\eta$
    4. $\sigma := \alpha\sigma$

SOM - X (d×N) — матрица со входными данными[2], где N — количество элементов в наборе данных, а d — размерность данных. W— матрица весов (d×M), где M — количество нейронов в карте. η — скорость обучения, σ — коэффициент кооперации (см. ниже), E — количество эпох.

(M×M) — матрица расстояний между нейронами в слое. Последняя матрица — это не то же самое, что $\|w_i \rightarrow - w_j \rightarrow\|$. *Для нейронов есть два расстояния*: расстояние в слое (D) и разница между значениями.
Также стоит обратить внимание, что хоть нейроны часто для удобства рисуются так, будто бы они «нанизаны» на квадратную сетку, это совсем не означает, что и хранятся они как тензор M×M×d. Сетку олицетворяет D.

γ и α — показатели затухания скорости обучения и затухания кооперации соответственно.

Каждую эпоху обучения мы перебираем элементы входного набора данных. Для каждого элемента мы находим ближайший к этому x нейрон, затем обновляем его веса и веса всех его соседей по слою в зависимости от

расстояния до x и от коэффициента кооперации σ. Чем больше σ, тем больше нейронов эффективно обновляют веса на каждом шаге.

1.1.2. Growing Neural Gas

«Расширяющийся нейронный газ — это алгоритм, позволяющий осуществлять адаптивную кластеризацию входных данных, то есть не только разделить пространство на кластеры, но и определить необходимое их количество исходя из особенностей самих данных. Расширяющийся нейронный газ не требует априорной информации о данных, таких как оценка количества кластеров или форма кластеров.» В данной модели не фиксировано соседство узлов, а динамически меняется по мере улучшения кластеризации. Переменными являются не только отношения соседства, но и число нейронов-кластеров.

Алгоритм 1:
Вход: $X, \alpha, \beta, \gamma, \eta_{winner}, \eta_{neighbour}, \lambda, a_{max}, E, w_1, w_2$

В W изначально 2 нейрона w_1 и w_2, они связаны дугой с возрастом 0
$error$ инициализируется нулями

До сходимости выполнять:

1. Сэмплировать входной вектор $\vec{x}$
2. Найти два нейрона, ближайших к $\vec{x}$. Пусть s — номер ближайшего, t — следующего за ним.
3. Накопить в массиве ошибок расстояние от ноды до сгенерированного образца данных:
$$error_s \leftarrow error_s + \left\|\overrightarrow{w_s} - \vec{x}\right\|$$
4. Обновить местоположение s и всех нейронов, соединённых с ним рёбрами. Заметьте, что используются разные скорости обучения:
$$\overrightarrow{w_s} \leftarrow \overrightarrow{w_s} + \eta_{winner}(\vec{x} - \overrightarrow{w_s})$$
$$\overrightarrow{w_n} \leftarrow \overrightarrow{w_n} + \eta_{neighbour}(\vec{x} - \overrightarrow{w_n}), \forall w_n$$
5. Увеличить возраст всех дуг, исходящих из нейрона s
6. Если s и t уже соединены дугой, то обнулить возраст этой дуги, иначе просто создать дугу с возрастом 0 между ними
7. Удалить все дуги с возрастом больше a_{max}

8. Удалить все ноды, из которых не исходит ни одной дуги

9. Если текущая итерация делится на λ без остатка и количество нод не достигло максимального значения β то

    1. Найти нейрон u с наибольшей накопленной ошибкой

    2. Среди соседей u найти нейрон v с наибольшей ошибкой

    3. Создать новую ноду r между u и v:

$$\overrightarrow{w_r} \leftarrow \frac{\overrightarrow{w_u} + \overrightarrow{w_v}}{2}$$

    4. Создать рёбра между u — r и v — r, удалить ребро u — v

    5. Уменьшить ошибки нейронов u и v, передать новорожденному нейрону часть этих ошибок:

$$error_u \leftarrow \alpha \times error_u$$

$$error_v \leftarrow \alpha \times error_v$$

$$error_r \leftarrow \frac{error_u + error_v}{2}$$

10. Уменьшить весь вектор ошибок:

$$error \leftarrow \gamma \times error$$

В пространство данных специальным образом внедряются нейроны[3], которые в ходе работы алгоритма подстраивают местоположение. После окончания цикла оказывается, что в местах, где плотность данных высока, нейронов тоже много и они связаны друг с другом, а где мала — один-два или вовсе ни одного.

Нейронный газ не использует гипотезу о стационарности данных. Если вместо фиксированного массива данных X, у вас есть функция f(t), сэмплирующая данные из некоторого меняющегося со временем распределения, GNG запросто отработает и на ней. Моменты, относящиеся к случаю меняющегося распределения рассмотрены отдельно.

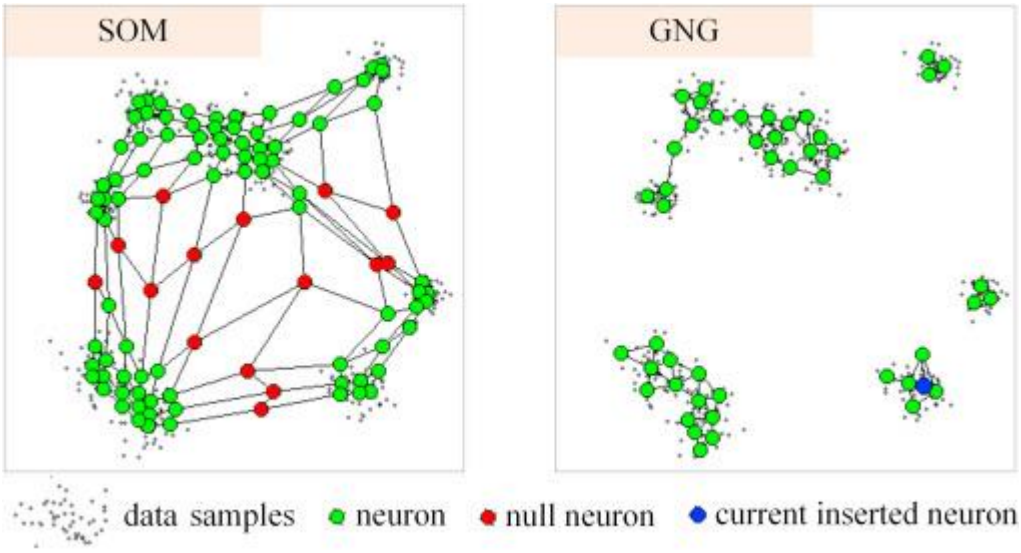

Рис. 2. Разница в работе между SOM и GNG на одном наборе данных.

Пусть $\vec{x}$ — образец данных размерности d, W — матрица с положениями нейронов размера не больше, чем β×d, где β — гиперпараметр, показывающий максимальное количество нейронов (реальный размер W может меняться в ходе работы алгоритма). error — вектор размера не больше, чем β.

Дополнительные гиперпараметры, влияющие на поведение алгоритма: η_{winner}, $\eta_{neighbour}$ — скорость обучения нейрона-победителя и его соседей соответственно, a_{max} — максимальный возраст связи между нейронами, λ — период между итерациями порождения новых нейронов, α — показатель затухания накопленных ошибок при создании новых нейронов, γ — затухание ошибок каждую итерацию, E — количество эпох.

1.1.3. Adaptive Resonance Theory

Разработана Стивеном Гроссбергом и Карпентером в середине 80-х гг. Парадигма использует неконтролируемое обучение, анализирует значимые входные данные, выявляет возможные признаки и классифицирует образы в входном векторе.

Сеть адаптивной резонансной теории состоит из двух взаимосвязанных слоев нейронов, расположенных между входным и выходным слоями. Каждый входной образ низшей слоя резонанса стимулирует ожидаемый образ на высшем слое, который пересылается к низшему слою, чтобы влиять на следующий вход. Это создает "резонанс" между низшим и высшим слоями для облегчения сетевой адаптации образов.

Сеть преимущественно используется в биологическом моделировании, тем не менее существуют некоторые технические применения. Главным ограничением сетевой архитектуры является ее шумовая чувствительность. Даже небольшое количество шума на входном векторе путает обобщающие возможности наученной сети. [10]

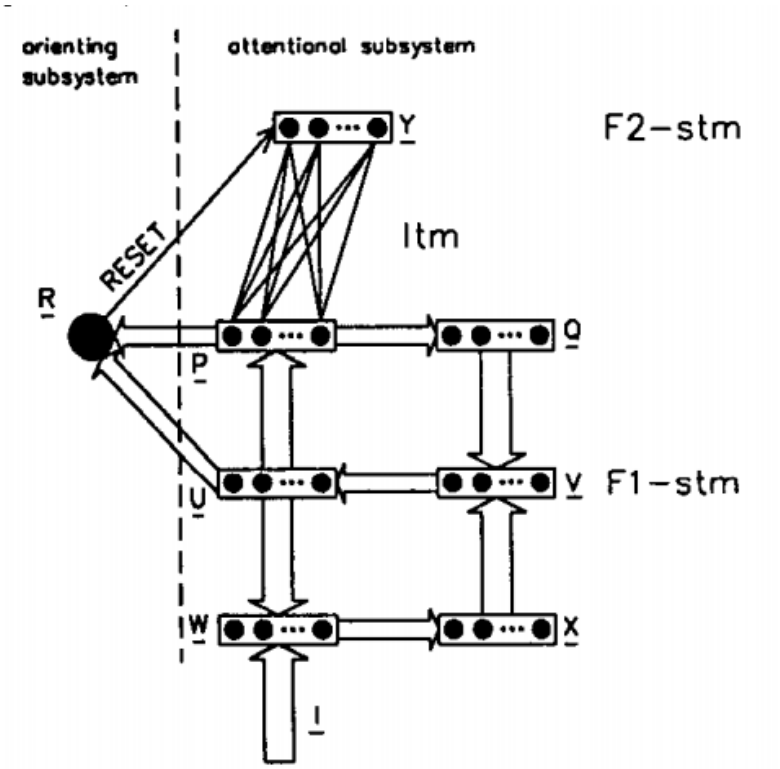

Рис. 3. Архитектура сети ART 2

– АРТ-1 (от англ. ART – AdaptiveResonanceTheory) – для кластеризации, хранения и идентификации образов в форме двоичных сигналов;

– АРТ-2 – для кластеризации, хранения и идентификации образов, представленных как в форме двоичных сигналов, так и в форме аналоговых сигналов, в том числе с использованием обоих типов сигналов в одной структуре.

В отличие от большинства существующих архитектур нейронных сетей АРТ-сети не предусматривают строгого деления жизненного цикла на стадии обучения и практического использования. Они продолжают обучаться на протяжении всего времени их практического использования, включая этап практической эксплуатации.

Алгоритм ее функционирования включает две стадии жизненного цикла: инициализацию и кластеризацию (сравнение) образов.

Стадия **инициализации** состоит из следующих этапов:

1.1. Устанавливается параметр сходства $R_{кр}$, аналогичный по физическому смыслу и области допустимых значений такому же параметру архитектуры АРТ-1.

1.2. Для имеющейся выборки данных выполняется нормализация значений переменных в пределах [0, 1].

1.3. Производится нормирование элементов каждого входного вектора таким образом, чтобы сумма квадратов этих элементов равнялась 1:

$$\hat{x}_i = \frac{\widetilde{x}_i}{\sqrt{\sum_{p=1}^{M} x_p{}^2}}.$$

1.4. Создается первый нейрон (кластер) с весовыми коэффициентами, численно равными нормированным входным значениям первого примера (образа):

$w_{i1} = \hat{x}_i$.

Стадия **кластеризации образа (сравнения)** включает следующие этапы:

2.1. На входы нейронной сети подается очередной образ и определяется количественная мера его сходства с каждым из имеющихся кластеров:

$$R_j = \sum_{i=1}^{M} w_{ij} \hat{x}_i$$.

2.2. Выбирается кластер с номером *J* с максимальным значением меры сходства: $R_{\max} = \max_{j=\overline{1,K}} (R_j)$.

2.3. Если условие $R_{\max} < R_{кр}$ выполняется, считается, что сходство входного образа ни с одним из кластеров не установлено. В этом случае создается новый нейрон (кластер) с весовыми коэффициентами, равными элементам соответствующего нормированного вектора $\hat{x}$.

2.4. Если условие $R_{\max} >= R_{кр}$ не выполняется, считается, что установлено наибольшее сходство входного образа с кластером *J*. Тогда весовые коэффициенты соответствующего нейрона пересчитываются по соотношению:

$$w_{iJ}^{(q+1)} = (1 - v) \cdot w_{iJ}^{(q)} + v \cdot \hat{x}_i$$.

Далее алгоритм продолжает работу на стадии кластеризации с п. 2.1.[9]

2. ОПИСАНИЕ АНАЛОГОВ

В данной главе описаны аналоги с похожей проблематикой, или в общем полезные статьи на тему кластеризации распределённых данных и нейронных сетей. Также общие достоинства и недостатки аналогов относительно того, как задача должна будет решена.

2.1. Список аналогов

2.1.1. Flavius 2010

В книге Flavius L. Gorgônio [Flavius 2010] описывается полностью функционирующий фреймворк partSOM для работы методов кластеризации Self-Organizing Maps и K-means с распределёнными данными. Даются основные понятие термину кластеризации, какие основные шаги надо предпринять для работы с распределёнными данными в совокупности, с проблемами, которые из этого последуют. В этой работе предлагается новая стратегия, путем применения алгоритма кластеризации к каждому распределённому набору данных, относительно вертикально распределенных баз данных, делается это для получения подмножества каждого локального набора данных. В последовательности обработки они отправляются на центральный сайт, который выполняет объединение этих данных. Затем снова применяется алгоритм кластеризации, только уже над всем набором данных. Для достижения новых, возможно лучших результатов рекомендуют применять разные алгоритмы на первом и втором этапе, так для примера, некоторые алгоритмы добиваются лучших результатов при применении к небольшим наборам данных, но дают неточную оценку при работе с большими. Этот подход был применён над известными наборами данных, такими как: Iris Dataset, Wine Dataset, Wisconsin Breast Cancer Dataset и Mushroom Dataset. Где больше, чем в половине случаев дал более точные результаты на 0,5% - 1% и меньшую ошибку. [4]

2.1.2. Махров 2014

В статье Махров С.С [Махров 2014] представляет работу по внедрению Самоорганизующийся карты Кохонена в беспроводной сенсорной сети, с определением критериев для проведения кластеризации и анализом выбора входных данных для данной сети. Описываются пункты для подготовки входной выборки и обучающей выборки соответственно, и какие шаги нужно предпринять чтобы обучить сеть Кохонена. Результаты по работе представлены в виде 2D диаграмм кластеризации. Даются выводы, на основании которых ясно что, СКК (Самоорганизующаяся карта Кохонена) является эффективным методом кластеризации матрицы мощностей. [5]

2.1.3. Bouchachia 2010

В статье Abdelhamid Bouchachia [Bouchachia 2010] описаны основные причины использования распределённых данных в большинстве современных систем, и почему вследствие этого возникают ограничения в виде индивидуальной и корпоративной конфиденциальности. И кластеризация даёт возможность справиться с ограничениями соответствующим образом. Фактически, идея состоит в том, чтобы передать образец данных только после согласования конфиденциальности между центральным сайтом и распределенными сайтами. После сбора с разных сайтов данные должны быть отформатированы, чтобы стать единым набором данных. Таким образом, вопросы, связанные со структурой и точками данных, должны быть приняты на центральном сайте. В статье предложено выполнить три шага следующим образом: 1) построить кластера C_i (называемые локальным кластером)в каждом наборе данных, 2) Затем кластеры K_j (глобальные кластеры) построенные из наборов данных полученных в результате объединения отдельных наборов данных на центральном сайте, 3) после создания кластеров на обоих уровнях, целью становиться определить тип отношений между локальными и глобальными кластерами. Для проверки подхода они использовали набор данных

статистики рака груди, 300 точек данных использовалось для обучения ассоциатора, для проверки эффективности объединения для обоих случаев использовалось 150 точек. Соответственно в случае абстракции получен результат в 97,33%, в случае с ассоциацией 94%. Это означает, что распределенные сайты сообщают центральному сайту только матрицу разбиения, полученную в результате процесса кластеризации, выполняемого локально. Исходные данные будут больше не нужны. Следовательно, конфиденциальность данных сохраняется. [6]

2.1.4. Klusch 2003

В статье Matthias Klusch [Klusch 2003] рассмотрен метод кластеризации на основе выборочных оценок локальной плотности, даётся понятие хранилища данных и объясняется их применение в кластеризации распределённых данных, описываются достоинства и недостатки. В KDEC(Kernel Density Estimate Clustering) каждый источник данных передает оценку вероятности плотности ее локальных данных на вспомогательный сайт и затем выполняет алгоритм кластеризации на основе плотности, которая основанна на общей оценке плотности, которая построена помощником из выборок локальных плотностей. Подход использует статистическую оценку плотности и информацию теоретической выборки для минимизации связи между сайтами. Кроме того, сохраняется конфиденциальность данных в значительной степени, так как никогда не передаётся значения данных. Такой подход не требует значительных затрат на CPU и I/O, затраты аналогичного централизованного подхода и его расходы на связь могут быть ниже. [7]

2.1.5. Du 2010

В статье K.-L. Du [Du 2010] описаны подходы кластеризации с использованием нейронных сетей. Перечисляются основные особенности и

возможности таких подходов как: Самоорганизующиеся карты Кохонена (SOM), квантовании векторного обучения (LVQ), метод нечёткой кластеризации C-средних (fuzzy C-means), Нейронный газ (Neural gas), Сети адаптивной резонансной теории (ART networks). Также даются описания и модификации некоторых проблем при инициализации подходов: Проблема мёртвых элементов (dead-unit problem), надёжная кластеризация – для устранения влияния выбросов (Robust clustering), Конструктивные методы кластеризации, Обоснованность кластера. С предоставлением примера работы SOM над набором данных в 1000 экземпляров. [8]

3. СРЕДСТВА РАЗРАБОТКИ

В данной главе описаны средства, которые будут использоваться для реализации поставленной задачи

3.1. Выбор языка программирования: Python

Python - интерпретируемый, объектно-ориентированный, высокоуровневый язык программирования с динамической семантикой. Его высокоуровневые встроенные структуры данных в сочетании с динамической типизацией и динамической привязкой делают его очень привлекательным для быстрой разработки приложений, а также для использования в качестве языка сценариев или связующего существующих компонентов. Простой и понятный синтаксис Python подчеркивает читаемость и, следовательно, снижает затраты на обслуживание программы. Python поддерживает модули и пакеты, что способствует модульности программы и повторному использованию кода. Интерпретатор Python и обширная стандартная библиотека доступны в исходной или двоичной форме бесплатно для всех основных платформ и могут свободно распространяться.

Поскольку этап компиляции отсутствует, цикл редактирования тестирования-отладки невероятно быстр. Отладка программ Python проста: ошибка или плохой ввод никогда не вызовут ошибку сегментации. Вместо этого, когда интерпретатор обнаруживает ошибку, он вызывает исключение. Когда программа не обнаруживает исключения, интерпретатор печатает трассировку стека. Отладчик исходного уровня позволяет проверять локальные и глобальные переменные, оценивать произвольные выражения, устанавливать точки останова, каждый раз переходить через код строки и т. д. Отладчик написан на самом Python, что свидетельствует о возможности интроспекции Python. С другой стороны, часто самый быстрый способ отладки программы состоит в том, чтобы добавить в источник несколько операторов печати: быстрый цикл редактирования-тестирования-отладки делает этот простой подход очень эффективным.

3.2. Библиотеки для машинного обучения

Существует несколько библиотек машинного обучения которые используются для обучения искусственного интеллекта, нейронных сетей.

3.2.1. TensorFlow

Когда вы попадаете в мир AI, один из первых фреймворков, о которых вы услышите, — это Google TensorFlow.

TensorFlow — это программное обеспечение с открытым исходным кодом для проведения численных расчетов с использованием графиков потоков данных. Эта инфраструктура известна своей архитектурой, которая позволяет выполнять вычисления на любом процессоре или графическом процессоре, будь то настольный компьютер, сервер или даже мобильное устройство. Эта структура доступна на языке программирования Python. Как и аналогичные платформы, она предназначена для оптимизации процесса разработки и выполнения приложений для таких пользователей, как дата аналитики, статисты и прогнозисты.

Достоинства:

- использует лёгкий в изучении язык (Python);
- использует вычислительные графы абстракции;
- доступность TensorBoard для визуализации.

Недостатки:

- медленный, так как Python не самый быстрый язык;
- отсутствие многих предварительно обученных моделей;
- не полностью в свободном доступе.

3.2.2. Theano

Сильный конкурент TensorFlow, **Theano** - мощная библиотека Python, которая позволяет выполнять числовые операции с использованием многомерных массивов с высоким уровнем эффективности.

Явное использование библиотекой графического процессора для выполнения ресурсоемких вычислений вместо процессора приводит к высокой эффективности ее работы.

По этой причине Theano использовался для запуска крупномасштабных вычислительных операций в течение приблизительно десятилетия.

Достоинства:

- хорошо оптимизирован для ЦП и ГП;
- эффективен для вычислительных задач;

Недостатки:

- сам по себе является низко уровневым сравнению с другими библиотеками;
- необходимо использовать в совокупности с другими библиотеками для получения высокого уровня абстракции.

3.2.3. Keras

Keras — это библиотека нейронных сетей с открытым исходным кодом, написанная на Python.

В отличие от TensorFlow, CNTK и Theano, Keras не предназначен для сквозной системы машинного обучения.

Вместо этого он служит интерфейсом и обеспечивает высокий уровень абстракции, что упрощает настройку нейронных сетей независимо от того, на какой платформе он находится.

Достоинства:

- удобный для пользователя в связи с абстракцией;
- легко расширяемый;
- работает без проблем на процессоре и графическом процессоре;
- работает без проблем с совокупности Theano и TensorFlow.

Недостатки:

- не может быть эффективно использован в качестве независимой структуры.

3.2.4. Scikit-learn

Scikit-learn — это очень мощная библиотека Python для машинного обучения, которая в основном используется при построении моделей.

Созданный с использованием других библиотек, таких как numpy, SciPy и matplotlib, он очень эффективен для методов статистического моделирования, таких как классификация, регрессия и кластеризация.

Scikit-learn поставляется с такими функциями, как контролируемые алгоритмы обучения, неконтролируемые алгоритмы обучения и перекрестная проверка.

Достоинства:

- доступность использования многих известных алгоритмов кластеризации;
- эффективен для интеллектуального анализа данных.

Недостатки:

- теряет в производительности, если все основные вычисления производятся на ГП.

3.2.5. Modular toolkit for Data Processing

Модульный инструментарий для обработки данных (MDP) — это среда обработки данных, написанная на Python.[12] С точки зрения пользователя, MDP представляет собой набор контролируемых и неконтролируемых алгоритмов обучения и других блоков обработки данных, которые могут быть объединены в последовательности обработки данных и более сложные архитектуры прямой связи. Вычисления выполняются эффективно с точки зрения скорости и требований к памяти. С точки зрения разработчика, MDP — это модульная структура, которую можно легко расширить. Реализация новых алгоритмов проста и интуитивно понятна. Новые реализованные блоки затем автоматически интегрируются с остальной частью библиотеки. MDP был написан в контексте теоретических исследований в области неврологии, но он был разработан, чтобы быть полезным в любом контексте, где используются обучаемые алгоритмы обработки данных.

Его простота на стороне пользователя, разнообразие легкодоступных алгоритмов и возможность повторного использования реализованных модулей делают его также полезным образовательным инструментом.

ЗАКЛЮЧЕНИЕ

Цели и задачи, поставленные для Научно-Исследовательской работы – проанализировать алгоритмы нейросетевой кластеризации и рассмотреть аналоги статей и научных работ по кластеризации распределённых данных, можно считать выполненными было произведено описание и краткий разбор.

Кроме рассмотренных целей были выделены библиотеки машинного обучения для реализации поставленной задачи: MDP в совокупности с Scikit-Learn, и был выбран язык программирования, на котором алгоритм GNG будет реализовываться: Python.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. D.X. Tian., Y.H. Liu., J.R. Shi., Dynamic clustering algorithm based on adaptive resonance theory. College of Computer Science and Technology Jilin University Changchun China, 2007. – 290c.
2. Нестандартная кластеризация 4: Self-Organizing Maps, тонкости, улучшения, сравнение с t-SNE // Хабрахабр. URL: https://habr.com/post/338868/ (дата обращения: 05.11.2018).
3. Нестандартная кластеризация 5: Growing Neural Gas// Хабрахабр. URL: https://habr.com/post/340360/ (дата обращения: 14.11.2018).
4. Gorgônio F. L., Costa J. A. F. PartSOM: A framework for distributed data clustering using SOM and K-Means //Self-Organizing Maps. – InTech, 2010.
5. Махров С. С. Нейросетевая кластеризация узлов беспроводной сенсорной сети // T-Comm. 2014. №6. URL: https://cyberleninka.ru/article/n/neyrosetevaya-klasterizatsiya-uzlov-besprovodnoy-sensornoy-seti (дата обращения: 16.11.2018).
6. Bouchachia A. Distributed Data Clustering //CAiSE Short Paper Proceedings. – 2003.
7. Klusch M., Lodi S., Moro G. Distributed clustering based on sampling local density estimates //IJCAI. – 2003. – С. 485-490.
8. Du K. L. Clustering: A neural network approach //Neural networks. – 2010. – Т. 23. – №. 1. – С. 89-107.
9. Нейронные сети адаптивного резонанса// Neuronus. URL: https://neuronus.com/theory/nn/956-nejronnye-seti-adaptivnogo-rezonansa.html (дата обращения: 03.12.2018).
10. Классификация известных нейросетей по основным категориям применения// Портал магистров. URL: http://masters.donntu.org/2008/fvti/shakhovaya/library/index8.htm (дата обращения: 05.12.2018).

11. 10 Best Frameworks and Libraries for AI// DZone. URL: https://dzone.com/articles/progressive-tools10-best-frameworks-and-libraries (дата обращения: 12.12.2018).
12. Zito T. et al. Modular toolkit for Data Processing (MDP): a Python data processing framework //Frontiers in neuroinformatics. – 2009. – Т. 2. – С. 8.
13. Fišer D., Faigl J., Kulich M. Growing neural gas efficiently //Neurocomputing. – 2013. – Т. 104. – С. 72-82.
14. Анализ данных и процессов: учеб. пособие / А. А. Барсегян, М. С. Куприянов, И. И. Холод, М. Д. Тесс, С. И. Елизаров. — 3-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2009. — 512 с.: