МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

Кафедра информационных систем и технологий

Специальность Информационные системы и технологии

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №14 НА ТЕМУ:

Согласование криптографических ключей на основе технологий искусственных нейронных сетей

Выполнила студентка 3 курса 1 группы

Пригодич Вера Валерьевна

Минск 2023

Цель: изучение основ построения и функционирования искусственных нейронных сетей (ИНС), а также использования ИНС в криптографии; приобретение практических навыков программной реализации алгоритма согласования ключевой информации на основе технологии ИНС.

**Теоретические сведения**

Нейронная сеть – математическая модель, построенная по принципу организации и функционирования биологических нейронных сетей. Искусственные нейронные сети (ИНС) состоят из перцептронов. Искусственный нейрон, называемый также персептроном или перцептроном (англ. perceptron), симулирует работу биологического нейрона.

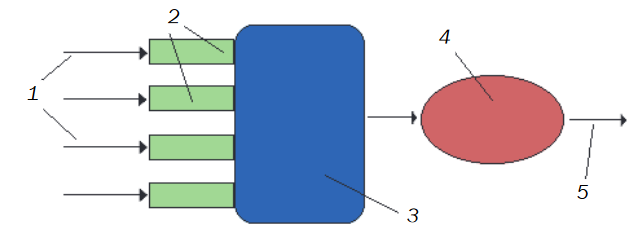


Рисунок 1 ­– Модель искусственного нейрона – персептрона: 1 – входы; 2 – веса; 3 – суммирующий блок; 4 – блок активации; 5 – выход

В нейронной сеті каждый нейрон получает одінаковый набор входных сигналов и каждый из них имеет свій собственный вектор весов. образом, значение выхода нейрона равно скалярному произведению входных значений на векторы весовых коэффициентов (нейрон с линейной функцией активации).

Нейронная сеть должна быть обучена, т.е. должны быть подобраны коэффициенты вектора весов в контексте решаемой задачи.

В криптографии может быть использована модель взаимного обучения ИНС. Это значит, что сети учатся друг у друга, используя для этого полученные результаты, и, стремясь к «общей цели», находят общие элементы в результате своих вычислений.

Общее значение векторов весов обоих персептронов, отличающееся только знаком, может быть использовано в качестве секретного ключа.

Древовидная машина чётности (Tree Parity Machine, TPM)– многослойная искусственная нейронная сеть, состоящая из одного выходного нейрона, *K* скрытых нейронов и *K\*N* входных нейронов.

Архитектуру сети составляют *K* классических персептронов, принадлежащих внутреннему уровню; внешний уровень (обозначен римской цифрой II) объединяет выходы персептронов внутреннего уровня (I) (рис. 2).

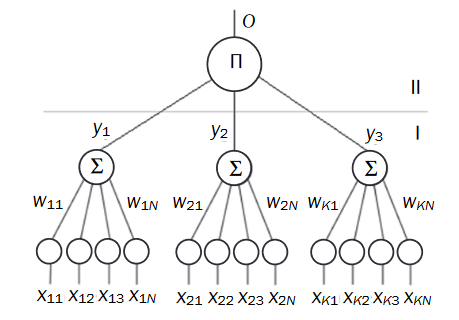


Рисунок 2 – Схема TPM

Секретные элементы, на которых основывается конструкция системы, – это начальные состояния векторов весов.

Процесс взаимного обучения, или синхронизации, начинается с инициализации весовых параметров обеих сетей подачей на вход соответствующих бинарных последовательностей.

Каждый последующий шаг (*t* + 1) начинается с подачи на входы обеих сетей выбранного случайным образом вектора *X*. Затем вычисляется выходная величина *О* для каждой из архитектур. Обе сети «знают» полученные выходные величины друг друга. На основании обеих полученных выходных величин реализован процесс обучения. Активизация векторов весов обеих сетей происходит только тогда, когда обе выходные величины равны друг другу.

Число шагов, за которое векторы весов достигают одинаковых значений, зависит от начальных состояний векторов весов, входных значений и от таких параметров системы, как *K* и *N*. Вся процедура взаимного обучения никак не влияет на два первых параметра, так как выбор начальных состояний векторов весов и значение входных величин генерируются случайным способом. Однако параметры *K* и *N* пропорционально влияют на увеличение времени синхронизации.

**Практическое задание**

1. Разработать приложение, реализующее модель ТСР, – эмулятор процесса синхронизации весовых коэффициентов двух ИНС. Рекомендуется выполнить следующие шаги:

1.1. Реализовать класс, описывающий схему персептрона:

• *N* входов;

• *N* весовых коэффициентов;

• выход *у*;

• метод расчета значения выхода *у* или, более точно, выхода *k*-го персептрона *уk*; используем для этого выражения

• реализацию одного из алгоритмов обучения (синхронизации) сетей:

− алгоритм Хэбба; при этом соотношение представим в несколько ином виде (для сети А):

(*wi*)A(*t* + 1) = (*wi*)*A*(*t*) + (*уixi*)*А*Θ((*уi*)*А* (*ОB*))Θ((–*О*)*А*(*ОB*));

обучение другой сети по Хэббу основано на использовании последнего математического выражения при замене А на В;

− алгоритм анти-Хэбба; основано также на использовании, оператор суммирования в обоих случаях следует поменять на оператор вычитания;

− алгоритм «случайного блуждания»;

1.2. Реализовать класс, описывающий архитектуру нейронной сети TPM:

• *K* персептронов;

• выход *О* сети;

• метод расчета *О* на основе соотношения; с учетом того, что основная часть вычислений, входящих в данную формулу, нами уже выполнена, достаточно здесь выполнить лишь операцию умножения, т. е. реализовать первый оператор после первого слева знака равенства;

• метод обучения нейронной сети (вызывает метод обучения конкретных – выходы которых совпадают с выходом нейронной сети – персептронов).

1.3. Реализовать процесс синхронизации двух нейронных сетей, построенных на архитектуре TPM, который в общем виде можно сформулировать так:

1) задаем случайные значения весовых коэффициентов сетей А и В: wij ∈ {–L, –L + 1, …, 0, …, L};

2) выполняем следующие шаги, пока не наступит синхронизация:

2.1) генерируем случайный входной вектор X: хij = {–1, +1};

2.2) вычисляем yk – e для обеих сетей;

2.3) вычисляем ОА и ОВ,

2.4) сравниваем выходы двух TPM:

а) ОА ≠ ОВ – переход к п. 2.1,

б) ОА = ОВ – применяем выбранное правило обучения.

1.4. Не допустить процесса «бесконечной синхронизации».

1.5. Выводить значения весовых коэффициентов нейронных сетей на экран после наступления синхронизации.

2. С использованием разработанного приложения произвести не менее 500 реализаций алгоритма синхронизации сетей А и В, параметры которых соответствуют варианту задания

2.1. В каждом из выполненных опытов подсчитать количество шагов, после реализации которых наступает синхронизация.

2.2. Составить распределение: число синхронизаций – число шагов.

2.3. В каждом из опытов выполнять контроль промежутка времени наступления синхронизации после запуска эмулятора. Подсчитать среднее время процесса синхронизации. Сравнить это показатель, а также данные по п. 2.2 с соответствующими результатами, полученными коллегами.

Для реализации модели TCP на языке программирования python был разработан скрипт, содержащий указанный класс. Данный класс в качестве конструктора принимает значения количества входов персептрона [N], количество персептронов [K], ограничения, налагаемые на весовые коэффициенты [L], определенные по таблице из методического пособия в соответствии с вариантом.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Вариант | Алгоритм обучения | K | N | ±L |
| 8 | Анти-Хэбба | 4 | 7 | 4 |

Данный класс реализует метод для преобразования в конфигурацию персептрона, определенный по варианту как алгоритм Анти-Хэбба.

Ниже приведен листинг для данного класса:

class TPM:

    def \_\_init\_\_(self, k=3, n=4, l=6):

        self.k = k

        self.n = n

        self.l = l

        self.W = np.random.randint(-l, l + 1, [k, n])

    def get\_output(self, X):

        k = self.k

        n = self.n

        W = self.W

        X = X.reshape([k, n])

        sigma = np.sign(np.sum(X \* W, axis=1))

        tau = np.prod(sigma)

        self.X = X

        self.sigma = sigma

        self.tau = tau

        return tau

    def \_\_call\_\_(self, X):

        return self.get\_output(X)

    def update(self, tau2, update\_rule='hebbian'):

        X = self.X

        tau1 = self.tau

        sigma = self.sigma

        W = self.W

        l = self.l

        if (tau1 == tau2):

            if update\_rule == 'hebbian':

                hebbian(W, X, sigma, tau1, tau2, l)

            elif update\_rule == 'anti\_hebbian':

                anti\_hebbian(W, X, sigma, tau1, tau2, l)

            elif update\_rule == 'random\_walk':

                random\_walk(W, X, sigma, tau1, tau2, l)

            else:

                raise Exception("Invalid update rule. Valid update rules are: " +

                    "\'hebbian\', \'anti\_hebbian\' and \'random\_walk\'.")

Листинг 1 – Класс TPM

Алгоритм обучения сети задан в соответствии с приведенными формулами. Для приведенного в задании алгоритма Анти-Хэбба будет иметь следующий вид:

def anti\_hebbian(W, X, sigma, tau1, tau2, l):

    k, n = W.shape

    for (i, j), \_ in np.ndenumerate(W):

        W[i, j] -= X[i, j] \* tau1 \* theta(sigma[i], tau1) \* theta(tau1, tau2)

        W[i, j] = np.clip(W[i, j], -l, l)

Листинг 2 – Алгоритма Анти-Хэбба

Разработанное приложение производит 500 синхронизаций двух TPM-машин с заданными параметрами и применением правила обучения Хебба. Приложение также при каждой синхронизации подсчитывает число шагов, за которое она происходит, и среднее значение шагов для синхронизации на данном шаге. Затем на основе полученных значений строится график зависимости среднего числа шагов, необходимых для полной синхронизации двух машин.

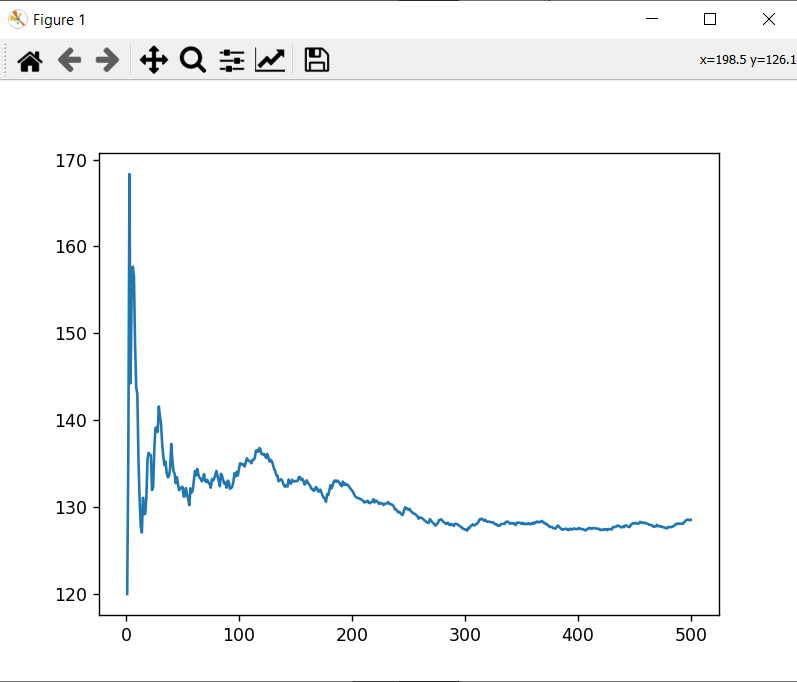


Рисунок 1 – График зависимости среднего числа шагов для полной синхронизации от числа синхронизаций

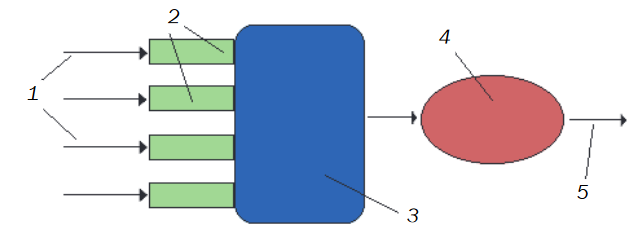


Рисунок 2 – Консольный вывод приложения

**Вывод:** в ходе лабораторной работы было разработано приложение, позволяющее синхронизировать 2 TPM-машины. Среднее число шагов для полной синхронизации для заданных параметров – 128. Среднее время полной синхронизации двух машин составило примерно 0,063 секунды.

**Контрольные вопросы**

1. Изобразить схематично структуру персептрона и пояснить аналогии между его компонентами и частями биологического нейрона.



1 – Входы перцептрона (дендриты);

2 – Веса (синапсы);

3 – Суммирующий блок (ядро);

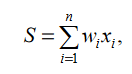
4 – Блок активации (основание аксона);

5 – Выход (аксон).

1. Охарактеризовать (и показать на примерах) области использования ИНС.

ИНС активно используются при анализе и обработке больших массивов информации (задачи классификации и прогнозирования) и при обучении искусственного интеллекта. Некоторые методы на основе ИНС могут быть использованы в криптографии для согласования ключевой информации.

3. Как в простейшем виде записывается формальное представление персептрона?



Где *S* – выходной сигнал;

*wi* – вес входного сигнала;

*xi* – значение входного сигнала.

1. Охарактеризовать (и показать на примерах) области использования ИНС в криптографии.

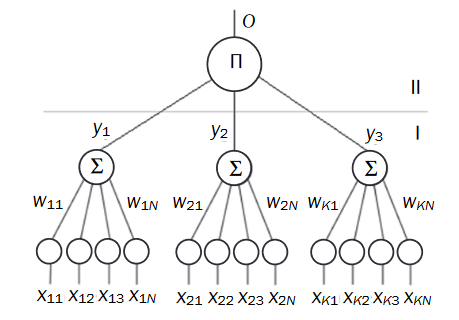
Искусственные нейронные сети (ИНС) имеют применение в криптографии для различных задач. Вот несколько областей использования ИНС в криптографии с примерами:

Криптоанализ: ИНС могут быть использованы для криптоанализа, то есть для анализа и взлома криптографических алгоритмов и систем. ИНС могут обучаться распознаванию шаблонов в зашифрованных сообщениях и поиск уязвимостей в криптографических алгоритмах.

Генерация случайных чисел: ИНС могут использоваться для генерации случайных чисел, которые важны для многих криптографических протоколов и алгоритмов. ИНС обучаются моделировать статистические свойства случайных чисел и генерировать высококачественные случайные последовательности.

Криптография секретного разделения: ИНС могут быть применены в схемах секретного разделения, которые позволяют разделить секретную информацию на несколько частей, такие как секретные ключи, и распределить их между несколькими участниками. ИНС могут использоваться для создания и управления этими схемами.

5. Дать пояснение к структуре и функционалу информационной системы на основе ИНС, предназначенной для согласования ключевой информации.



На входные нейроны подаются сигналы, которые суммируются скрытыми нейронами. Сигналы с выходов скрытых нейронов суммируются выходным нейроном

6. Дать характеристику известным алгоритмам обучения ИНС.

Обратное распространение ошибки (Backpropagation): это один из самых распространенных и эффективных алгоритмов обучения нейронных сетей. Он основывается на принципе минимизации ошибки и позволяет обновлять веса сети на основе разницы между предсказанными и желаемыми значениями.

Генетические алгоритмы (Genetic algorithms): эти алгоритмы основаны на принципе естественного отбора и эволюции. Они создают популяцию нейронных сетей, затем выбирают лучшие особи и создают новые поколения путем комбинирования и мутации весов сети. Этот процесс повторяется до достижения желаемого результата.

Метод опорных векторов (Support Vector Machines): это алгоритм классификации, который также может быть использован для обучения нейронных сетей. Он основан на поиске оптимального гиперплоского разделения между классами данных. SVM может быть использован для обучения нейронной сети путем определения оптимальных весов и смещения.

Стохастический градиентный спуск (Stochastic Gradient Descent): это алгоритм оптимизации, который часто используется для обучения нейронных сетей. Он обновляет веса сети путем итеративного уточнения, используя градиент функции потерь. Стохастический градиентный спуск является быстрым и эффективным методом обучения.

7. Какие алгоритмы используются для обучения ИНС, предназначенных для согласования ключевой информации между двумя сторонами?

Правило обучения Хебба – правило для синхронизации TPM-машин, которое может быть сформулировано следующим образом: если нейрон получает входной сигнал другого нейрона и выходные значения обоих нейронов имеют одинаковый знак, веса входных значений должны быть увеличены.

Правило Анти-Хебба – если нейрон получает входной сигнал другого нейрона и выходные значения обоих нейронов имеют одинаковый знак, веса входных значений должны быть уменьшены.

Обучение на основе случайного блуждания – строится на запоминании и забывании значений из определённого диапазона с корректировками на каждом шаге в зависимости от степени синхронизации.

8. Могут ли легитимно участвовать в процессе синхронизации более трех сетей? Мотивируйте ответ.

Да. Если эти сети выступают в качестве посредников передачи информации, можно создать цепочку из TPM-машин, которые на каждом шаге будут синхронизироваться между собой.

9. Дать характеристику криптостойкости системы на основе двух взаимодействующих ИНС.

Так как при синхронизации используются случайные вектора входных значений, системы обладает высокой криптостойкостью, однако она должна быть достаточно быстрой, чтобы при потенциальная атакующая нейронная сеть не успела синхронизироваться с одной из машин.

10. Какие виды атак на нейрокриптографические системы вам известны? В чем заключается их сущность?

Обучение собственной TPM – атакующий имеет такую же TPM, которые есть в атакуемой системе, и пытается синхронизировать её с системными машинами. Для подобных атак нужны большие вычислительные мощности, так как атакующая TPM обучается медленнее, чем синхронизируются системные машины.