ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра інженерії програмного забезпечення

Пояснювальна записка
до магістерської роботи
на ступінь вищої освіти магістр

на тему: «Вдосконалення системи руху для штучного інтелекту з використанням нейронних мереж»

Виконав: студент 6 курсу, групи ПДМ–61
спеціальності
121 Інженерія програмного забезпечення
(шифр і назва спеціальності/спеціалізації)

Боровик Р.П.
(прізвище та ініціали)

Керівник Дібрівний О.А.
(прізвище та ініціали)

Рецензент _____
(прізвище та ініціали)

Київ –2022

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення
Ступінь вищої освіти -«Магістр»
Спеціальність підготовки – 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного забезпечення
_____Негоденко О.В.
"____" _____2022 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТА

Боровику Роману Павловичу
(прізвище, ім'я, по батькові)

1. Тема роботи: «Вдосконалення системи руху для штучного інтелек з використанням нейронних мереж»

Керівник роботи: Дібрівний О.А. к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом вищого навчального закладу від «13» жовтня 2020 року №230.
2. Строк подання студентом роботи_____
3. Вхідні дані до роботи
Методи навчань нейронних мереж;
Науково-технічна література з питань, пов'язаних з програмним забезпеченням щодо штучного інтелекту за допомогою нейронних мереж;

4. Зміст розрахунково-пояснювальної записки(перелік питань, які потрібно розробити).
4.1 Вимоги та оцінка якості системи.
4.2 Опис проектування системи.
4.3 Опис використаних технологій.

5. Перелік демонстраційного матеріалу (назва основних слайдів)

1. Актуальність проблеми
2. Існуюче програмне забезпечення та методи розпізнавання
3. Принцип роботи інформаційної системи
4. Розпізнавання та групування даних конкретних документів
5. Архітектура бази даних
6. Логічна діаграма компонентів архітектури програмного забезпечення

6. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської роботи	Строк виконання етапів роботи	Примітка
1	Підбір науково-технічної літератури		Виконано
2	Вимоги до системи		Виконано
3	Створення та навчання моделі для вилучення полів		Виконано
4	Створення та навчання моделі для вилучення таблиць		Виконано
5	Концепція та архітектура програмного забезпечення		Виконано
6	Вступ, висновки, реферат		Виконано
7	Розробка обов‘язкових демонстраційних матеріалів		Виконано
8	Попередній захист роботи		
9	Здача роботи		

Студент __ _____
(підпис) (прізвище та ініціали)

Керівник роботи _____ _____
(підпис) (прізвище та ініціали

РЕФЕРАТ

Випускна кваліфікаційна робота 75 с., 24 рис.,., 11 джерел. Ключові слова: штучні нейронні мережі, глибокі нейронні мережі, навчання з учителем, глибоке навчання, рекуррентна нейронна мережа, LSTM, згорткова нейронна мережа, аналіз тональності тексту, мішок слів, word2vec.

Мета роботи: Вдосконалення системи руху для штучного інтелекту з використанням власноруч створених нейронних мереж для забезпечення високої ефективності роботи ігрових об’єктів

Об’єкт дослідження: процес розробки штучного інтелекту з використанням нейронних мереж створений для керування системою руху

Предмет дослідження: штучний інтелект, що розроблений з використанням нейронних мереж для ігор в жанрі аркади, розробка оптимальної топології для навчання нейронної мережі за найменшу кількість ітерацій.

Об'єктом для досліду є методи на основі нейронних мереж для розробки штучного інтелекту для керування автомобілями. Для виконання поставленої в роботі мети необхідно вирішити такі завдання: Вивчити теоретичний матеріал про навчання глибинних нейронних мереж і їх особливості які стосуються обробки природної мови; Вивчити доступну інформацію за доступною документацією про можливість інтеграції нейронних мереж в Unity. Створити алгоритми для розробки моделей згорткової і рекуррентної нейронних мереж; Розробити реалізацію лінійних і нелінійних методів класифікації; Порівняти точність і інші показники якості реалізованих нейромережевих моделей з класичними методами. Для візуалізації навчання використовується Untiy. В роботі показано перевагу класифікаторів на основі глибоких нейронних мереж над класичними методами ласифікації. Найвищу передбачувану точність для даного виду завдання має модель рекуррентної нейронної мережі.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

Deep learning - глибоке навчання;

Supervised learning - навчання з учителем;

SVM (support vector machine) - метод опорних векторів; Pooling - операція об'єднання в свёрточних нейронних мережах; Softmax - функція м'якого максимуму;

Dropout - метод регуляризації для запобігання перенавчання мережі;

Batches - групи прикладів, які використовуються для навчання мережі;

CBOW (continuous bag of words) - безперервний мішок зі словами, один з алгоритмів навчання Word2Vec;

CNN (convolutional neural network) - свёрточная нейронна мережа; RNN (recurrent neural network) - рекуррентная нейронна мережа;

LSTM (long short term memory) - довга короткострокова пам'ять,
різновид архітектури рекурентних нейронних мереж.

ВСТУП

Теперішній світ не можна уявити без читання і обробки інформації. Масиви інформації, яку отримують люди, росте по експоненті. Дана інформація має можливість бути обробленою різними інформаційними системами. На даний час найпростіший для інженера-програміста спосіб обробки інформації, це нейронна мережа. Нейронними мережами може оброблюватися будь-якого виду інформація, від графічної до величезних масивів даних. У даній роботі розглядається використання нейронної мережі для розробки штучного інтелекту

Щодня в міжнародній мережі інтернет з'являється величезна кількість різного виду контенту: тисячі видів ігор розроблюються та публікуються. І кожен в них висловлює свою точку зору, відгуки і оцінки. Завдяки цьому виникає питання спрощення розробки штуного інтелекту. Для її вирішення використовуються методи розробки за допомогою нейронних мереж. Рішення даної проблеми досить відома задача.

Розробка штучного інтелекту за допомогою навчаємої нейронної мережі дозволяє, не звертати увагу на складність задуманого маршруту для об’єкту, адже нейронна мережа сама навчається і через декілька поколінь може набагато краще вирішувати поставлені завдання.

В ході роботи треба було побудувати бінарний класифікатор, який визначає яким, виявився трек, з цією метою були обрані два види характеристики позитивна, негативна. Для досягнення потрібного результату було використано різні методи і векторні моделі уявлень. В результаті навчання були отримані види нейронних мереж, що дозволяють з достатньою достовірністю визначати найуспішніші нейрони, а також проведено порівняння ефективності використання реалізованих методів. Кінцевим результатом стала перевірка системи на тестовій вибірці найточнішим методом.

РОЗДІЛ 1. ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Визначення нейронної мережі

1.1.1 Штучні нейронні мережі та їх складові

Штучні види нейронних мереж були побудовані за принципом різних біологічних нейронних мереж, які являють собою мережі нервових клітин, які виконують певні фізіологічні функції. Складовою частиною нейронних мереж є нейрони (представлені на рис.1.1).

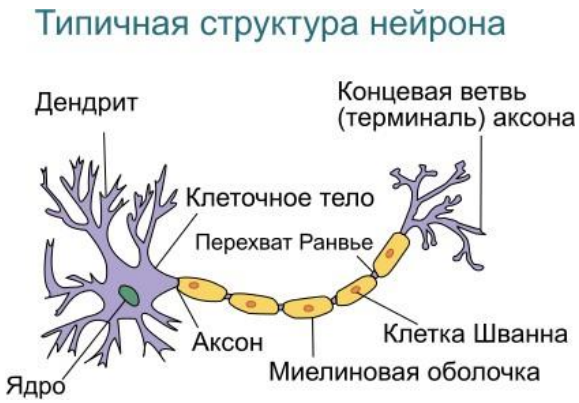


Рисунок 1.1 - Типова структура нейрона

Нейрон має декілька функцій: Приймальна функція: синапси одержують інформацію; Інтеграційна функція: на виході нейрона сигнал, який містить деяку інформацію про всі синапси підсумованих в нейроні сигналах; Провідникова функція: через аксон переміщається інформація Передаваюча функція: імпульс, що досяг закінчення аксона, змушує іншу частину мережі таку як медіатор передавати збудження наступного нейрона.

Синапсами називається зв'язка, за якою вихідні сигнали одних деяких нейронів надходять на входи деяких інших нейронів. Кожна зв'язка охарактеризовується своєю вагою. Зв'язки що мають позитивну вагу мають назву збудливі, а з негативним – гальмуючі. Вихід нейрона називається аксоном. В штучної нейронної мережі така частина як штучний нейрон - це деяка нелінійна

функція, що в якості аргументу має лінійну комбінацію всіх вхідних сигналів. Така функція носить назву активаційна. Далі результат активаційної функції відправляється на вихід нейрона. Об'єднуючи дані нейрони з іншими, отримується штучна нейронна мережа.

1.1.2 Активаційна функція

Функція активації що є у нейрона охарактеризовує залежність сигналу на виході нейрона від суми сигналів на його входах. Зазвичай дана функція являється монотонно зростаючою і лежить в області значень [-1,1] (гіперболічний тангенс) і [0,1] (сигмоїда). Для певних алгоритмів навчання потрібно для того, щоб активаційна функція була безперервно диференційованою на всій числовій осі. Штучний нейрон характеризується своєю активаційною функцією (наприклад, назва "сигмоїдальний нейрон").

Основними активаційними функціями є:

1.Порогова активаційна функція (функція Хевісайда). Не може бути використана для алгоритму зворотного поширення помилки;

$$f(x) = \begin{cases} 1, & x \geq -\omega 0 x 0 \\ 0, & else \end{cases} \tag{1.1}$$

2.сигмоїдальна активаційна функція;

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{1.2}$$

3.Гіперболічний тангенс;

$$\tanh(Ax) = \frac{e^{Ax} - e^{-Ax}}{e^{Ax} + e^{-Ax}} \tag{1.3}$$

1.1.3 Перцептрон

Перцептрон - різнови штучного нейрона, що був розроблений Френком Розенблатом в 1950-х і 1960-х роках. У сучасних роботах найбільш часто

використовують іншу модель штучного нейрона, таку як сигмоїдальний нейрон. Щоб зрозуміти, як працює сигмоїдальний нейрон, необхідно розглянути та зрозуміти структуру і принцип функціонування перцептрону.

Перцептрон приймає на вхід значення 1, 2 ... і видає бінарний результат (див. Рис.3.2).

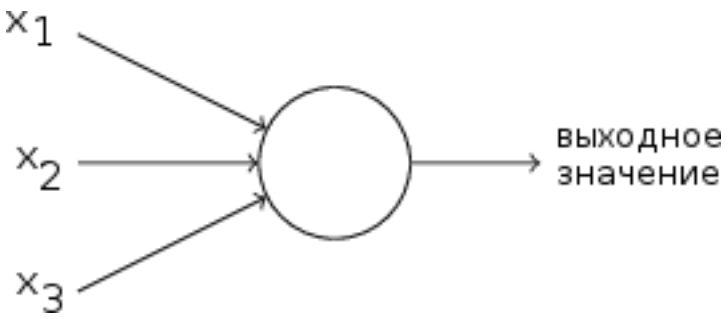


Рисунок 1.2-Схема перцептрона

Розенблатт запропонував задіяти та використовувати ваги – це числа, що виражають значимість вкладу кожного входу в кінцевий результат. Зважена сума (або ваги) порівнюється з кінцевим граничним значенням (threshold), і вже за даними результатами визначається, чи буде виданий 0 або 1. Межа також являється параметром нейрона.

$$\begin{cases} 0, \ IF \ \sum_j \omega 0 x 0 \ \leq threshold \\ 1, \ IF \ \sum_j \omega 0 x 0 \ > threshold \end{cases} \quad (1.4)$$

Перцептрони класифікуються як штучні нейронні мережі що мають один прихований шар; пороговою активаційною функцією; з прямим розповсюдженням сигналу. Навчання такої штучної мережі як перцептрон полягає в зміні матриці ваг в процесі навчання. Існують всього лише 4 історично сформованих видів перцептронів: Перцептрон з одним прихованим шаром; Одношаровий перцептрон: елементи що є вхідними безпосередньо з'єднані з вихідними елементами за допомогою системи ваг. Являється найпростішим видом мереж прямого поширення (feedforward network); Багатошаровий перцептрон (по Розенблатту): присутні додаткові приховані шари; Багатошаровий перцептрон (по Румельхарту):

присутні додаткові приховані шари, а навчання проводиться за таким методом як зворотнє поширення помилки (backpropagation algorithm). Якби невелика зміна ваг (або зсуву) викликало собою невелику зміну на виході мережі, то бажану поведінку нейронної мережі можливо було б отримати за допомогою простих модифікацій зсувів і ваг в процесі навчання. Однак навчання не є таким простим щоб його здійснити, якщо нейронна мережа переважно складається з перцептронів. Невелика зміна ваг або зсуву одного з перцептронів мережі може кардинально змінити вихідне значення перцептрону, наприклад, з 0 на 1. Тому найменш незначна зміна значень лише одного з елементів мережі може створити значні труднощі для розуміння у зміни поведінки мережі. Оскільки завдання навчання нейронної мережі є аналогією до завдання пошуку мінімуму функції помилки в просторі станів навчання, то для її вирішення можуть бути застосовані стандартні методології теорії оптимізації.

1.1.4 Сигмоїдальний нейрон

Сигмоїдальні нейрони найбільш схожі на перцептрони, однак невеликі зміни в їх вагах і зсувах не настільки кардинально змінюють вихід нейрона. Цей факт дозволяє мережі що складається із сигмоїдальних нейронів навчатися. На вхід сигмоїдальна нейрона подаються будь-які значення між 0 і 1. На виході також видається значення між 0 і 1, так як в якості

активаційної функції використовується сигмоїда, що є нелінійної (див. рис. 1.3).

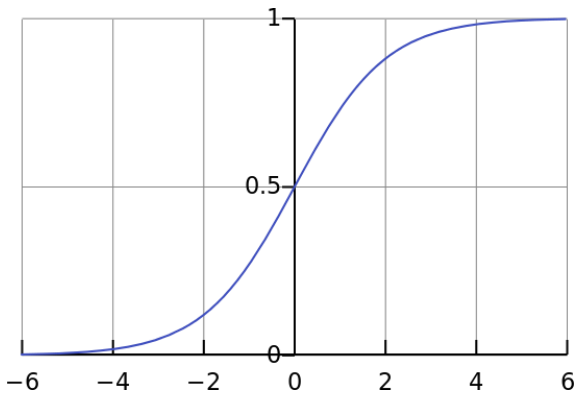


Рисунок 1.3 - Графік логістичної кривої сигмоїди

$$f(x) = \frac{1}{1 + e^{-\beta x}} \tag{1.5}$$

Чим більше β (параметр нахилу сигмоїдальної функції активації), тим більша крутизна її графіка. При $\beta \to \infty$ сигмоїда прагне до функції Хевісайда. Важливою властивістю такого виду нейронних мереж як сигмоїдальна функція є її диференційність. Використання неперервної функції активації дозволяє застосовувати при навчанні нейронної мережі градієнтні методи.

Нейрони можливо розподілити на групи що залежать від їх загального положення в мережі:

вхідні нейрони, що приймають вихідний вектор даних; в проміжних нейронах відбувається основний вид обчислювальних операцій - навчання; вихідні нейрони - результат роботи мережі.

1.1.5 Архітектура нейронних мереж

Розглянемо задачу навчання з учителем. Дано велика кількість тренувальних прикладів X що мають мітки (labels) Y. Нейронні мережі визначають нелінійну гіпотезу hW, b (x) з параметрами W і b. Нейронна мережа, що складається з великої кількості простих нейронів так, що вихід одного з нейронів буде являтися входом іншого (див. рис. 1.4).

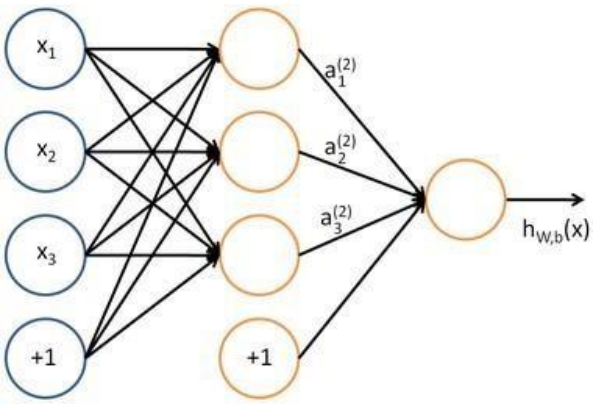


Рисунок 1.4 - Схема найпростішої мережі прямого поширення

Крайній лівий шар має назву вхідний, а шар, що знаходиться з правого боку - вихідний. Шар по центру являється прихованим і має свою назву через те, що його значення не спостерігаються в тренувальних прикладах. Таким чином в даній мережі є всього три елементи входу, три прихованих елемента і один вихідний елемент. Нехай nl - кількість шарів в мережі (в даному випадку 3). Параметри мережі (W, b) = (W (1), b 1), W (2), b (2)). Результат застосування функції активації (виходу) позначається ai Для i-ого елемента. З цього отримуємо таку систему

$$\begin{cases} a_1^{(2)} = f(W_{11}^{(1)}x_1 + W_{12}^{(1)}x_2 + W_{13}^{(1)}x_3 + b_{(1)}^1) \\ a_2^{(2)} = f(W_{21}^{(1)}x_1 + W_{22}^{(1)}x_2 + W_{23}^{(1)}x_3 + b_{(1)}^2) \end{cases} \quad (1.6)$$

Позначивши функцію підсумовування через z, отримаємо векторної формі:

$$\begin{cases} z^{(2)} = W^{(1)}x + b^{(1)} \\ a^{(2)} = f(z^{(2)}) \\ z^{(3)} = W^{(2)}a^{(2)} + b^{(2)} \\ h = f(z^{(3)}) \end{cases} \quad (1.7)$$

Загальна формула матиме такий вигляд:

$$\begin{cases} z^{(l)} = W^{(l)}a^{(l)} + b^{(l)} \\ a^{(l)} = f(z^{(l+1)}) \end{cases} \quad (1.8)$$

Мережа прямого поширення таку назву носять такі нейронні мережі, які мають для використання вихід одного шару в якості вхідних даних для наступного шару.

Навчання нейронних мереж Загальні поняття в навчанні нейронних мереж Епоха - прямий і зворотний прохід по всім тренувальним прикладам. Розмір серії (batch) - кількість тренувальних прикладів що використовується одного повтору прямого та зворотного проходів. Кількість повторів - кількість проходів: кожна ітерація використовує приклади (batch). Один прохід = прямий прохід + зворотний прохід. Тобто маючи тисячу прикладів, batch = 500, нам знадобиться дві ітерації, щоб закінчити одну епоху. З точки зору математики, навчання нейронних мереж –

це багато параметричне завдання нелінійної оптимізації.

Алгоритм зворотного поширення помилок Алгоритм зворотного поширення помилки являє собою визначення стратегії підбору ваг багатошарової мережі із використанням таких методів як градієнтна оптимізація. Оскільки цільова функція, зазвичай має визначення квадратичної різниці суми між фактичними і вихідними значеннями, що очікуються, є безперервною, градієнтні методи оптимізації є найбільш ефективними при навчанні мережі. при навчанні багатошарової нейронної мережі необхідно обчислювати вектор градієнта щодо параметрів всіх шарів мережі. Нехай є кінцевий набір тренувальних даних (m прикладів). Для навчання нейронної мережі застосовується такий метод як пакетний градієнтний спуск (batch gradient descent). Квадратична помилка цільової функції (squared-error cost function) для одного прикладу буде обчислена за формулою:

$$J(W,b,x,y) = \frac{1}{2}\left\|h_{W,b}(x) - y\right\|^2 \tag{1.9}$$

Тоді цільова функція для m прикладів буде виглядати так:

$$J(W,b) = \left[\frac{1}{m}\sum_{i=1}^{m} J\left(W,b,\,x^i,\,y^i\right)\right] + \frac{\lambda}{2}\sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_l+1}(W_{ij}^{(l)})^2 =$$

$$\left[\frac{1}{m}\sum_{i=1}^{m}\left(\frac{1}{2}\left\|h_{W,b}(x^i) - y^i\right\|^2\right)\right] + \frac{\lambda}{2}\sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_l+1}(W_{ij}^{(l)})^2 \tag{1.10}$$

Перший член вираження J (W, b) це сума квадратів помилок, -другий член регуляризації (L2 - зменшення ваг - weight decay), дозволяє зменшити значення ваг і запобігти перенавчання. Параметр регуляризації ваг λ використовують для того , щоб перевірити відносну значущість частин даного виразу. У завданнях бінарної класифікації у представленнях 0 або 1 (так як сігмоідної функції видає значення в межах [0; 1], а проте при використанні гіперболічного тангенса лейблами класів були б -1 і 1). Завдання - мінімізувати J (W, b). Для навчання нейронної мережі необхідно ініціалізувати кожен параметр $W_{ij}^{(l)}$ и $b_i^{(l)}$ малими випадковими величинами, що є найбільш близькими до нуля, а потім застосувати алгоритм оптимізації (згадуваний вище градієнтний спуск). Так як J (W, b) не є опуклою функцією, то градієнтний спуск є сприйнятливий до локальних оптимумів. Кожна

повтор градієнтного спуску оновлює параметри за такою формулою:

$$W_{ij}^{(l)} = W_{ij}^{(l)} - \alpha \frac{\partial}{\partial W_{ij}^{(l)}} J(W,b)$$

$$b_i^{(l)} = b_i^{(l)} - \alpha \frac{\partial}{\partial b_i^{(l)}} J(W,b) \quad (1.11)$$

де α - швидкість навчання (learning rate).

$$\frac{\partial}{\partial W_{ij}^{(l)}} J(W,b) = \left[\frac{1}{m}\sum_{i=1}^{m} \frac{\partial}{\partial W_{ij}^{(l)}} J\left(W,b,x^i,y^i\right)\right] + \lambda W_{ij}^{(l)}$$

$$\frac{\partial}{\partial b_i^{(l)}} J(W,b) = \frac{1}{m}\sum_{i=1}^{m} \frac{\partial}{\partial b_i^{(l)}} J\left(W,b,x^i,y^i\right) \quad (1.12)$$

Кроки алгоритму зворотного поширення помилки

1. Здійснюється прямий прохід по мережі, обчислюються активації шарів L2, L3 і так далі до вихідного шару Lnl;
2. Для кожного вихідного елемента i в вихідному шарі nl (the outputlayer) розраховується помилка

$$\delta_i^l = \frac{\partial}{\partial z_i^{(n_l)}} \frac{1}{2} \left\| h_{W,b}(x^i) - y^i \right\|^2 = -\left(y_i - a_i^{n_l}\right) \cdot f'(z_i^{n_l}) \quad (1.13)$$

3. Для l = nl - 1, nl - 2, nl - 3,. . . , 2:

Для кожного елемента в шарі l, розраховується

$$\delta_i^l = \left(\sum_{j=1}^{s_{l+1}} W_{ij}^{(l)} \delta_i^{l+1}\right) f'(z_i^l) \quad (1.14)$$

4. Йде обчислення приватних похідних

$$\frac{\partial}{\partial W_{ij}^{(l)}} J(W,b,x,y) = a_j^l \delta_i^{l+1}$$

$$\frac{\partial}{\partial b_i^{(l)}} J(W,b,x,y) = \delta_i^{l+1} \quad (1.15)$$

У матричної форми алгоритм буде записаний наступним чином (• позначено похідну Адамара - покомпонентне похідна двох матриць):

Здійснюється прямий прохід по мережі, обчислюються активація шарів L2, L3 і так далі аж до вихідного шару Lnl

1. Матриця помилок для вихідного шару nl

$$\delta^{n_l} = -(y_i - a^{n_i}) \cdot f'(z^{n_i}) \tag{1.16}$$

2. Для шару l = nl - 1, nl - 2, nl - 3,. . . , 2

$$\delta^{l} = ((W^{l})^{T}\delta^{l+1}) \cdot f'(z^{l}) \tag{1.17}$$

3. Обчислення приватних похідних .

$$. \nabla_{w^l} J(W,b,x,y) = \delta_i^{l+1}(a^l)^T$$
$$. \nabla_{b^l} J(W,b,x,y) = \delta^{l+1} \tag{1.18}$$

Недоліки градієнтного спуску

Основні труднощі навчання нейронних виходу з локальних мінімумів. Недоліками навчанні мережі є:

1. Параліч мережі полягає в самих методах градієнтного спуску при тому, що значення ваг мережі за результатами корекції можуть бути дуже великими величинами. Так як помилка, що відправляється назад в процесі навчання, є пропорційною похідною яка стискає дані функції, то процес навчання може практично припинитися. Запобігти цьому можна, зменшуючи крок η, однак процес навчання нейронної мережі буде відбуватися довше.
2. Розмір кроку

   Якщо значення кроку є не змінним, і воно є досить невелике, то метод буде сходитися дуже повільно. Якщо ж крок занадто великий, то може виникнути параліч мережі. Необхідно змінювати значення кроку: збільшувати до того моменту, доки не припиниться покращення оцінки в напрямок антиградієнту і зменшувати, якщо оцінка не поліпшується.
3. Порівняння стохастичного і пакетного градієнтних спусків

   Якщо для пакетного градієнтного спуску функція брам обчислюється по

всім зразкам разом узятим по закінченню даної епохи, а далі змінюються вагові коефіцієнти нейронів, то для стохастичного методу вагові коефіцієнти змінюються після обчислення виходу мережі на одному з навчальних прикладів. Основним недоліком пакетного градієнтного спуску є його "застрягання" в локальних мінімумах. Незважаючи на те що стохастичний метод працює на порядок з меншою швидкістю ніж пакетний, він здатний виходити з локальних мінімумів, що призводить до кращих результатів навчання мережі (стохастичний метод використовує недовичисленний градієнт). Моніторинг стану мережі Функція перехресної ентропії в якості цільової функції Функція перехресної ентропії використовується в якості функції втрат y_i' - передбачені значення, уі - вірні значення.

$$L(x,y) = -\frac{1}{n}\sum_{i=1}^{n} y^i \log a(x^i) + (1-y^i)\log 1 - a(x^i) \quad (1.19)$$

Техніки регуляризації

1. L1-регуляризація: відбувається зміна нерегуляризованої цільової функції за допомогою методу додавання суми абсолютних значень ваг

$$J = J_0 + \frac{\lambda}{n}\sum_W |W| \quad (1.20)$$

При використанні L1-регуляризації відбувається прагнення одного або декількох вагових значень до 0.0, тому відповідна функція (feature) більше не потрібна. Цей ефект називається селекцією функцій (feature selection);

2. L2-регуляризація (також відома як weight decay) На відміну від L1- регуляризації, в L2 ваги ваг:зменшуються на величину, пропорційну вагам

$$J = J_0 + \frac{\lambda}{2n}\sum_W W^2 \quad (1.21)$$

Dropout не впливає на значення цільової функції: відбувається зміна структури мережі. Кожен нейрон видаляється з мережі з певною ймовірністю p. За отриманою

прорідженої мережі робиться зворотне поширення помилки, для решти ваг робиться градієнтний крок. Після цього вилучені нейрони відновлюють в мережі. При навчанні нейронної мережі вихід кожного нейрона множиться на (1-p). Так буде отримано маточікування відповіді мережі по її 2N (де N - кількість нейронів в мережі) архітектурам. Навчена таким чином мережа являється результатом усереднення 2N мереж. Окрема нейронна мережа, яка є навчена за допомогою ранньої зупинки, містить у собі дуже велику помилку, однак усереднення декількох нейронних мереж призводить до значного зменшення помилки.

1.3.Штучне розширення даних для навчання.

1.1.6 Гіперпараметри мережі

Темп навчання: для початку необхідно оцінити значення яке є граничним для η, в якому значення цільової функції миттєво починає знижуватися без коливань. Спочатку значення оцінки встановлюється η = 0.01. Якщо значення цільової функцій знижується під час перших епох, то потрібно збільшувати темп навчання, поки буде не знайдено значення коливання цільової функції. Якщо ж при початковому темпі навчання значення цільової функції коливаються, то необхідно його зменшувати. Темп навчання регулює розмір кроку в градієнтному спуску і спостерігає за значеннями цільової функції, визначаючи, чи був розмір кроку градієнтного спуску занадто великим;

Використовувати ранню зупинку (early stopping) для визначення розміру епох навчання: рання зупинка означає, що в кінці кожної епохи потрібно обчислити вірогідність класифікації на даних перевірки (validation set). Коли поліпшення точності припиниться, зупинити процес навчання.

Така зупинка також запобігає перенавчання;

Графік навчання: ідея - зберігати темп навчання незмінним до моменту, коли достовірність даних перевірки не почне погіршуватися. Тоді необхідно зменшити темп навчання (зменшивши, наприклад, в 10раз).

Параметр регуляризації λ: після визначення η, можна почати з λ = 1.0 і потім збільшувати або зменшувати значення (в 10 разів);

Розмір пакетів: якщо розмір пакетів занадто малий, неможливо повністю використовувати переваги хороших матричних бібліотек, оптимізованих для швидкого обладнання. Якщо ж розмір пакетів занадто великий, то ваги мережі будуть оновлювати дуже нечасто. Необхідно вибрати компромісне значення, яке максимізує швидкість навчання.

1.1.6 Глибокі нейронні мережі

Глибокими нейронними мережами називаються такі нейронні мережі, де є декілька прихованих шарів. Оскільки кожен прихований шар несе функцію обчислення нелінійного перетворення попереднього шару, глибока мережа може мати набагато більшу репрезентативну потужність (тобто може бути представлена значно складнішими функціями), ніж малошарова. При навчанні глибокої мережі дуже важливе є використання нелінійної функції активації в кожному з прихованих шарів. Це пов'язано з тим, що велика кількість шарів лінійних функцій самотужки вираховували б отже, не мали б більше ніж один прихований шару. Головною перевагою використовувати тільки лінійну функцію введення і, виразними, ніж застосування тільки глибинних мереж є стисле уявлення достатнього великої кількості функцій. Можна показати, що існують функції, які k-шарова мережа може представляти стисло, а (k-1) – шарова мережа не може цього зробити, якщо тільки вона не містить у собі експоненціально велику кількість елементів в прихованих шарах.

1.1.7 Доступність даних

За допомогою методу, описаного вище, можна покладатися тільки на марковані дані для навчання. Однак помічених даних нейронної мережі дуже часто буває недостатньо, і, отже, для багатьох завдань важко отримати достатню кількість прикладів для відповідності параметрам складної моделі.

Наприклад, з огляду на високий ступінь виразності глибинних мереж, навчання при невеликій кількості даних призведе до перенавчання.

1.1.8 Локальний оптимум

Навчання малошарової мережі (з 1 прихованим шаром) із застосуванням контрольованого навчання зазвичай призводить до зближення параметрів з відповідними значеннями. Але при навчанні глибокої мережі, це працює набагато рідше. Зокрема, навчання нейронної мережі із застосуванням навчання з учителем включає в себе вирішення проблеми з неопуклого оптимізацією (наприклад, мінімізація помилки навчання $\Sigma \|h , ( ) - \|$ 2 Залежно від мережевих параметрів W). У глибокій мережі з'являється велика кількість локальних оптимумів, тому навчання з градієнтним спуском перестає працювати.

1.1.9 Градієнтна дифузія

При використанні методу зворотного поширення помилки для обчислення похідних, градієнти, які поширюються від вихідного шару до більш ранніх шарів мережі, швидко зменшуються по мірі збільшення глибини мережі. В результаті похідна від загальної вартості по відношенню до ваги в більш ранніх шарах дуже мала. Таким чином, при використанні градієнтного спуску ваги ранніх шарів повільно змінюються і більш ранні шари не можуть багато чому навчитися. Цю проблему часто називають "дифузією градієнтів" (diffusion of gradients).

1.1.10 Проблеми навчання глибоких мереж і їх рішення

1.1.10.1 Зникаючий градієнт

Проблема зникаючого градієнта – це складність, що виникає під час навчання штучних нейронних мереж під час використання методів навчання заснованих на

використанні градієнта та зворотному поширенні помилки. У цих методах будь-яка потреба нейронної мережі пропорційно оновлюється градієнту функції помилки залежно від поточної ваги кожному повторі навчання. Стандартні функції активації, такі як гіперболічний тангенс, мають градієнти, що знаходяться в діапазоні (-1, 1), а метод зворотного поширення помилки здійснює їх обчислення методом ланцюгового правила. Після множення цих чисел обчислення градієнтів " фронтальних " шарів в n-шарової мережі, що означає, що градієнт (сигнал помилки) експоненційно зменшується разом з n, а шари, що знаходяться попереду, навчаються з дуже малою позитивною динамікою. При використанні функцій активації, похідні яких можуть набувати досить великих значень, існує ризик зіткнутися з exploding gradient problem. Можливі рішення: Багаторівнева ієрархія: шар мережі попередньо навчається, використовуючи методи навчання без учителя, а потім його значення регулюється за допомогою методу зворотного розповсюдження помилки. Таким чином, кожен шар мережі вивчає коротке уявлення спостережень, яке подається на наступний шар; довга короткострокова пам'ять: Своєрідна архітектура нейронних мереж, що повторюються. Коли значення помилок передаються в напрямку, протилежному до початкового рівня, блок LSTM не забуває помилку з пам'яті. Він постійно доставляється до кожного з клапанів, поки вони не будуть навчені відхиляти ті самі значення. Глибока мережа має більш високу помилку навчання, ніж мережа меншого рівня.

КомандаMicrosoft Research виявила, що поділ глибокої мережі на секції (скажімо, кожна секція має три таблиці зв'язку) і перенесення ресурсів з кожної секції в наступну секцію (з частиною, що залишилася, за винятком використання нового input.fragment).) зникнення збудження. Жодних доповнень або змін до алгоритму навчання не потрібно. ResNets показав менше помилок навчання (і моделювання помилок), ніж їх менша модель, сигмоїдальна функція активації.

Використання сигмоїдальних активаційних функцій може викликати проблеми в навчанні глибинних мереж, тобто витрати на рух на останній фазі будуть близькі до нуля на початку навчання, сповільнюючи цей процес. Інші види планування не

сильно страждають від ліміту.

1.1.10.2 Вибір відповідних ваг

Вибір відповідних ваг і momentum schedule в імпульсному стохастичною градієнтному спуску (momentum-based stochastic gradient descent) Существенно впливають на здатність навчати глибокі мережі.

Згортувальні нейронні мережі

Згортка є операцією, яка застосовується до двох послідовностей f і g і породжує третю послідовність.

$$(f * g)(c) = \sum_a f(a)\, g(c-a), \text{где } a = b + c \quad (1.25)$$

Формула для двовимірної згортки:

$$(f * g)(c_1, c_2) = \sum_a f(a_1, a_2)\, g(c_1 - a_1, c_2 - a_2) \quad (1.26)$$

Розглянемо одновимірний згортувальний шар з входами xn і виходами yn (див. Рис. 1.5). Тоді функція для виходів буде представлена наступним чином:

$$y_n = A(x_n, x_{n+1} \ldots) \quad (1.27)$$

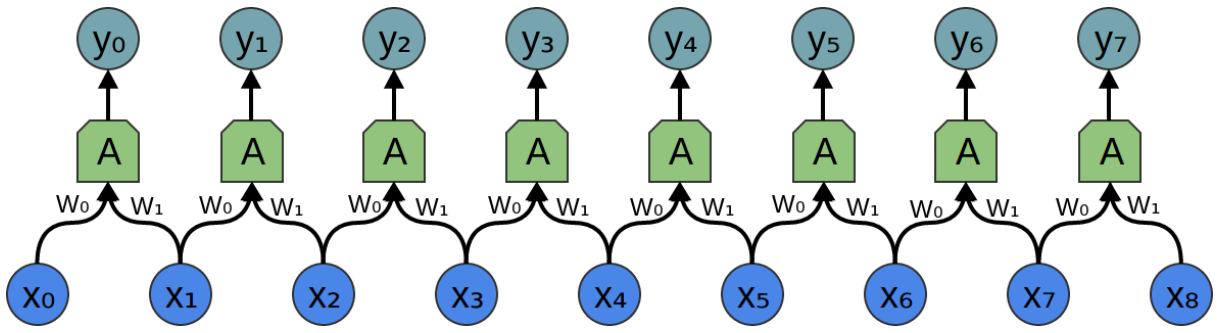


Рисунок 1.5 -Теана одновимірного згортувального шару

У згортувальному шарі знаходиться безлічі копій одного і того ж нейрона, тому багато ваги з'являються в декількох позиціях.

y0 = σ (W0x0 + W1x1 - b) (1.28)

y1 = σ (W0x1 + W1x2 - b) (1.29)

Стандартна матриця ваг з’єднує кожен вхід з кожним нейроном з різними вагами. Матриця для згорткового шару відрізняється, тому різні ваги можна побачити в кількох положеннях, а оскільки нейрони підключені до всіх можливих входів,

$$M = \begin{bmatrix} \omega_0 & \omega_1 & 0 & \dots \\ 0 & \omega_0 & \omega_1 & \dots \\ 0 & 0 & \omega_0 & \dots \end{bmatrix}$$

матриця має багато нульових елементів:

Тобто множення на матрицю вище - те ж саме, що і пакунок з [.. . 0, w1, w0, 0. . .]. Ядро згортки, ковзне по різних частинах зображення, відповідає наявності нейронів в цих частинах. Згортку можна пояснити на прикладі обробки зображень.

Якщо ми припустимо, що зображення є повторюваними, то різні модифікації зображення є не що інше, як конвенція функції зображення з консольним ядром локальної функції. Кожен новий піксель на зображенні представляє зважену кількість пікселів, які ядро пройшло повторно. Двовимірний згортковий шар представлений на рис. 1.6.

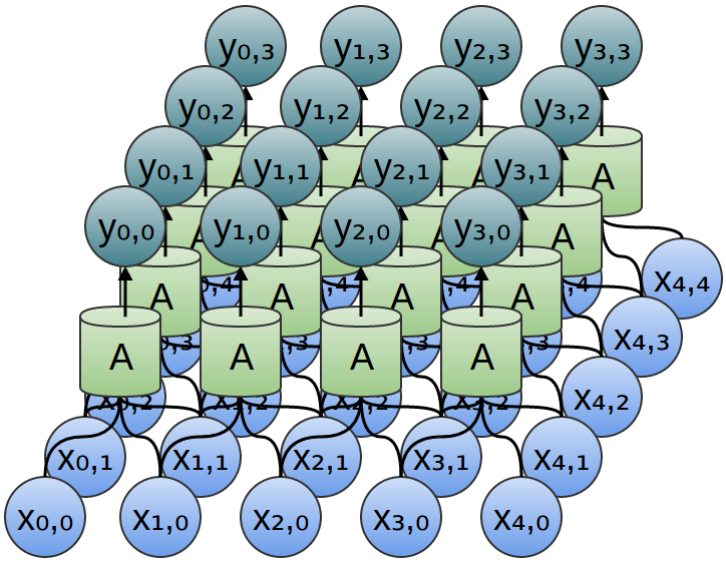


Рисунок 1.6 -Двовимірний згортковий шар

Згорткова нейронна мережа — структура нейронних мереж, спочатку створена і використана для ефективного розпізнавання зображень: чергуються згорткові шари (convolutions) з нелінійними активаційними функціями (ReLU або

гіперболічний тангенс tanh) і шари об'єднання (pooling layers).

На відміну від мережі прямого поширення, де кожен вхідний нейрон з'єднується з вихідним нейроном в наступному шарі, в згорткових мережах для Наприкінці прямого розсіювання зовнішня нейронна мережа з'єднує зовнішній нейрон із майбутньою сферою на фізичних кінцях. для отримання вихідних значень використовуються згортки по кожному вхідному шарі. В операції згортки використовується матриця ваг невеликого розміру, яка зсувається по всьому шару, що формується, формуючи після кожного зсуву сигнал активації для нейрона наступного шару з аналогічним положенням. Ця матриця називається ядром згортки; він використовується для різних нейронів вихідного шару. При виконанні операції згортки кожен фрагмент (наприклад, зображення) поелементно множиться на матрицю згортки, і результат підсумовується і записується в аналогічну позицію в оригінальному зображенні. Матриця згортки - це графічне кодування ознаки. Наступний шар, отриманий в результаті операції згортки, показує наявність цієї функції. У згортковій нейронній мережі існує багато наборів ваг, які кодують елементи зображення. Ядра згортки формуються під час навчання мережі. Під час передачі кожного набору ваг генерується карта об’єктів. Оскільки на одному шарі є багато незалежних карт об’єктів, мережа стає багатоканальною.

Кожен шар згортки має свій власний фільтр для кожного каналу, ядро згортки, яке обробляє попередній шар на фрагменти. Результат використання різних фільтрів поєднується. Це те, що зробили профспілки. Операція Subsampling зменшує розмірність згенерованих об’єктів карти. У цій архітектурі мережі вважається, що інформація про наявність потрібного ознаки важливіша за точне знання її координат, тому з ряду сусідніх нейронів на карті ознак вибирається максимум і приймається за нейрон на ущільненому карта об’єктів у менших розмірах. В результаті цієї операції, крім прискорення подальших обчислень, мережа стає інваріантною за розміром вхідного зображення.

Після початкового шару сигнал проходить серію свёрточних шарів, в яких чергується операції згортки і об'єднання (pooling). Чергування шарів дозволяє

складати карти ознак: на кожному наступному шарі карта зменшується в розмірі, а кількість каналів збільшується. Практично це означає здатність розпізнавати складні ієрархії функцій. Після проходження кількох шарів карта об'єктів вироджується у векторну або скалярну, але таких карт об'єктів сотні. На виході згорткових шарів мережі додатково встановлюються кілька шарів повної нейронної мережі (наприклад перцептрон), на вхід яких подаються підсумкові карти ознак. Мережеві гіперпараметри

Гіпепараметри згорткової нейронної мережі: Вузькі та вузькі згортки: заповнення нулями дозволяє зробити згортку широким, якщо, наприклад, перший елемент матриці не має суміжних елементів зліва та зверху. Без додавання нулів отримуємо вузьку згортку; Розмір кроку: розмір кроку визначає величину усунення фільтра на кожному кроці. Чим більший крок, тим менший застосовуваний фільтр і менше розмір вихідної матриці. Зазвичай використовується крок, рівний одиниці, але більший крок може дозволити вам побудувати модель, поведінка якої буде нагадувати рекурсивну нейронну мережу (тобто мережа з великим кроком буде виглядати як дерево); Об'єднання шарів: об'єднання шарів допомагає зменшити розмір вихідної інформації при збереженні найбільш видимої інформації. Наприклад, якщо фільтр визначає, чи запропонує заперечення. ("not good"). Якщо десь в реченні є ця фраза, то результат застосування фільтра до цього регіону дасть велике значення, але мале для інших регіонів. Після застосування операції максимуму для регіону, залишається тільки інформація, з'являлося чи заявлене заперечення в пропозицію, проте інформація про те, де воно з'являлося, зникає. Тобто інформація про місцезнаходження пропадає, а локальна інформація залишається (очевидно, що "not good" сильно відрізняється від "good not"); Вентиляційні канали (channels): канали-це різні "погляди" на вхідні дані. Наприклад, в розпізнаванні зображень, у нас зазвичай три канали -RGB. В обробці природної мови такими каналами можуть бути різні векторні уявлення слів (word2vec або GloVe), пропозиція на різних мовах або перефразовані пропозиції.

Типова структура Структура мережі є односпрямованої, а для навчання зазвичай

використовується метод зворотного поширення помилки. Мережа складається з великої кількості шарів (див. Рис. 1.7).

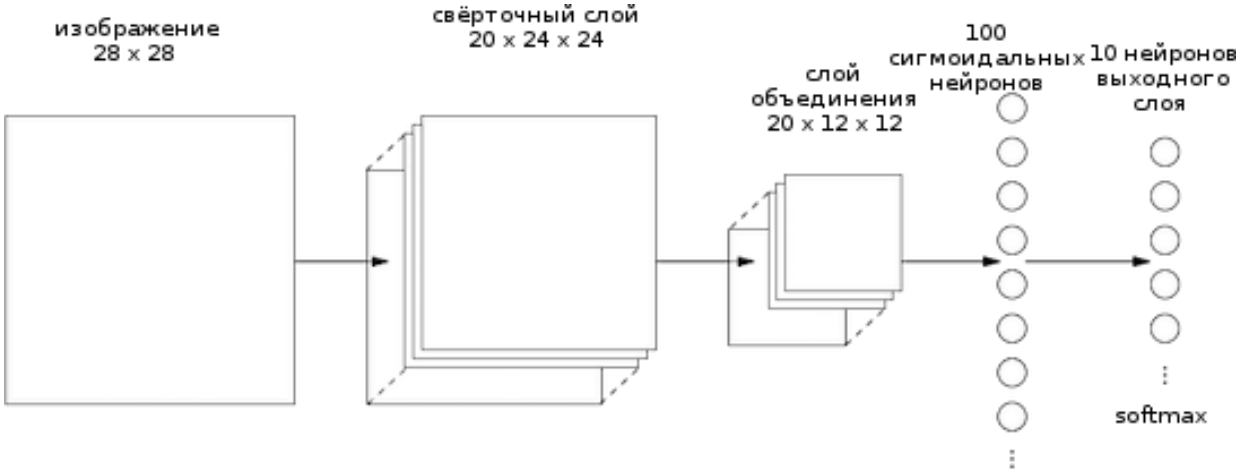


Рисунок 1.7 - Приклад архітектури згорткової нейронної мережі для розпізнавання об'єкта на зображенні розмірності 28x28 px

1.1.10.3 Застосування згортувальних нейронних мереж в аналізі тональності тексту

На вхід нейронної мережі буде подана матриця, кількість рядків якої залежить від розмірності словника, а ширина фільтрів дорівнює кількості стовпців цієї матриці (тобто використовується розмірність для кодування кожного слова). Висота (або розмір фрагмента вхідних даних) може змінюватися, але зазвичай становить близько 2-5 слів. Перші шари представляють слова як низьковимірні вектори.

Наступний шар виконує згортки векторного представлення слів за допомогою фільтрів різного розміру (тобто вони захоплюють 3-5 слів за раз). Потім за результатом згортки виконується об’єднання (max-pool). Раніше над результатом згортки. До отриманого довгому вектору ознак додаємо регуляризацію (dropout в цьому випадку). Нарешті, відбувається класифікація результату за допомогою шару softmax (див. Рис. 1.8).Як входів задаються не тільки стандартні X і Y, а й імовірність того, що нейрон виявиться в шарі дропаутів (дропаутів задається тільки під час тренування мережі). Перший шар - шар уявлення слів у вигляді векторів

word2vec - є таблицею перетворення (відповідності). Результат застосування цього шару - тривимірний тензор розмірності [None, sequence_length, embedding_size].

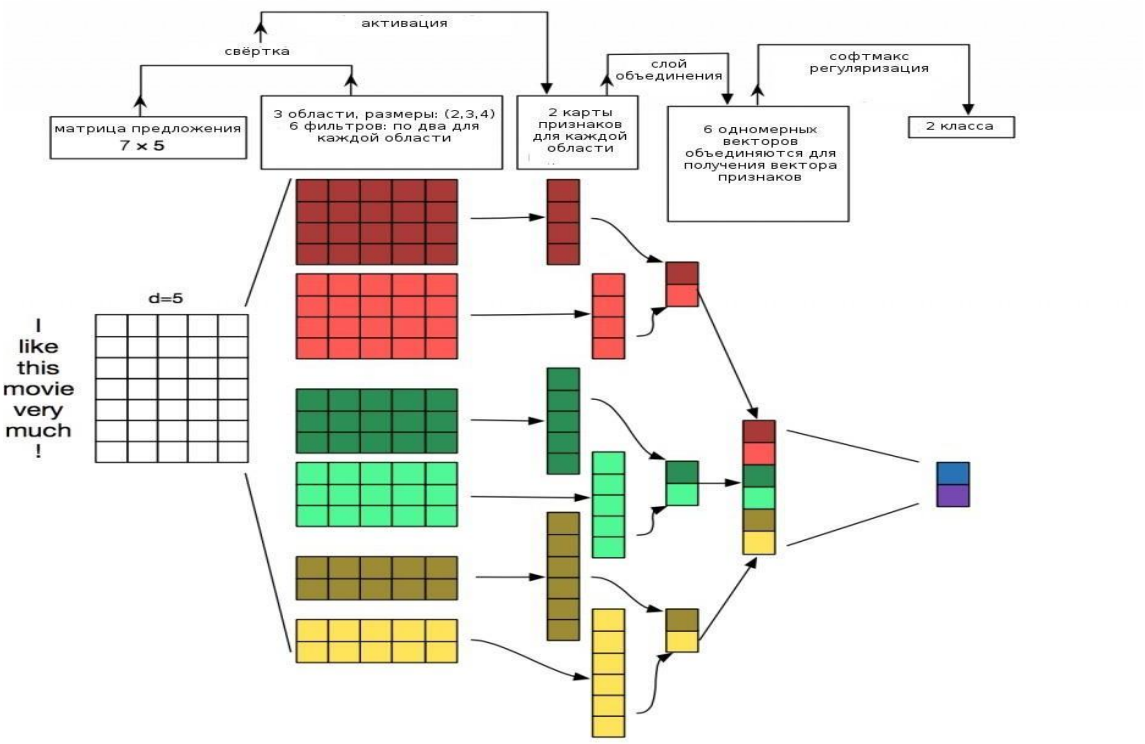


Рисунок 1.8 - Приклад архітектури згорткової нейронної мережі для класифікації пропозицій

Наступний шар згортки приймає на вхід 4-мірний тензор (Батч, ширина, висота і канал). До отриманого на попередньому кроці тензора просто додається новий вимір [None, sequence_length, embedding_size, 1]. Кожен фільтр проходить повністю через все уявлення, відмінність тільки в тому, скільки він обробляє слів. У даній роботі використовується вузька згортка (narrow convolution) - тому на виході тензор має форму [1, sequence_length - filter_size + 1, 1, 1]. Після застосування ПУЛІНГ над виходом певного фільтра вийде тензор форми [batch_size, 1, 1, num_filters] По суті це і є вектор ознак, останній вимір якого відповідає потрібним нам ознаками. Після отримання всіх тензорів, потрібно об'єднати їх в один довгий вектор ознак форми [batch_size, num_filters_total]

Найпопулярніший метод упорядкування випадання згорткових нейронних

мереж. Випадання, щоб заблокувати групу нейронів одного рангу. Ці «живі» коричневі плями діють самі по собі. Група нейронів, як би піклується про свою долю в навчанні, повинна початися з dropout_keep_prob як входу в примху.

Якщо ви спробуєте видалити зламаний векторний знак, вам може бути надано огляд того, який клас надати. Необхідно помножити матрицю і вібрувати клас з найкращим результатом. Намистинку softmax також можна використовувати для нормалізації. Vicory, зі значенням, ми можемо зробити функцію воріт. Наша мета мінімізувати втрати, як тільки ми отримуємо прощення (захоплення) нейронної мережі. Для цоя перехресна ентропія є переможною.В роботі використовується функція softmax_cross_entropy_with_logits. Потім береться середнє значення втрат. Як класифікуються пропозиції за допомогою даної архітектури:

Визначити розмірності трьох фрагментів даних: 2, 3 і 4, для кожного фрагмента існує два фільтра;

Кожен фільтр виконує згортку над матрицею пропозицій і генерує карти ознак (feature maps); Шар 1-max pooling обробляє кожну карту, тобто зберігається найбільша кількість з кожної карти. Отриманий одновимірний вектор ознак з карт об'єднується в вектор ознак для передостаннього шару;
Останній (softmax) шар отримує цей вектор на вхід і використовує його для класифікації пропозиції (бінарна класифікація).

Завдяки використанню згорткових шарів, кількість параметрів в них різко скорочується, і навчити мережу стає набагато простіше. А використовуючи різні види регуляризації (особливо dropout), вийшло значно зменшити перенавчання мережі. Нарешті, застосування ReLU замість сигмоїдальних нейронів допомогло прискорити навчання. Може здатися, що ідея свёрточних нейронних мереж не дуже застосовна для задач природної мови: дійсно, якщо на зображенні сусідні пікселі в основному є частиною одного і того ж об'єкта, то це невірно в разі слів у реченні (частини фраз можуть бути розділені іншими словами) . Можливо, рекурентні нейронні мережі - більш інтуїтивно зрозуміла модель (пропозиція представлено у вигляді дерева розбору), однак це не означає, що свёрточние мережі зовсім не

застосовні для поставлених в роботі задач.

1.1.10.4 Рекурентні нейронні мережі

У мережах прямого поширення використовується єдиний вхід, який повністю визначає активації всіх нейронів в останніх шарах. Таку мережу неможливо навчити передбачати події, наприклад, в сюжеті фільму - неясно, як би могла бути використана інформація про попередні події у фільмі. Рекурентні нейронні мережі покликані вирішити цю проблему.

Маючи всередині цикли, RNN дозволяє інформації зберігатися: поведінка прихованих нейронів буде визначатися не тільки

активацією в інших прихованих шарах, а й отриманими раніше активації самих нейронів. RNN може бути представлена в якості безлічі копій однієї і тієї ж нейронної мережі, де кожна копія передає повідомлення наступного копії. Тобто,

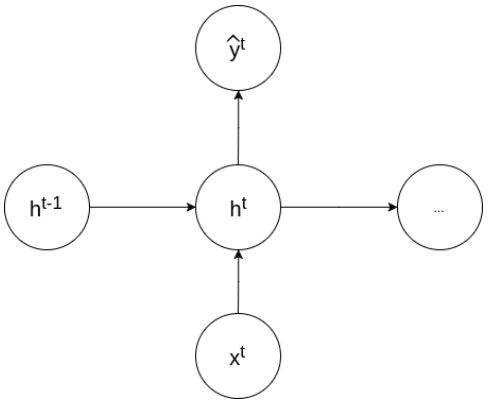


Рисунок 1.9-нейрони рекуррентной мережі

маючи цепеобразность структуру, як послідовності або списки, RNN є природною архітектурою нейронної мережі, що використовується для таких даних. RNN здатні обумовлювати модель за всіма попередніми обробленим словами з корпусу текстів. На рис. 3.10 прямокутник є прихованим шаром на часовому кроці t. Кожен шар містить нейрони (див. Рис. 3.9), кожен з яких виконує операцію лінійної матриці на своїх входах, за якою слідує нелінійна операція (наприклад, tanh). На кожному временномшаге вихідні дані попереднього кроку разом з наступним вектором

слова xt тексту, являє собою вхідні дані для прихованого шару для створення передбачення ŷ t і ознак ht

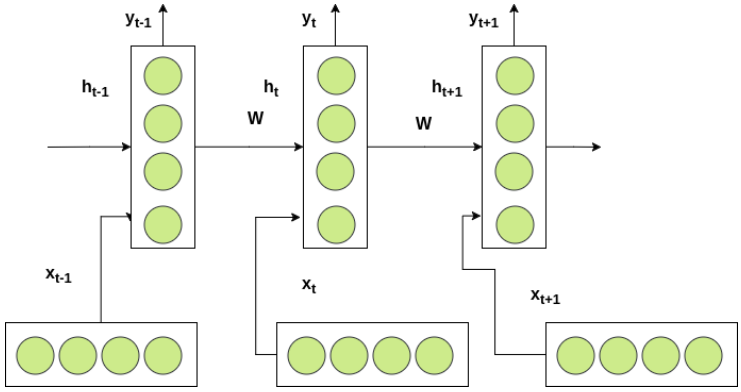


Рисунок 1.10 - Рекурентна нейронна мережа

Рекурентна нейронна мережа три тимчасових кроку Обсяг пам'яті, необхідний для запуску шару RNN, пропорційний кількості слів в корпусі текстів. пропозиція, що складається з k слів, буде зберігатися в пам'яті як k векторів. Розмір матриці ваг W що не масштабується відповідно до розміру корпусу текстів. Для рекуррентной мережі, що складається з 1000 рекурентних шарів, розмір матриці завжди буде 1000 × 1000, у незалежності від розміру корпусу текстів.

Рекурсивна і рекуррентна нейронні мережі

Рекурентні нейронні мережі повторюються (recurring) з плином часу. Нехай необхідно передбачити наступний символ після "hell" і "вгадує" наступну букву слова - "o"

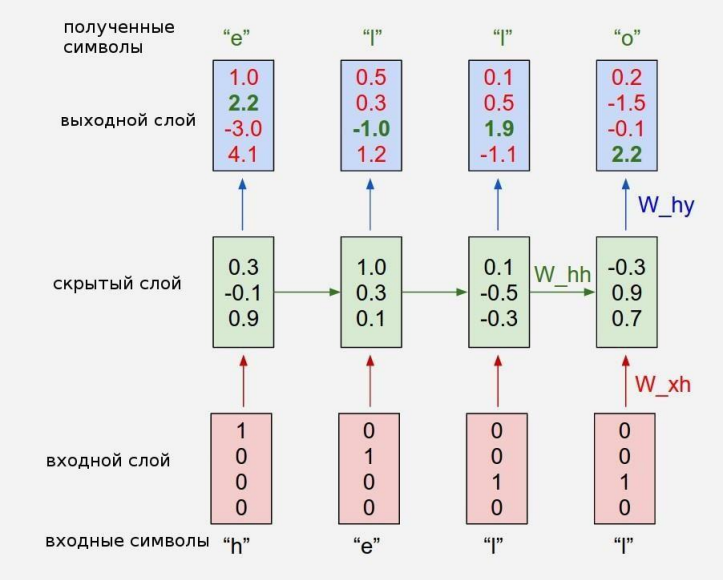


Рисунок 1.11 - Схема нейрона, що приймає на вхід послідовність

Важливо розуміти, що поняття рекуррентної і рекрсивної нейронних мереж це різні речі і потрібно розрізняти дані поняття. Рекурсивна нейронна мережа - це узагальнення рекуррентної. У рекуррентної мережі ваги загальні (і розмірність залишається однаковою) по всій довжині послідовності. У рекурсивної мережі ваги також загальні в кожному вузлі. Це означає, то все W_xh ваги будуть однаковими (загальними) і таким же буде вага W_hh через те, що все відбувається в єдиному нейроне, розгортається в часі (див. Рис. 3.12).

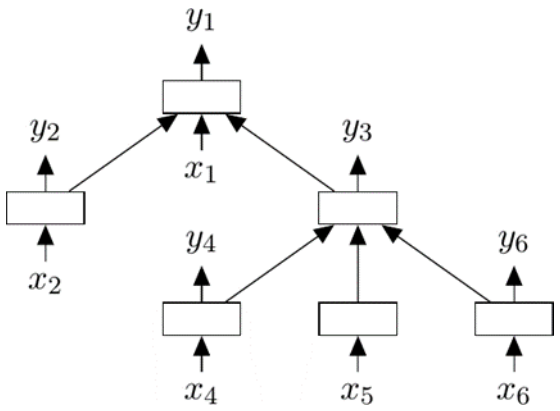


Рисунок 1.12-Схема рекурсивної нейронної мережі

Якщо мережі використовуються для генерації нових символів, то підійдуть

рекурсивні мережі. Однак для генерації дерева розбору краще використовувати рекурентні мережі. Проблема зникаючого градієнта на прикладі мовної моделі Нехай є мовна модель, за допомогою якої необхідно передбачити наступне слово, використовуючи попередні. Коли розрив між важливою інформацією і уривком, де вона необхідна, невеликий, RNN можна навчити використовувати інформацію, отриману раніше ("Хмари в небі"). Але якщо необхідний контекст, як в прикладі "Я виріс у Франції. Я вільно розмовляю французькою ", то розрив між суттєвою інформацією і місцем вставки ширшає. На жаль, у міру зростання розриву, RNN неможливо навчити пов'язувати інформацію. Однією з проблем RNN є те, що її ранні моделі дуже складно навчати через нестійкий градієнтного спуску. Навчання в попередніх шарах відбувається дуже повільно, так як градієнт стає все менше і менше при зворотному поширенні. Тобто якщо мережа працює досить довго, то градієнт може стати вкрай нестійким.

Логістична регресія

Статистична модель, використовувана для передбачення ймовірності виникнення деякої події. Scikit-learn (sklearn.linear_model.LogisticRegression). В роботі доданий єдиний параметр: random_state = 1. Наївний байесовский класифікатор Простий імовірнісний класифікатор, заснований на застосуванні теорема баєса зі строгими припущеннями про незалежність елементів вектора ознак. Перевагою наївного байєсівського класифікатора є мала кількість даних для навчання, необхідних для оцінки параметрів, необхідних для класифікації. Scikit-learn (sklearn.naive_bayes.GaussianNB). В роботі використовуємо наївний байесовский класифікатор Гаусса - розподіл ймовірностей ознак збігається з функцією Гаусса (тобто нормальний розподіл). σy і μy розраховані за методом максимальної правдоподібності. P (x_i | y) = 1 / √ (2πσ_ŷ 2) exp (- (x_i-μ_y) ^ 2 / (2σ_ŷ 2)) (2.1.1) 50

Випадковий ліс (random forest)

Алгоритм полягає в використанні комітету вирішальних дерев. Класифікація об'єктів проводиться шляхом голосування: кожне дерево комітету відносить

класифікується об'єкт до одного з класів, і перемагає клас, за який проголосувала найбільша кількість дерев. Оптимальне число дерев підбирається таким чином, щоб мінімізувати помилку класифікатора на тестовій вибірці. Використовує усереднення для підвищення точності прогнозування і контролю надлишкової підгонки. Розмір підвибірки завжди збігається з розміром оригінальної вибірки. Scikit-learn (sklearn.ensemble.RandomForestClassifier). дерев (n_estimators) в роботі одно 100. Чим їх більше, тим краще, але може виникнути як проблема перенавчання, так і проблеми з пам'яттю пристрою. Метод опорних векторів (SVM) Основна ідея методу - переклад вихідних векторів в простір більш високої розмірності і пошук розділяє гиперплоскости з максимальним зазором в цьому просторі. Стандартна функція з scikitlearn (sklearn.svm.SVC) не підходить, оскільки тимчасова складність даного алгоритму квадратична. Ефективність методу опорних векторів значно знижується, якщо кількість прізнакових описів дуже велике. Він має велику гнучкість у виборі штрафів і функцій втрат і повинен краще масштабироваться для великої кількості зразків. Оптимізація функції втрат SVM за допомогою градієнтного спуску: кількість L (ω, D) = 1/2 ‖ω‖_2 ^ 2 + CΣ_ (i = 1) N⁒ ⟦max (0,1-y_i (ωx_i + b)),⟧ (2.1.2) де D = ⟦{(x_i, y_i)}⟧ _ (i = 1) ^ N, x_i εRd і y_i ε {-1, + 1} (2.1.3)

1.1.11 Обзор засобів розробки

1.1.11.1 Рушій Unity

Unity - кросплатформове середовище для розробки комп'ютерних ігор , розроблене для світового ринку американською компанією Unity Technologies . Unity дозволяє вести розробку програм, що можуть функціонувати на більш ніж 25 різних платформах , до яких входять такі платформи як персональні комп'ютери , ігрові консолі , мобільні пристрої , інтернет-додатки та інші . Випуск Unity в світ відбувся у 2005 році і з того часу компанія постійно підтримує продукт та періодично випускає оновлення, триває постійний

розвиток. Основними перевагами Unity є наявність та підтримка візуального 2д та 3д середовища розробки , кросплатформна підтримка та модульні компоненти системи. До недоліків можна віднести складність роботи з багатокомпонентними схемами та труднощі із підключенням зовнішніх бібліотек. Unity написала тисячі ігор, програм, візуалізацій математичних моделей, що охоплюють безліч платформ та жанрів. У цьому Unity використовують як великі розробники, і незалежні студії. Редактор Unity має простий інтерфейс Drag & Drop, а також встановлення плагінів KALI, які легко налаштовуються та складаються з різних вікон, тому ви можете налагоджувати гру прямо у редакторі. Двигун використовується для написання скриптів C#.Раніше також підтримувалися Boo (діалект Python, підтримка видалена у версії 5) та модифікація JavaScript, відома як UnityScript (підтримка припинена у версії 2017.1). Фізичні розрахунки виконуються фізичним двигуном NVIDIA PhysX. Графічний API – DirectX (нині DX 11, підтримується DX 12) Проект Unity поділений на сцени (рівні) – окремі файли, які містять свої ігрові світи з власним набором об'єктів, сценаріїв та налаштувань. Сцени можуть містити як, власне, об'єкти (моделі), і порожні ігрові об'єкти - об'єкти, мають моделі («пустушки»). Об'єкти, своєю чергою, містять набори компонентів, із якими взаємодіють скрипти. Також об'єкти мають назву (в Unity допускається наявність двох і більше об'єктів з однаковими назвами), може бути тег (мітка) і шар, на якому він повинен відображатися. Так, у будь-якого об'єкта на сцені повинен бути компонент Transform - він зберігає координати розташування, повороту та розміру об'єкта по всіх трьох осях. Об'єкти з видимою геометрією також мають компонент Mesh Renderer, який робить модель видимою. Колізії можуть застосовуватися до об'єктів (в Unity, так званих колайдерів), яких існує кілька типів. Unity також підтримує фізику твердого тіла та тканини, а також фізику Ragdoll (ганчіркова лялька). У редакторі є система наслідування об'єктів; дочірні об'єкти будуть повторювати всі зміни положення, повороту та масштабу батьківського об'єкта. Скрипти у редакторі прикріплюються до об'єктів як окремі компоненти.

В Unity включено систему контролю версій для ігрових об'єктів та скриптів під назвою Unity Asset Server. Система використовує PostgreSQL, роботу зі звуком, побудовану на основі бібліотеки FMOD (з можливістю програвати Ogg Vorbis аудіофайли), відеопрогравач із кодеком Theora, рушій для побудови ландшафтів рослинності, вбудовану систему карт освітлення (Beast), мережу для мультиплеєру (RakNet) та вбудовані навігаційні меші для пошуку шляху. Сервер наборів ресурсів Unity — це платне доповнення, що додає інструментарій для спільної розробки на базі Unity багатьома користувачами одночасно та контроль версій у функціоналі Unity.

Багатогігабайтні проєкти з тисячами мегабайтних файлів піддаються легкому керуванню. Налаштування імпорту та інші метадані також зберігаються разом з історією їх версій. Переглядати зміни ресурсів\версій можна одразу всередині редактора Unity. Якщо файли змінюються, їх статус негайно оновлюється. Перейменування і переміщення ресурсів не створює будь-яких перешкод для безперервного робочого процесу. Сервер ресурсів Unity управляється базою даних PostgreSQL.

Сервер ресурсів доступний як для Mac OS X Installer, так і для Linux RPMs . Підтримка декількох платформ забезпечує гнучкість у впровадженні Сервера ресурсів Unity у наявну IT-інфраструктуру

1.1.11.2 Visual Studio

Microsoft Visual Studio - це лінійка продуктів Microsoft, яка включає інтегроване середовище розробки програмного забезпечення та низку інших інструментів. Ці продукти дозволяють розробляти як консольні, так і графічні програми, включаючи підтримку технології Windows Forms, а також веб-сайти, веб-програми, веб-служби як у власному, так і в керованому коді для всіх платформ, підтримуваних Windows, Windows Mobile, Windows CE, . Framework, Xbox, Windows Phone .NET Compact Framework та Silverlight. Visual Studio включає в себе редактор вихідного коду з підтримкою технології IntelliSense і можливістю

найпростішого рефакторінга коду. Вбудований відладчик може працювати як відладчик рівня вихідного коду, так і відладчик машинного рівня. Інші інструменти плагінів включають редактор форм для спрощення створення графічного інтерфейсу програми, веб-редактор, конструктор класів і конструктор схем бази даних. Visual Studio дозволяє створювати та підключати сторонні програми (плагіни) для розширення функціональності майже на кожному рівні, включаючи додавання підтримки систем контролю версій вихідного коду (таких як Subversion і Visual SourceSafe), додавання нових наборів інструментів (наприклад, для редагування). і візуальний дизайн) ... код на предметно-специфічних мовах програмування) або інструменти для інших аспектів процесу розробки програмного забезпечення (наприклад, клієнт Team Explorer для роботи з Team Foundation Server).

Можливості:

1. Підсвічування синтаксису - виділення синтаксичних конструкцій тексту різними кольорами, шрифтами та гарнітурами. Він використовується для покращення читання вихідного тексту комп'ютерних програм, покращення візуального сприйняття.

2. Згортання коду, або згортання (англ. Folding) - одна з функцій текстового редактора, що дозволяє приховати певний фрагмент відредагованого коду або тексту, залишивши лише один рядок. Наприклад, згортання функції згортає весь код функції в один рядок, щоб було видно лише назву функції. Зазвичай, щоб згорнути фрагмент, потрібно натиснути на "?" ліворуч від нього. Щоб побачити весь фрагмент, тобто розгорнути його, потрібно натиснути на символ «+», що з’являється у згорнутих фрагментах.

3. Код автозаповнення - функція у програмах, які забезпечують інтерактивне введення тексту (редактори, оболонки командного рядка, браузери тощо. буд.) доповнення тексту на введеної його частини.

4. Вбудований налагоджувач – використовується на етапі налагодження комп’ютерної програми, в якій виявляють, локалізують та усувають помилки. Щоб

зрозуміти, де сталася помилка, потрібно:

з'ясувати поточні значення змінних;

з'ясувати, яким способом була виконана програма.

5. . Модульне тестування – модульне тестування або модульне тестування (англ. Unit testing) – процес у програмуванні, який дозволяє перевіряти правильність одиниці вихідного коду, наборів одного або декількох програмних модулів з відповідними даними управління, використання та процедур обробки. тому писати тести для кожної нетривіальної функції чи методу. Це дозволяє швидко перевірити, чи не призвело чергове зміна коду до регресу, тобто. помилок у вже перевірених місцях програми, а також полегшує виявлення та усунення таких помилок.

РОЗДІЛ 2. ПРОЕКТУВАННЯ ТА ТЕОРИТИЧНЕОБГРУНТУВАННЯ СИСТЕМИ

2.1.Постановка завдання

Функціональні вимоги до системи

1. Надання користувачеві проаналізувати власний текст. Можливості

2. Реалізація алгоритму поновлення даних в реальному

Розпізнати тональність відкликання. Розбити відгук на окремі слова.

Видалення зайвої інформації з тексту (небажані символи). 6.7. 8. 9. Провести морфологічний аналіз тексту. Провести лематизації.

Побудувати векторну модель з тексту. Навчити нейронну мережу. Чи не Функціональні вимоги

Реалізація алгоритму оновлення даних у реальному часі.

3. Розпізнати тональність відкликання.

4. Розбити відгуки на окремі слова.

5. Видалення зайвої інформації з тексту (небажані символи).

6. Провести морфологічний аналіз тексту.

7. Провести лематизацію.

8. Побудувати векторну модель із тексту.

9. Навчити нейронну мережу.

1.інтернету.2.Наявність персонального комп'ютера, що має доступ до Забезпечити запис в базу даних оцінки аналізованого Тексту У разі неможливості правильно аналізувати текст, видавати повідомлення про помилку. Переносимість на різні комп'ютерні платформи.

3. Простота в експлуатації досвідченим оператором.

Не функціональні вимоги

1. Наявність персонального комп'ютера, що має доступ до інтернету.

2. Забезпечити запис до бази даних оцінки аналізованого тексту.

3. У разі неможливості правильно аналізувати текст, видавати повідомлення про помилку.

4. Перенесення на різні комп'ютерні платформи.

5. Простота експлуатації досвідченим оператором.

2.2 Програмні вимоги

Сервер з підтримкою python 3 і баз даних на мові SQL Мережеві протоколи Операційні системи, що підтримуються SQL, містять вбудоване ПО з підтримкою мережевих протоколів: іменовані канали, спільна пам'ять і TCP / IP. Жорсткий диск Потрібно мінімум 6 Гб вільного місця на диску. Монітор Потрібно монітор з роздільною здатністю 800x600 пікселів або вищим. Інтернет Потрібно доступ в Інтернет для підтримки функціональних засобів Інтернету.

2.3 Вимоги до обладнання

ОЗУ Мінімально потрібно: 1 Гб 61
Рекомендовані вимоги: 4 Гб зі збільшенням у міру зростання розмірів бази даних.
Продуктивність процесора Мінімально потрібна: процесор з архітектурою x64 та тактовою частотою 1.4 ГГц.
Рекомендовані вимоги: процесор з архітектурою x64 та тактовою частотою 2.0 ГГц або вище.
Тип процесора Процесор архітектури x64: AMD Athlon 64, Intel Xeon з підтримкою Intel EM64T, Intel Pentium IV з підтримкою Intel EM

2.4 Показники якості нейронної мережі

Алгоритм навчання нейронної мережі Для того щоб нейронна мережа почала

виконувати свої завдання її необхідно навчити, процес навчання відбувається за принципом показаним

Показники якості нейронної мережі Частка коректних прогнозів (accuracy) - відсоток помилок, які допускаються класифікатором. Наступні показники будуть використані лише для класичних методів класифікації Міра точності (precision) – відношення tp до (tp+fp), де tp – кількість істинних позитивних величин, а fp – кількість хибних позитивних величин. Тобто міра точності характеризує скільки одержаних від класифікатора позитивних рішень вважаються вірними; • Міра повноти (recall) – відношення tp до (tp + fn), де fn – кількість хибних негативних величин. Міра повноти встановлює вміння класифікатора дізнаватися так само як і можливе найбільше позитивних рішень з прогнозованих; Міра F1 - середнє гармонічне міри точності та міри повноти. Визначає лімінальну властивість класифікатора; Носій міри (support) - кількість інформації кожного з класів. Найбільш жорстке визначення: мінімальне закрите велике число, в якому сконцентрована міра.

2.5 Розробка проекту

Фізика руху автомобіля

Оскільки ми робимо аркадну гру, нам не потрібна фізично точна модель автомобіля. Достатньо того, щоб автомобіль поводився візуально реалістично. При цьому модель повинна підтримувати рух вперед/назад, рульове управління, гальмо і робити реалістичні замети при великій швидкості руху. Зміна передачі, зміщення центру мас та інші складніші процеси моделювати не будемо. Принаймні поки що. Для початку умовимося, що моделювати автомобіль ми будемо лише на площині. Тому позиція автомобіля та напрямок руху - будуть двомірними векторами. На етапі моделювання в Unity наші автомобілі будуть тривимірними, але управління, "мозки" та фізика буде двовимірною. векторів. Для початку розглянемо простий рух автомобіля, коли немає заметів і вектор руху

машини збігається з напрямком корпусу. В такому випадку можна вважати, що всі колеса рухаються з однаковою швидкістю Speed. При цьому рух задніх коліс збігається з напрямком корпусу автомобіля, а передні колеса рухаються під певним кутом до корпусу (залежно від кута обертання керма). Картинка пояснює як зрушуватимуться колеса:

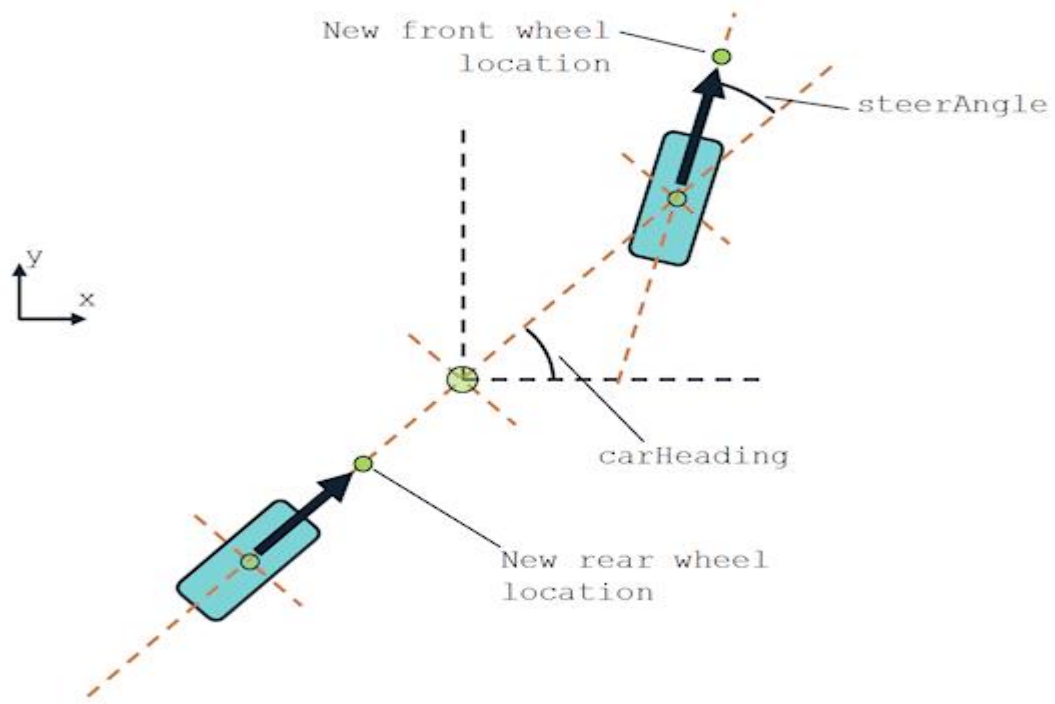


Рисунок 2.1 Схема фізики коліс

З'ясувавши, як зрушать колеса, ми можемо визначити нові координати центру корпусу та новий кут повороту корпусу:

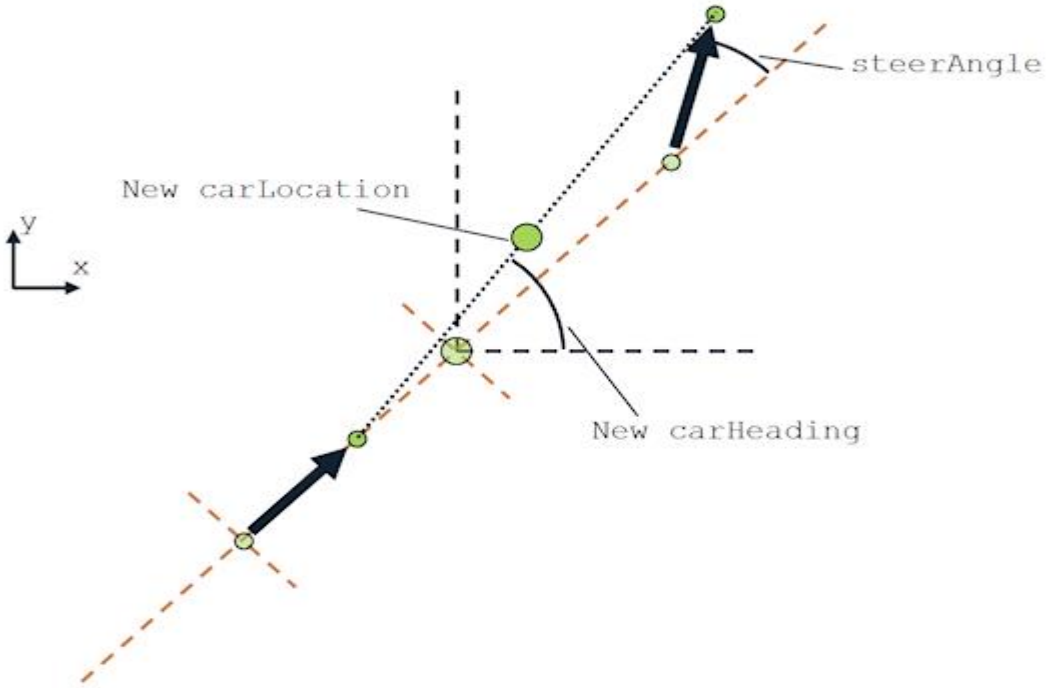


Рисунок 2.2 Схема заносу об’єкта

Тепер розглянемо ситуацію, коли машину заносить. При занесення вектор руху автомобіля не збігається з орієнтацією корпусу:

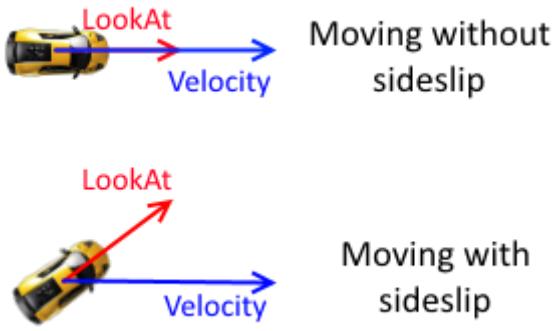


Рисунок 2.4 Вектор руху при заносі

При повороті вектор швидкості ніби запізнюється від вектора напрямку корпусу. І в момент занесення ці вектори не збігаються. У формулах це можна виразити так:

NewVelocity = Velocity * (1 - k) + LootAt * Speed * k

де Speed - довжина вектора Velocity, а k - коефіцієнт тертя (від 0 до 1) і залежить

від величини зчеплення колеса з трасою. Якщо прийняти k = 1 (абсолютне зчеплення з трасою) і підставити у формулу, то побачимо, що вектор NewVolicty збігатиметься з напрямком корпусу LookAt:
NewVelocity = LootAt * Speed
Навпаки, якщо k = 0 (абсолютно слизька траса), то вектор швидкості взагалі не залежатиме від напряму корпусу: NewVelocity = Velocity

Якщо ж k буде між 0 і 1, то отримаємо рух при якому вектор швидкості поступово буде наближатися до напрямку корпусу - тобто буде рух із занесенням. При цьому, чим менша абсолютна швидкість, тим меншою буде різниця між векторами LookAt та Velocity. Чим більша швидкість - тим різниця між цими векторами буде більшою.
Лінійне прискорення автомобіля вважатимемо за класичною формулою Ньютона:
Force = Throttle * EnginePower; - сила тяги
NewVelocity = Velocity + LookAt * (Force * dt / Mass); - второй закон Ньютона F = ma де Throttle - коефіцієнт "вичавлення газу" (від 0 до 1), EnginePower - потужність двигуна в умовних одиницях, LookAt - вектор напрямку корпусу автомобіля, Mass - маса автомобіля, dt - збільшення часу.
Звертаємо увагу, що прискорення завжди у напрямку збігається з орієнтацією корпусу LookAt. Це вірно навіть у разі занесення, оскільки прискорення створюється задніми колесами, а вони завжди спрямовані вздовж корпусу автомобіля.
Далі, для моделювання тертя про повітря було використано таку формулу:

NewVelocity = Velocity - Velocity * (AirFriction * dt)

Ця формула відображає той факт, що сила тертя (а значить і прискорення) пропорційна швидкості руху тіла, і протилежна у напрямку. AirFriction - коефіцієнт тертя повітря, dt - збільшення часу.
Моделювання гальма аналогічне моделюванню тертя повітря, але коефіцієнт тертя береться значно більшим.
Звівши всі формули разом, створюємо клас CarBase, який моделює рух автомобіля:

Після створення фізичної моделі автомобіля, було зроблено модель треку та сенсори для розроблюваної машинки.

2.3.1Модельтреку

Зазвичай, моделювання треку роблять у 3D двигунах. Це робиться тому, що в 3D движку типу Unity можна зробити стінки треку у вигляді колайдерів і потім шукати перетин променів сенсорів автомобіля зі стінками. Таким чином, реалізується машинний зір, за допомогою якого машина "бачить" трек. Я так не робитиму. По-перше, тому що я хочу зробити двигун незалежним від Unity. А по-друге, моделювання в Unity буде відносно повільним. Якщо реалізувати трек і зір без Unity, то можна зробити високошвидкісне моделювання (поза реальним часом) за допомогою якого ми дуже швидко навчатимемо нашу нейронну мережу.

Даний трек складається зі з'єднаних між собою точок (клас PathPoint), що утворюють замкнутий контур. Ключові точки малюються у спеціальному редакторі.

Трек є двозв'язним кільцевим списком точок типу PathPoint:
Кожна точка містить координати відрізків лівої та правої стінок треку, а також іншу інформацію - довжину ділянки треку, що визначається цією точкою, ширину трека в даному місці, вектор напряму та нормалі треку, та ін.

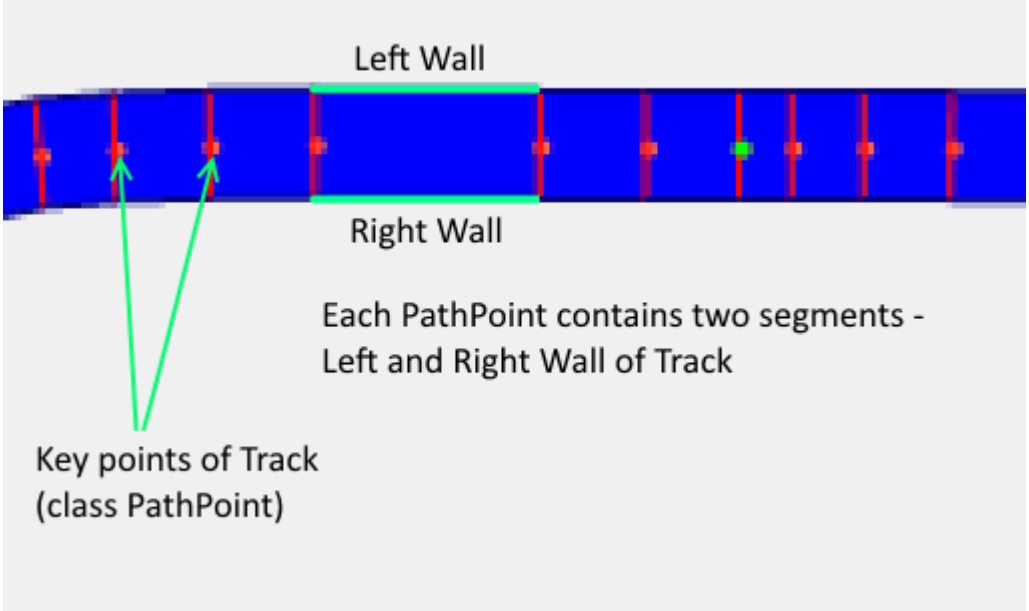


Рисунок 2.4 Схема побудови треку

При русі автомобіля по треку, машина буде переходити від однієї точки треку до іншої, а оскільки точка PathPoint містить відрізки для стінок, то ми завжди можемо швидко отримати найближчі до машини стінки треку і таким чином швидко та ефективно реалізувати обчислення відстані від машини до стін.

До речі, трек може мати самоперетинання, що дозволить у майбутньому зробити треки у вигляді вісімки або складніші фігури. Редактор треку дозволяє швидко намалювати трек, відредагувати його, встановити ширину треку в різних точках, задати коефіцієнт тертя для траси. В результаті можна зробити досить складні траси, наприклад:

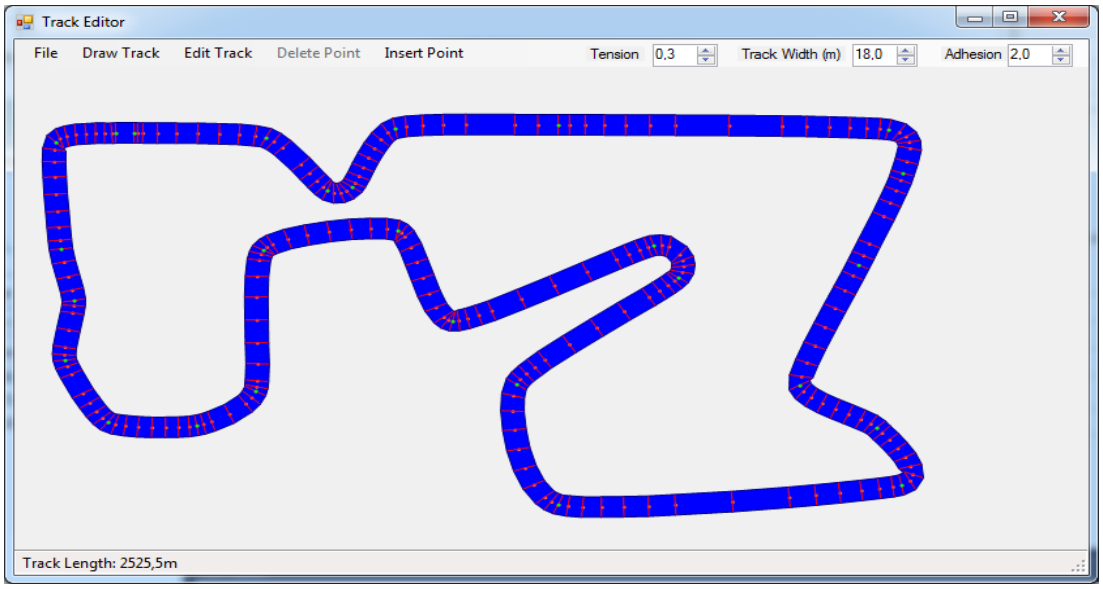


Рисунок 2.5 Приклад треку

Редактор треку зберігає трек у вигляді XML файлу, а також зберігає спеціальне зображення треку, за допомогою якого потім у Unity зможемо автоматично створити трек на террейні.

2.3.2 Зір для автомобіля

Для отримання інформації про стінки треку яким рухається автомобіль, використано п'ять сенсорів, кожен з яких дає інформацію про відстань від автомобіля до стіни в заданому напрямку:

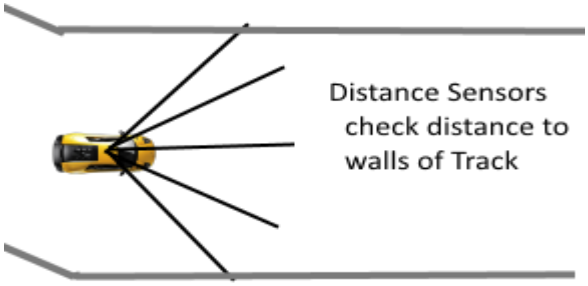


Рисунок 2.6 Схема розташування сенсорів

Після створення моделі автомобіля та треку все готово для головного – створення ІІ на базі нейронної мережі для керування машиною. Нейронна мережа

Для керування автомобілем будемо використовувати пов'язну нейронну мережу з трьома шарами. Топологія 6-6-6-3.
Число вхідних нейронів – 6, число вихідних – 3.
На вхід нейронної мережі було подано показання п'яти сенсорів (тобто відстані до найближчих стін треку) та шостий показник – власна швидкість автомобіля в даний момент. Три вихідні нейрони будуть визначати кут повороту керма, натискання газу, і гальмо.
Усі вихідні нейрони можуть видавати значення від -1 до 1. Для вхідних нейронів обмежень на значення немає.
Нейронна мережа виглядає так:

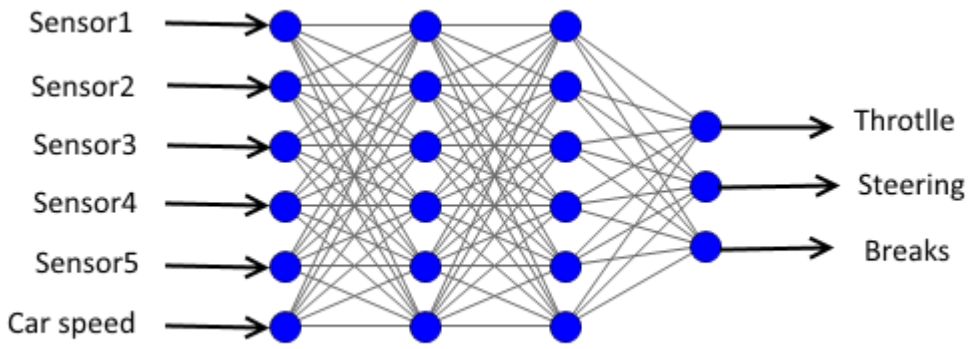


Рисунок 2.7 Схема нейронної мережі

Спочатку ваги зв'язків задаються випадковим чином.
Для активації нейронів використовується функція SoftSign

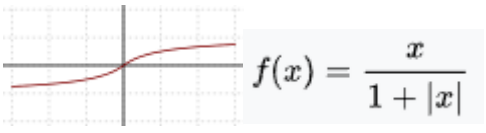


Рисунок 2.8 Функція активації нейронів

Ця функція підходить для мереж глибокого навчання (deep learning NN). Число шарів та топологія НС 6-6-6-3 були знайдені експериментально.

Якщо задати менше шарів або зменшити число нейронів у проміжних шарах - мережа починає погано навчатися і не виходити на прийнятну поведінку. Якщо ж збільшити кількість шарів чи нейронів – починається хаотична поведінка та повільне навчання.

Навчання нейронної мережі за допомогою генетичного алгоритму
Зазвичай мережа навчається за допомогою спеціальних методів – наприклад методу зворотного розповсюдження помилки. Але в проекті застосовано інший підхід – генетичний алгоритм.
Суть методу у тому, що генерується набір різних випадкових нейронних мереж (популяцію) і тестуємо їх (даючи кожній НМ машинку, і даючи їй покерувати). Далі ми оцінюємо ефективність кожної мережі за якоюсь ознакою (для керування автомобілем у гонках вибрано максимальний шлях, який змогла проїхати машинка вздовж треку, не стикаючись зі стінками).

Далі відбуваються процеси відбору найкращих СР. З них формується нова популяція, ваги зв'язків у нейронах випадково змінюються (за допомогою мутацій та рекомбінації генів). Нова популяція знову тестується на треку і повторюється. В результаті, з кожним новим поколінням ми відбираємо найкращі НМ, і еволюційний процес дає нам усі найкращі та найкращі результати, які буде показувати наша популяція НМ.
Кожне нове покоління НМ формується за допомогою трьох процедур – селекції, рекомбінації та мутації.

2.3.3Селекція

Завдання селекції - відібрати найкращі генотипи (НС) з популяції та створити на їх основі нову популяцію такого ж розміру, як і вихідна. Для відбору використовується метод, який називається Remainder Stochastic Sampling .
Це відбувається так: після того, як пройшов процес моделювання, для всіх особин (НМ) задається оцінка їх успішності (для перегонів - це пройдений шлях вздовж треку). Далі вважається середня оцінка популяції. Потім їх популяції відбираються

ті особи, які мають оцінка вище середньої (їх приблизно половина). І далі ці особини клонуються, причому кількість клонів - пропорційно оцінці батька. Підсумкове число популяції у своїй стає рівним розміру вихідної популяції. Таким чином ми відбираємо та розмножуємо кращі НР, і при цьому зберігаємо розмір популяції.

Мутація

Це простий процес у якому з популяції вибирається певне число НС і них випадкові зв'язку змінюються на випадкову невелику величину. В результаті ми отримуємо нову НС з трохи іншими вагами:

Рисунок 2.9 Мутація

2.3.4Рекомбінація

Це процес обміну генами (тобто вагами нейронних зв'язків) між двома НР. Інакше цей процес також називається кросинговер або просто схрещування (crossover).

Для рекомбінації з популяційного набору відбираються випадково дві НС і відбувається обмін вагами між ними:

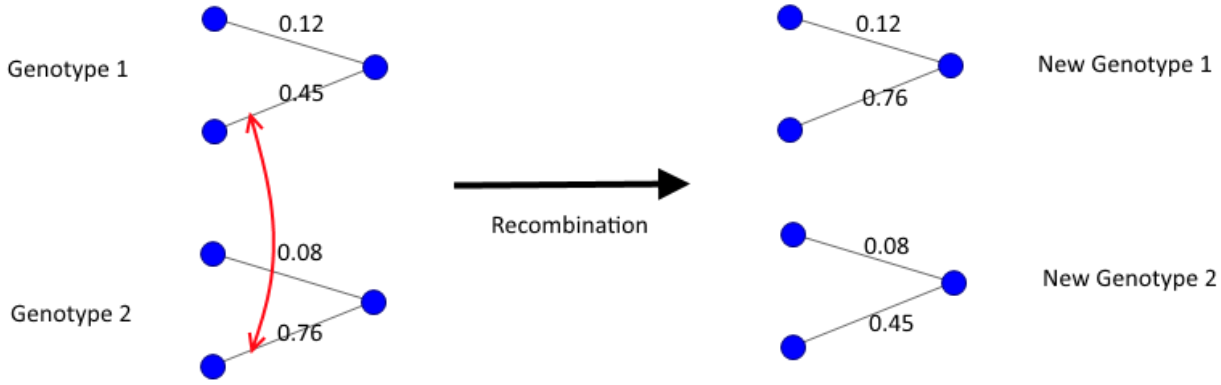


Рисунок 2.10 Рекомбінація

В результаті було отримано дві нові особи (НС) які поєднують у собі зв'язки двох НМ – предків.

Для зручності роботи з генетичними нейронними мережами було винесено роботу з НМ в окремий проект та створено клас GeneticNN. Цей клас інкапсулює всю роботу з генетичного відбору НР і має простий зовнішній інтерфейс. При створенні GeneticNN необхідно вказати топологію нейронної мережі та розмір популяції. GeneticNN сам створить необхідну кількість нейронних мереж із необхідною топологією.

Потім починається навчання НМ, використовуючи всього три методи:

1. public NeuralNetwork GetNN(int index) - отримання НР для особи з індексом index
2. public void SetEvaluation(int index, float evaluation) - завдання оцінки успішності для особи index
3. public void BuildNextGeneration() - генерація наступного покоління НС

Таким чином, алгоритм навчання СР виглядає так:

1. Створюємо об'єкт GeneticNN з потрібною топологією та потрібним розміром популяції.
2. Отримуємо нейронні мережі кожної особи методом GetNN(), і з допомогою них проводимо моделювання поведінки кожної особи.
3. Встановлюємо оцінку моделювання для кожної особи методом SetEvaluation.

4. Генеруємо нове покоління методом BuildNextGeneration
5. Починаємо наступну ітерацію навчання, переходячи до пп2.
6. Після проведення певної кількості ітерацій, відбираємо особину з максимальною оцінкою. Ця НР і буде найкращим результатом навчання.

Повністю код GeneticNN можна знайти у приєднаному файлі. Зрозуміло, GeneticNN можна використовувати не тільки для створення НР управління автомобілем, а для будь-яких завдань, в яких необхідно навчити і використовувати НМ.

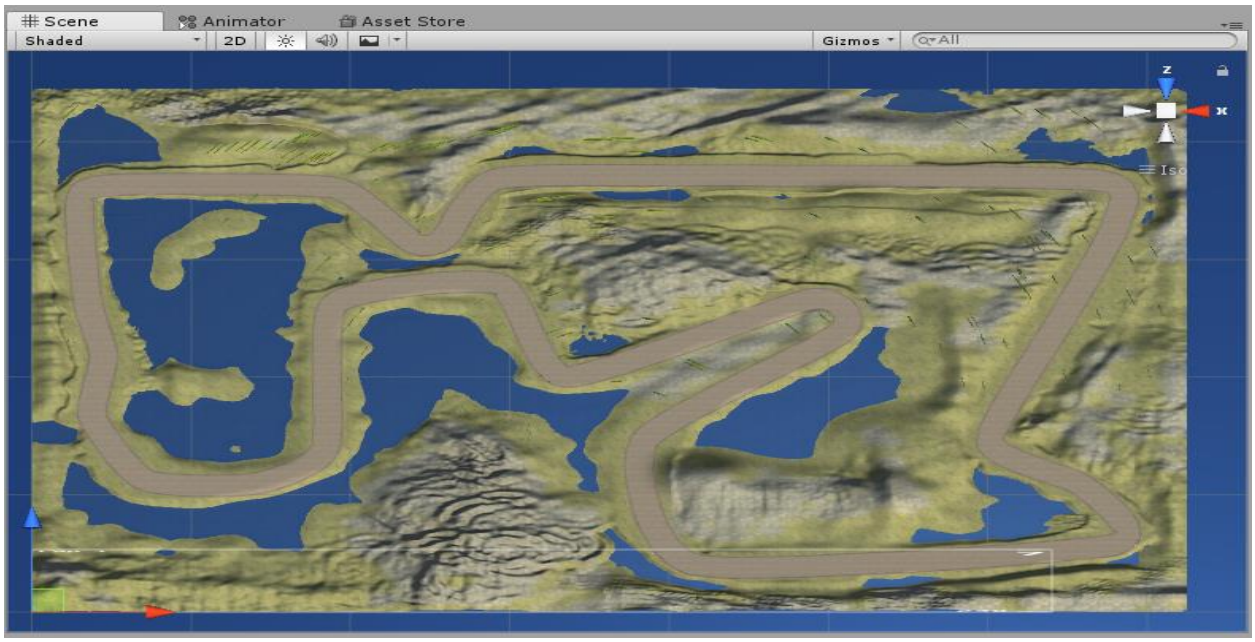
2.3.5 Результати навчання нейронної мережі

Для навчання мережі було зроблено спеціальний трек, який включає як ліві повороти так і праві, довгі прямі на яких потрібно розганятися і короткі сегменти дороги де потрібно посилено керувати. Також є шикани та шпильки. Цей трек добре підходить для універсального навчання. НМ яка успішно кермуватиме на цьому треку добре рулюватиме і на інших треках.
Розмір популяції – 60 особин. Перша популяція нейронних мереж поводиться досить незграбно. Більшість з них просто відразу врізається в стіни.
Проте вже на 2-3 поколінні з'являються особини, які можуть більш-менш упевнено триматися в рамках треку:

Ще через пару поколінь у нас вже є НР які можуть проїхати трек повністю жодного разу не врізавшись у стіни: Подальша еволюція йде вже для досягнення максимальної швидкості подолання треку.
Через десяток поколінь популяція вже їздить набагато краще:

Редактор треків, який було описано раніше, зберігає спеціальну текстуру із зображенням треку.
Даний механізм було експортовано до проекту Unity та за допомогою нього сформовано текстуру треку на террейні.
Це робиться за допомогою такого коду:

В результаті поверх террейну накладається трек:



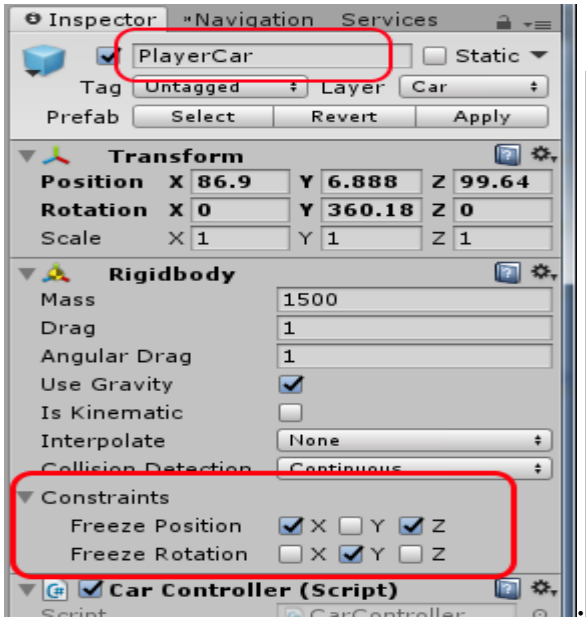
Крім текстури треку, відкривається в Unity та XML файл самого треку.

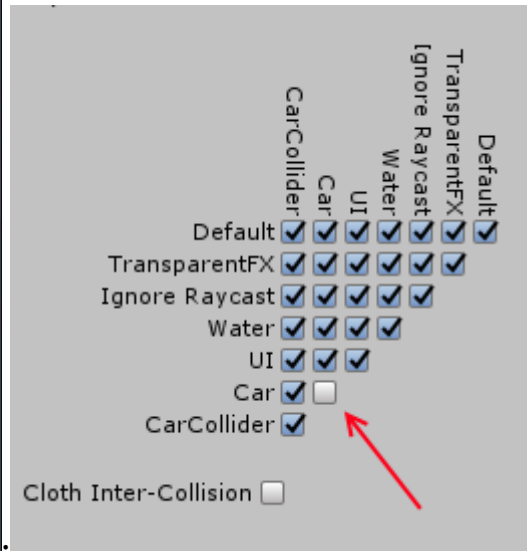
2.3.6Модельмашини

Для проекту було завантажено просту модель машинки, у якої корпус і колеса йдуть окремими мешами.
До машини вішаємо Rigidbody та колайдер.
Тут потрібно докладніше зупинитися на тому, як керуватиметься дана машинка. Оскільки нейронна мережа навчалася на плоскому двомірному треку, то й їздити вони мають по плоскому двовимірному треку. Для цього було завантажено трек, який використовувався при симуляції і наша СР їздитиме саме в цьому двовимірному треку. керування машинкою нічого не знає і не повинна знати. Просто до двовимірної позиції ми прив'яжемо нашу тривимірну модель машини.

Через те, що симулятор нічого не знає про зіткнення і тривимірний світ, не можна застосувати стандартну фізику Unity до машинок. Тому, для Rigidbody машинки було виставлено свободу руху тільки по осі Y, а обертання - тільки по осях X та Z. Координати X та Z будуть виставлятися з нашої двовимірної моделі - це координати машини на площині, а обертання по осі Y збігатиметься з обертанням

вектора LookAt створеної машинки. Також потрібно вимкнути стандартні зіткнення між машинками. Це робитьсячерезналаштування коллайдерів шарів





Тепер система управління може вільно встановлювати координати X і Z машинки, а також кут обертання по осі Y, не боячись, що в цю справу втрутиться фізика 3D движка.

Але, якщо зовсім не моделювати зіткнення між машинами - вийде дуже погано, машини будуть налазити один на одного і не взаємодіяти між собою. Ця проблема вирішується створенням тригера навколо машини та обробкою події зіткнення. Коли машинки стикаються між собою,

обчислюються вектори вздовж яких потрібно "розштовхнути" машинки і змінюються відповідні координати та швидкості. Але вони змінюються не для Rigidbody, а для вихідної моделі (тобто об'єкта Car). Скрипт обробки зіткнення виглядає так.

В результаті було розсуното машинки, доки їх тригери не перестануть стикатися.

2.3.7Донавчання нейронної мережі

При старті програми, було завантажено навчену нейронну мережу для керування машиною. Однак тут є дві невеликі проблеми:

1. Оскільки СР тільки одна, а ботів у грі буде багато, то всі боти їздитимуть абсолютно однаково. Для гри це погано та не цікаво.
2. Друга проблема в тому, що потрібно зробити в грі багато різних треків. Звичайно наша НР зможе ними їздити. Але її їзда не буде оптимальною для цього треку. Уявімо ситуацію, що НР навчилася на складному треку, приклад якого було наведено раніше. І водночас поточний трек набагато простіший (наприклад має форму кола). У такому випадку нейронна мережа буде занадто обережним і не розвиватиме максимально можливу швидкість на даному треку. Для вирішення цих двох проблем було зроблео таку річ - було донавчено нейронну мережу перед тим як запустити ботів на трек. Навчання ведеться так само як і в симуляторі – за допомогою GeneticNN. Завантажується і додається поточний трек, завантажується НР як першу особину. І потім проганяється симуляція, пройшовши пару поколінь. В результаті було отримано популяцію СР які будуть спадкоємцями вихідної зразкової СР. При цьому вони всі будуть трохи різними (за рахунок мутацій та рекомбінації). Таким чином кожному боту було надано свою унікальну СР, і всі боти будуть трохи по-різному поводитися на треку. Крім того, оскільки потрібно донавчати НР на поточному треку, то генетичний алгоритм відбере нейронні мережі найбільш пристосованих саме до цього треку. Оскільки алгоритм симуляції дуже швидкий, донавчання СР займає пару секунд і не позначається на грабельності.

2.4Результат

В цілому, машини керовані нейронною мережею відмінно поводяться на треку. На довгих ділянках вони розганяються, у поворотах – гальмують. Траєкторія руху також дуже гарна. НР воліє не входити в замет і майже весь трек вони проходять без заметів. Можливо траєкторія і не ідеальна, але вона дуже ефективна, вручну випередити ботів дуже важко. При дослідах не було можливості піднятися вище 6 місця у змаганні з ботами. Швидкість з якою їздять боти вражає

РОЗДІЛ 3. СОЦІАЛЬНА ВІДПОВІДАЛЬНІСТЬ

Спроектоване робоче місце – це офісне приміщення, в якому працюватиме інженер-програміст. У цій роботі відображено комплекс заходів організаційно-правового, технічного та режимного характеру, які мінімізують негативні наслідки розвитку інформаційної системи, а також розглядаються питання безпеки, охорони навколишнього середовища та протипожежної безпеки, даються рекомендації щодо створення оптимальної робочої умови. Специфіка і режим роботи забудовника характеризуються значним розумовим напруженням, сильним навантаженням на зоровий апарат, нерухомість. та напруга в шийному, грудному та поперековому відділах хребта, що призводить до стомлюваності, змін функціонального стану центральної нервової системи, болям у зап'ястях, ліктях, кистях, пальцях та спині. При тривалому використанні на екрані монітора з'являються біль в очах і біль голови. Розвиток інформаційної системи жодним чином не робить негативного впливу на суспільство та навколишнє середовище, але в процесі роботи з інформаційною системою фахівець може утворювати тверді відходи, такі як папір, акумулятори, лампочки, використані картриджі, харчові та особисті відходи. , канцелярські відходи, аксесуари та ін. 100

3.1 Техногенна безпека

За природою виникнення шкідливі та небезпечні виробничі фактори діляться на 4 групи:

- фізичні;
- хімічні;
- психофізіологічні;
- біологічні.

У нашому випадку біологічні та хімічні фактори істотного впливу на стан здоров'я виконавців не надають, то докладніше розглянемо лише фізичні та психофізіологічні фактори. Єдиним фактором, що відноситься до фізично

небезпечно, існує небезпека ураження електричним струмом. Наступні елементи були визначені як шкідливі виробничі фактори, які виникають при роботі з комп'ютером: До шкідливих виробничих факторів при роботі з комп'ютером належать:

1) підвищений рівень електромагнітного випромінювання, основним джерелом якого є електронно-променева трубка монітора комп'ютера; [5]

2) відхилення показників мікроклімату [3]

3) підвищений рівень шуму, джерелом якого є вентилятори всередині системного блоку і блок живлення комп'ютера, жорсткі диски і магнітні диски, люмінесцентні лампи. [5] 101

4) недостатнє освітлення робочої зони [6]

3.1.1 Рівень електромагнітних випромінювань

Вивчення людського тіла безпосередньо залежить від напруженості електричного і магнітного полів, потоку енергії, частоти коливань, а також розмірів тіла, що опромінюється. Під дією низьковольтних електромагнітних полів виникають в людини порушення оборотні. Однак якщо напруженість магнітного поля перевищує максимально допустимий рівень, страждають нервова та серцево-судинна системи, органи травлення, погіршуються деякі біологічні параметри крові. Найчастіше електромагнітне випромінювання виходить немає від екрана монітора, як від відеокабель і системний блок У портативних комп'ютерах майже все електромагнітне випромінювання надходить від системного блоку, розташованого під клавіатурою. Сучасні машини випускаються виробником зі спеціальним металевим захистом всередині системного блоку для зниження фону електромагнітного випромінювання. Згідно з [5], на відстані 50 см навколо ВДТ напруженість електричного поля електромагнітного поля повинна бути не більше: 25 В/м, якщо частота знаходиться в діапазоні 5 Гц ÷ 2 кГц, 2,5 В. / м, якщо частота знаходиться в діапазоні 2 кГц ÷ 400 кГц Щільність магнітного потоку не повинна перевищувати: 102250 нТл, якщо частота знаходиться в діапазоні 5 Гц ÷ 2 кГц 25

нТл, якщо частота знаходиться в діапазоні 2 кГц ÷ 400 кГц Можливі засоби захисту від ЕМП: Основний підхід - збільшити відстань від джерела, відео екран монітора не повинен бути ближче 50 см від користувача; Використання сітчастого фільтра, спеціального екрану, а також інших засобів індивідуального захисту, які пройшли випробування в акредитованих лабораторіях та мають відповідний гігієнічний сертифікат. 103

3.1.2 Показники мікроклімату

Давайте розберемо мікроклімат на робочому місці. Мікроклімат виробничих приміщень характеризується такими параметрами: температура, відносна вологість, швидкість повітря. Всі ці особливості впливають на організм людини як окремо, так і в цілому. Вони багато в чому визначають здоров'я. Оптимальні значення характеристик мікроклімату встановлені згідно з [3] і відображені в таблиці 5.1. За ступенем фізичної тяжкості робота інженера-програміста відноситься до легкої фізичної роботи I категорії і з енерговитратами організму до 120 Дж/с, оскільки робота виконувалася сидячи, не вимагаючи систематичних фізичних навантажень. . Таблиця 5.1 Оптимальні значення характеристик мікроклімату Період року Категорія робіт за рівнем енерговитрат, Вт Температура повітря, ◦С Температура поверхонь, ◦С Відносна вологість повітря, % Швидкість руху повітря, м/с Холодний 1а(до 139) 22-24 21-25 6 -40 0,1 Теплий 1а(до 139) 23-25 21-26 60-40 0,1 Допустимі величини показників мікроклімату встановлюються у випадках, коли за технологічними вимогами, технічними та економічно обґрунтованими причинами не можуть бути забезпечені оптимальні величини. 104 Таблиця 5.2 Допустимі значення мікроклімату робочого стола. Період року Категорія робіт Температура повітря, ◦С Температура поверхонь, ◦С Відносна вологість повітря, % Швидкість руху повітря, м/с Нижче опт Вище опт Нижче опт Вище опт Холодний 1а(до 139) 20- 21,9 24,2- 25 19-26 15-75 0,1 Теплий 1а(до 139) 21- 22,9 25,1- 28 20,29 15-75 0,1 0,2 Параметри мікроклімату приміщення, що регулюються системою центрального опалення, а також припливно- витяжною

вентиляцією мають такі значення: вологість 40%, швидкість руху повітря 0,1 м/с, температура влітку 20-25°С, взимку 15-18°С, що відповідає вимогам [3]. Якщо говорити про заходи щодо оздоровлення повітряного середовища, то у виробничому приміщенні до них належить правильна організація вентиляції та кондиціювання повітря, а також опалення приміщень. Вентиляція має здійснюватися як природним, і механічним шляхом. У робочому приміщенні необхідна подача наступного обсягу зовнішнього повітря: при обсязі приміщення до 20м3 на особу – не менше 30м3 на годину на особу; при обсязі приміщення понад 40м3 на особу 105 та відсутності виділення шкідливих речовин допускається природна вентиляція. В аудиторії примусової вентиляції немає. Але є природна, тобто. повітря надходить і віддаляється через вікна, двері, щілини. Вагомий недолік природної вентиляції в тому, що повітря надходить до приміщення без очищення та нагрівання. Природна вентиляція допускається у разі, якщо одного працюючого припадає щонайменше 40м3 всього обсягу повітря у приміщенні. Об'єм повітря на одну особу в аудиторіях КЦ — 28,88 м3, отже, потрібна наявність примусової вентиляції. У зимовий час у приміщенні має бути система опалення. Вона забезпечує достатнє, постійне та рівномірне нагрівання повітря. У приміщеннях з підвищеними вимогами до чистоти повітря слід використовувати водяне опалення. В аудиторіях використовується водяне опалення із вбудованими нагрівальними елементами та стояками. 106

3.1.3 Освітленість робочої зони

Недостатня освітленість згубно впливає на зоровий апарат, тобто знижує зорову працездатність, а освітленість робочої зони також впливає на психіку, емоційний стан людини, може викликати втому центральної нервової системи внаслідок додаткових зусиль. або сумнівні сигнали. Величезну роль для оптимізації умов праці відіграє освітлення робочих місць [6]. Організація освітлення робочих місць повинна відповідати двом вимогам: забезпечувати розрізнення об'єктів, що розглядаються, і зменшувати стрес і втому. органи зору. Виробниче світло має бути стабільним і рівномірним, мати правильну спрямованість, виключати відблиски та

утворення різких тіней. Основними якісними показниками світлового середовища є коефіцієнт пульсації освітленості (Кр). Для роботи з ПК цей показник не повинен перевищувати 5%. Оптимальна яскравість екрану 75-100 кд/м2. Завдяки такій яскравості екрану, а також яскравості поверхні столу в діапазоні від 100 до 150 кд/м2 забезпечується та зберігається працездатність візуального пристрою на рівні 80-90%. постійний розмір зіниці на допустимому рівні 3-4 мм. Місцеве освітлення не повинно створювати відблисків на поверхні екрана і не повинно збільшувати освітленість екрана ПК більш ніж на 300 люкс. Слід обмежити прямі та відбиті відблиски від будь-якого джерела світла. У лабораторії, де проводять СРС, використовується змішане освітлення, тобто поєднання природного та штучного освітлення. Через вікна проникає природне світло. 107 Використовуйте штучне освітлення, якщо природного світла недостатньо. У цьому приміщенні використовується загальне штучне освітлення. Приміщення, де проводиться САУ, освітлюється 3 лампами, в кожному з яких встановлено 4 люмінесцентні лампи типу ЛБ-40. Світильники рівномірно розподіляються на всій площі стелі, створюючи рівномірне освітлення робочих місць. Світловий потік кожної лампи у кімнаті свідчить про відповідність нормам освітлення. Необхідно обмежити відбиту яскравість від робочих поверхонь (екран, стіл, клавіатура тощо. буд.) у зв'язку з правильним вибором типів ламп і розташуванням робочих місць стосовно природним і штучним джерелам світла, яскравості відблисків на екрані ПК. не повинна перевищувати 40 кд/м2 та яскравість стель при використанні системи відбитого освітлення, не повинна перевищувати 200кд/м2 . Як джерела світла при штучному освітленні повинні застосовуватися переважно люмінесцентні лампи типу ЛБ. Загальне освітлення слід виконувати у вигляді суцільних або переривчастих ліній світильників, розташовані збоку від робочих станцій, паралельно до прямої видимості користувача при розташуванні ПК в ряд. Для освітлення приміщень з ПК слід використовувати лампи серії ЛПО36 із дзеркальними ґратами, обладнані високочастотними пусковими пристроями. Використання світильників без розсіювачів та захисних ґрат не допускається.

Яскравість світильників загального освітлення в області кутів випромінювання від 50 до 90 градусів з вертикаллю у поздовжній та поперечній площинах не повинна перевищувати 200 кд/м2, захисний кут світильників має бути не меншим за 40 градусів. 108 Місцеві освітлювальні прилади повинні мати непрозорий відбивач із кутом захисту не менше 40°.градусів. У кімнаті три віконні прорізи. ЕЕС з комбінованим освітленням і бічним природним освітленням для даного типу приміщень становить 0,7. Рівень штучного освітлення повинен бути не менше 300 люкс. [6] Таблиця 5.3 Параметри систем природного та штучного освітлення на робочих місцях Назва робочого місця Тип світильника та джерела світла Коефіцієнт природної освітленості, ЕЕС,% Освітленість при комбінованій системі, лк Фактичне та нормальне значення Фактичне та нормальне значення Робоче приміщення з 1021 люкс 300 ÷ 500 лк 109

3.1.4 Рівень шуму

Параметри, які завдають великої шкоди здоров'ю та різко знижують продуктивність, – це шум. Шум може створюватися чим завгодно, чи це робоче обладнання, системи кондиціонування, перетворювачі напруги, робочі денні ходові вогні або зовнішній шум. Дослідження показали, що шум та вібрація згубно діють на людський організм. Вплив шуму різний: він ускладнює розбірливість мови, знижує працездатність, збільшує стомлюваність, викликає зміни в органах слуху людини. Шум впливає на весь організм людини, а не лише на органи слуху. Зазначається ослаблення уваги, погіршення пам'яті, зниження реакції, збільшення кількості помилок під час роботи. Виробничі приміщення, у яких для роботи використовуються ПЕОМ, не повинні розташовуватися поблизу приміщень, де рівень шуму та вібрації перевищує нормовані значення. Допустимий рівень звукового тиску, звуку та еквівалентних рівнів звуку на робочих місцях повинні відповідати вимогам СанПіН 2.2.4.3359-16 [4]. При виконанні основної роботи на ПК рівень шуму на робочому місці не повинен перевищувати 50 дБА. 110

3.1.5 Розумова перенапруга

Організація роботи з ПК здійснюється залежно від виду та категорії робіт. Види робіт розділені на 3 групи: група А - робота з зчитування інформації з екрана ВДТ за попереднім запитом, група Б - робота з введення інформації, група Б - творча робота у діалозі з ПК. Виділяють 3 категорії складності та інтенсивності роботи з ПК за видами діяльності, що визначаються: для групи А – за загальною кількістю прочитаних знаків за зміну, але не більше 60 000 знаків за зміну; для групи Б за загальною кількістю прочитаних або введених знаків за робочу зміну, але не більше 40 000 знаків за зміну; для групи Б - за загальним часом безпосередньої роботи з ПК за робочу зміну, але не більше 6 годин за зміну. Залежно від категорії трудової діяльності та рівня навантаження за робочу зміну при роботі з ПК встановлюється загальний час регламентованих перерв. Категорія робіт за тяжкістю та напруженістю згідно ТОІ Р 45-084-01 представлена в таблиці 5. 111 Таблиця 5.4. Категорія роботи за ступенем тяжкості та напруженості за ТОІ Р 45-084-01 Категорія роботи з ПК Рівень навантаження робочої зміни за видами роботи з ПК Загальний час регламентованих перерв, хв Група А, кількість знаків Група В, кількість символів Група В, h О 8 годині зміна О 12 годині зміна III До 60 000 До 40 000 До 6 90 140 О 8 годині роботи на ПК повинні встановлюватися регламентовані перерви через 1,5-2 години на початку заняття і через 1,5- 2 години після обідньої перерви по 20 хвилин кожний або 15 хвилин кожну годину навчання. Для 12-годинного сеансу регульовані перерви повинні бути встановлені перші 8 годин. роботи аналогічно до перерв при 8-ми годинному сеансі, а протягом останні 4 години роботи, незалежно від категорії та виду роботи, щогодини по 15 хв. 112 5. 2 Безпека в надзвичайних ситуаціях. 5.2.1 Пожежна безпека приміщень Відповідно до ГОСТ Р 50571.17-2000 [8], залежно від характеристик використовуваних у виробництві речовин та їх кількості, за пожежно-вибухонебезпечністю приміщення поділяють на категорії А, Б, С, D, D. Наявність в аудиторії 204-КЦ дерев'яних виробів (столів, шаф), електропроводів 220В, а також застосування Електронагрівальні прилади з відкритими ТЕНами - паяльниками дають право

віднести приміщення за ступенем пожежо- та вибухобезпеки до категорії Б. Необхідно передбачити низку профілактичних заходів технічного плану. Можливі причини займання такі:

1) короткі замикання;

2) небезпечне навантаження мереж, що призводить до сильного нагрівання струмопровідних частин та пожежної ізоляції;

3) часто виникають пожежі під час пуску техніки після ремонту. За запобігання пожежам від коротких замикань і перевантажень необхідні для правильного вибору, монтажу та дотримання встановленого режиму роботи електричних мереж, дисплеїв та інших електротехнічних засобів автоматики. Отже, необхідно передбачити ряд профілактичних заходів технічного, оперативного, організаційного плану. 113

3.2 Можливі причини загоряння

Причиною займання може бути:

1) несправність струмовідних частин установок;

2) робота з відкритою електроапаратурою;

3) короткі замикання у блоці живлення або високовольтному блоці дисплейної розгортки;

4) недотримання правил пожежної безпеки;

5) наявність горючих компонентів: документи, двері, столи, ізоляція кабелів тощо. п.

3.2.1 Заходи щодо усунення та попередження пожеж

Щоб уникнути пожежі, дотримуйтесь наступних правил пожежної безпеки:

1) виключення утворення паливного середовища (герметизація обладнання, вентиляція, робоча та аварійна вентиляція);

2) використання вогнетривких або вогнестійких матеріалів при будівництві та

оздобленні будівель. У залі для глядачів необхідно проводити наступні протипожежні заходи:

1) організаційні заходи технічного процесу з урахуванням пожежної безпеки;

2) оперативні заходи, що враховують роботу існуючого обладнання; 3) технічні та конструктивні, пов'язані з правильним розміщенням та встановленням електрообладнання та опалювальних приладів.

Організаційні заходи:

1) протипожежний інструктаж обслуговуючого персоналу;

2) навчання персоналу правил техніки безпеки;

3) видання інструкцій, плакатів, планів евакуації.

Експлуатаційні заходи:

1) дотримання експлуатаційних норм обладнання;

2) забезпечення вільного підходу до обладнання;

3) зміст у справності ізоляції струмопровідних провідників.

Технічні заходи:

1) Дотримання протипожежних заходів при монтажі електропроводки, обладнання, систем опалення, вентиляції та освітлення. У кабінеті 204-КЦ знаходиться вуглекислотний вогнегасник типу ОУ-2, встановлений подрібнювач, який знеструмлює всю аудиторію, на дверях аудиторії наведено план евакуації на випадок пожежі, а на доступній відстані розташований протипожежний щит (2-й поверх КЦ). При виникненні спалаху в електроустановці для його усунення необхідно використовувати вуглекислотні вогнегасники типу ОУ-2. 2) Профілактичні огляд, ремонт та випробування обладнання. Окрім усунення самого осередку пожежі, потрібно своєчасно організувати евакуацію людей. 115

3.2.2 Електробезпека

У цьому розділі нас цікавить статична електрика, що виникає в результаті перерозподілу електронів та іонів при дотику двох поверхонь неоднорідних рідких або твердих речовин, на яких утворюється подвійний електричний шар. Поділ

поверхонь означає поділ зарядів цього шару, що означає, що між розділеними поверхнями існує різниця потенціалів та формується електричне поле. Статична електрика у приміщенні часто виникає, коли людина стосується компонентів комп'ютера. Виділення не становлять небезпеки для користувачів, але вони можуть призвести до проблем з комп’ютером. Щоб зменшити виникаючі заряди статичної електрики, підлогу всередині приміщень виготовляють з одношарового лінолеуму. При роботі з електроприладами дуже важливо дотримуватися техніки безпеки. Під заходами безпеки розуміється система організаційних заходів і технічних засобів, спрямованих на запобігання впливу на споживача шкідливих і небезпечних виробничих факторів. Електричні установки становлять серйозну потенційну небезпеку для користувача, що посилюється тим, що органи чуття людини не можуть визначити наявність електричної напруги на відстані. Небезпека поразки людини електричним струмом залежить від умов у приміщенні. Ризик пошкодження збільшується за наступних 116 умов: висока вологість (відносна вологість перевищує 75%), висока температура (більше 35°С), наявність струмопровідного пилу, струмопровідна підлога, а також можливість одночасного контакту з металом. елементи, з'єднані із землею, та металевий корпус електрообладнання. Тому роботи можна проводити лише у приміщеннях без підвищеного ризику, і є ризик ураження електричним струмом:

1) при дотику до струмоведучих частин, наприклад, при ремонті ПК;

2) при дотику до безструмових частин, що знаходяться під напругою (у разі порушення ізоляції струмоведучих частин ПК);

3) при зіткненні з підлогою, стінами під напругою;

4) існує небезпека короткого замикання високовольтних блоків: блоку живлення та блоку сканування дисплея. Аудиторії ЦК, в яких проводилися роботи, через небезпеку ураження електричним струмом не належать до приміщень підвищеної небезпеки. У лабораторіях використовуються прилади, які споживають напругу 220В змінного струму з частотою 50Гц. Ця напруга небезпечна для життя, тому обов'язкові такі запобіжні заходи:

1) перед початком роботи переконайтеся, що вимикачі та розетка закріплені і на них немає відкритих струмопровідних частин;

2) у разі виявлення несправності обладнання та пристроїв необхідно проінформувати відповідальне обладнання без внесення будь-яких самостійних виправлень; 117

3) заборонено захаращувати робоче місце непотрібними предметами. У разі аварії потерпілого слід негайно позбавити наслідків ураження електричним струмом і, викликавши лікаря, надати необхідну допомогу. 118

3.3 Екологічна безпека

Науково-технічний прогрес, підвищуючи здатність людини впливати на навколишнє середовище, створює умови для виникнення екологічної кризи. Водночас розвиток технологій відкриває нові шляхи збереження природного середовища та пропонує нові варіанти подолання існуючих проблем. Під навколишнім середовищем ми розуміємо сукупність природи та середовища, створеного людиною. Охорона довкілля – це складне питання, що вимагає зусиль людства. Найактивніша форма захисту навколишнього середовища навколишнього середовища від шкідливого впливу промислових викидів - це повний перехід до безвідходних та маловідходних технологій та виробництв. Це вимагатиме вирішення комплексу складних технологічних, конструкторських та організаційних завдань, заснованих на використанні нових досягнень науки та техніки.

3.3.1 Відходи

Основними видами забруднення літосфери є тверді побутові та промислові відходи. При розгляді впливу персонального комп’ютера на атмосферу, гідросферу та літосферу виявлено особливо шкідливі викиди згідно з ГОСТ Р 51768-2001

«Ресурсозбереження. Управління відходами». При виході з ладу комп’ютера їх списують і направляють у спецпоїзд, який у разі потреби вживає заходів щодо утилізації виведеного з експлуатації обладнання та комплектуючих. 119 Люмінесцентні лампи, якщо порушена цілісність корпусу виробів, які мають відслужили свій термін, виділяються пари ртуті.Лампи після цього терміну належать до здавати на спеціальні підприємства, де вони підлягають подальшій утилізації, суть якої полягає у збиранні та знешкодженні речовин, що містять ртуть. Захист ґрунту та надр від твердих відходів здійснюється шляхом збирання, сортування та поховання відходів та їх організованого поховання. 120 5.4 Організаційні заходи безпеки Під час організації робочого місця необхідно враховувати вимоги безпеки, виробничої санітарії, ергономіки, технічної естетики. Невиконання цієї вимоги може призвести до виробничої травми або професійного захворювання. Відповідно до вимог [9,10,11] в організації робота на ПК повинна відповідати таким умовам: - персональний комп’ютер (ПК), а робоче місце повинно бути розташоване так, щоб світло падало збоку, бажано ліворуч; - відстань від ПК до стін має бути не менше 1 м, тому по можливості слід уникати розташування робочого місця в кутах кімнати або звернено до стіни; - Краще встановлювати ПК так, щоб, дивлячись вгору від екрана, ви могли бачити якийсь віддалений об’єкт у приміщенні чи на вулиці. Переміщення погляду на далеку відстань є одним з найефективніших способів розвантажити зоровий апарат при роботі на ПК; - за наявності кількох комп'ютерів відстань між екраном одного монітора та задньою стінкою іншого має бути не менше 2 м, а відстань між бічними стінками сусідніх моніторів – не менше 1,2 м; - Вікна в кімнатах з ПК повинні бути обладнані регульованими пристроями (жалюзі, штори, зовнішні козирки тощо); - монітор, клавіатура та корпус комп'ютера повинні бути прямо перед оператором; висота робочого стола з клавіатурою має бути 680-800 мм над рівнем стола; а висота екрана (над підлогою) -900-1280 см; 121 - монітор повинен бути на відстані 60-70 см під кутом 20 градусів від оператора нижче рівня очей; - простір для ніг має бути: висотою не менше 600 мм, шириною не менше 500 мм, глибиною не менше 450 мм.

Повинна бути опора для ніг робітника шириною не менше 300 мм з регулюванням кута нахилу 0-20 градусів; - робоче крісло повинно мати м'яке сидіння та спинку, з регулюванням висоти сидіння, зі зручною поперековою опорою; - Положення тіла користувача відносно монітора має відповідати напрямку перегляду під прямим кутом або 75 градусів. Правильна постава і положення рук оператора дуже важливі, щоб уникнути порушень в роботі опорно-рухового апарату і виникнення синдрому постійного стресу. Згідно СанПіН 2.2.2.542-96 при 8-ми годинній робочій зміні на ВДТ та ПЕОМ перерви в роботі повинні становити від 10 до 20 хвилин кожні дві години роботи. 122

3.4 Особливості законодавчого регулювання проектних рішень

3.4.1 Спеціальні правові норми трудового законодавства

Умови праці, створені під час виконання ВКР, не є небезпечними або шкідливими для здоров'я та не несуть загрози екологічній безпеці. Графік роботи не порушувався. У лабораторії регулярно проводилися організаційно-технічні заходи, наприклад, первинний інструктаж з техніки безпеки та цільовий інструктаж із проведення робіт.

3.4.2 Організаційні заходи під час компонування робочої зони

Під час організації робочого місця необхідно враховувати вимоги безпеки, ергономіки, виробничої санітарії, технічної естетики. Невиконання цієї вимоги може призвести до виробничої травми або професійного захворювання. Організація роботи на ПК показана на рисунку 1. Рисунок 1 - Організація роботи на ПК 123 Правильна постава та положення рук оператора дуже важливі для виключення порушень опорно-рухового апарату та виникнення синдрому постійного навантаження. Відповідно до СанПіН 2.2.2.542-96 при 8-годинній зміні на ВДТ та ПК перерви в роботі мають бути від 15-20 хвилин кожні 2 години.

ВИСНОВОК ПО РОЗДІЛУ Проаналізовано фактори робочої зони з метою виявлення їх шкідливих проявів, це мікроклімат, недостатня освітленість, підвищений шум і психічне напруження. Виявлено імовірні джерела забруднення навколишнього середовища в результаті запропонованого проекту. Визначено організаційні заходи щодо забезпечення безпеки, описано основні джерела надзвичайних небезпек. Таким чином, за результатами досліджень рамках розділу «Соціальна відповідальність» було встановлено, що забезпеченими умовами праці на робочому місці попереджено та мінімізовано ризики впливу шкідливих та небезпечних факторів виробництва. Розглянуто заходи, що дозволяють такі умови забезпечити

ВИСНОВОК

Результатом даної випускної кваліфікаційної роботи є готовий програмний продукт штучний інтелект розроблений за допомогою навчання нейронної мережі У ході його реалізації було виконано:

1. Аналіз тематики проекту та існуючих додатків.
2. Під час проектування програмного додатка було створено архітектуру та методи його розробки.
3. Вибрані програмні інструменти для розробки цього проекту та двигун Unity як основа з обґрунтуванням вибору.
4. Проведено тести на наявність помилок, які були виявлені. Було проведено такі етапи тестування:

модульне тестування, інтеграційне тестування, системне тестування, альфа-тестування, бета-тестування

СПИСОК ВІКОРИСТАНОЇ ЛІТЕРАТУРИ

1.Особливості Spriter: [Електронний ресурс] // Сайт BrashMonkey // Режим доступу: https://brashmonkey.com/spriter-features;

2. Керівництво користувача Unity (2019): [Електронний ресурс] // Unity. Documentation - 2019 // Режим доступу: https://docs.unity3d.com/Manual/index.html9.

3. Об'єктно-орієнтоване програмування [Електронний ресурс.]. - Режим доступу: http://msdn.microsoft.com/ru-ru/library/dd460654.aspx.

4. Unity - Scripting [Електронний ресурс]. - Режим доступу: https://unity3d.com/learn/tutorials/modules/beginner/scripting. - Загл. з екрану.

5. Gregory J. Game Engine Architecture. – USA: A K Peters/CRC Press, 2009. – 864 p.

6. C# Programming Guide. [Електронний ресурс] URL: https://msdn.microsoft.com/en-us/library/67ef8sbd.aspx

7. Flower M. UML Distilled: A Brief Guide to the Standard Object Modeling Language. – USA: Addison-Wesley Publishing Company, 2004. – 150 p.

8. Gold J. Object-Oriented Game Development. – UK: Pearson Education Limited, 2004. – 404 p.

9. Kaner C., Bach J., Pettichord B. Lessons Learned in Software Testing. – USA: Wiley, 2001. – 320 p.

10. Tidwell J. Designing Interfaces. – USA: O’Reilly Media, 2004. – 578 p.

11. Unity3D Manual. [Електроннийресурс] URL: http://docs.unity3d.com/Manual/index.html

ДОДАТКИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра інженерії програмного забезпечення

Магістерська робота

«Вдосконалення системи руху для штучного інтелекту з використанням нейронних мереж»

Виконав: студент групи ПДМ-61 Боровик Роман Павлович

Керівник: к.т.н., доц., доцент кафедри ІПЗ Дібрівний Олесь Андрійович

Київ - 2021

МЕТА, ОБ'ЄКТА ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи: Вдосконалення системи руху для штучного інтелекту з використанням власноруч створених нейронних мереж для забезпечення високої ефективності роботи ігрових об'єктів

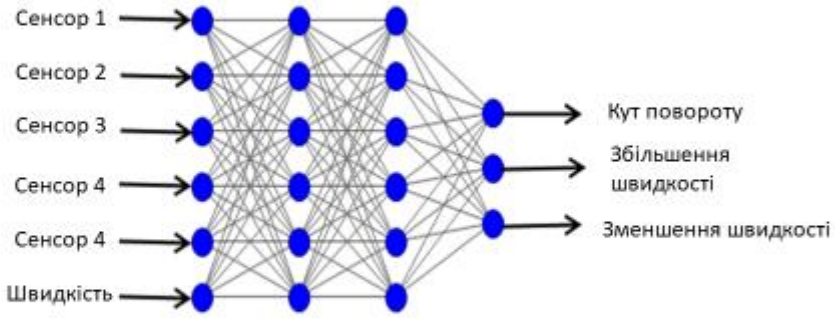
Об'єкт дослідження: процес розробки штучного інтелекту з використанням нейронних мереж створений для керування системою руху

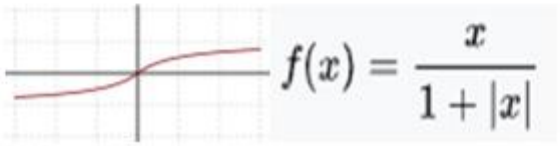
Предмет дослідження: штучний інтелект, що розроблений з використанням нейронних мереж для ігор в жанрі аркади, розробка оптимальної топології для навчання нейронної мережі за найменшу кількість ітерацій.

АКТУАЛЬНІСТЬ РОБОТИ

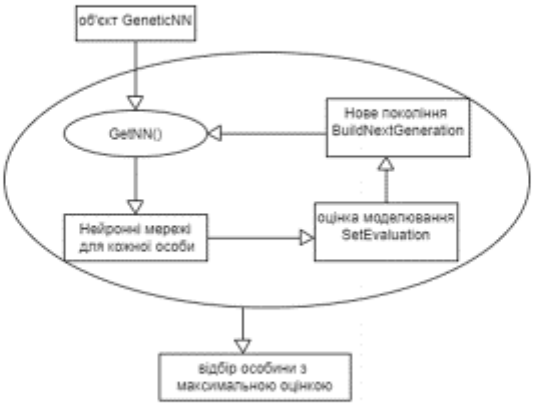
Актуальність роботи полягає в підвищенні ефективності штучного інтелекту для систем керування об'єктами ігор. Актуальність розробок в галузі нейронних мереж обумовлена перш за все тим, що застосування даної моделі широко використовуються в найрізноманітніших областях. За допомогою вирішення задач на основі нейронних мереж функціонування будь-якої системи стає ефективнішим.

НЕЙРОННА МЕРЕЖА

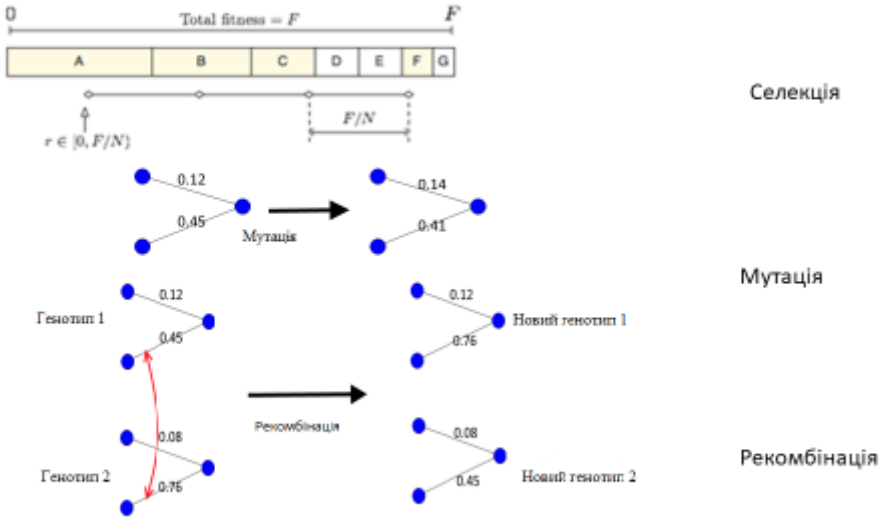




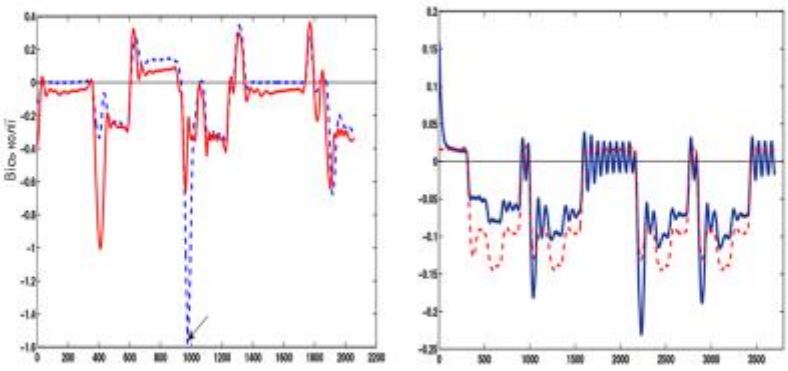
Алгоритм навчання нейронної мережі



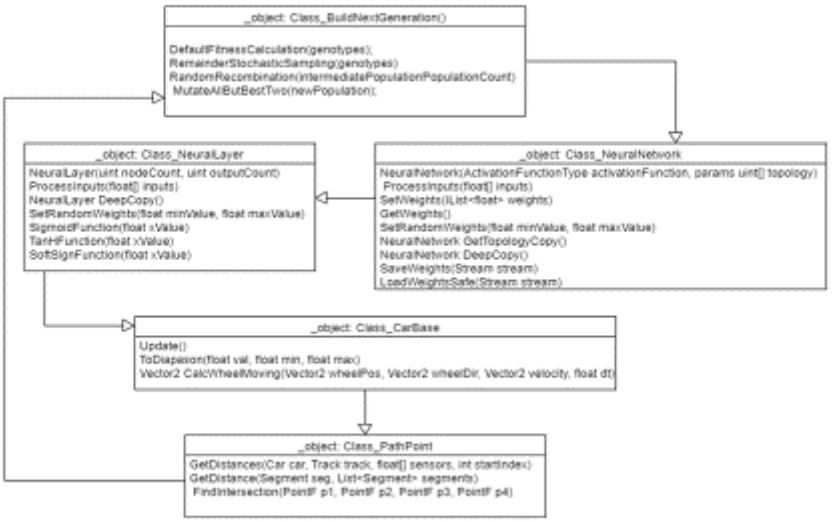
ФОРМУВАННЯ НОВИХ ПОКОЛІНЬ НЕЙРОННОЇ МЕРЕЖІ



РЕЗУЛЬТАТИ МОДЕЛЮВАННЯ



ПРАКТИЧНИЙ РЕЗУЛЬТАТ



ВИСНОВКИ

1. Аналіз тематики проекту та існуючих додатків.
2. Було створено архітектуру та методи його розробки.
3. Вибрані програмні інструменти для розробки цього проекту та рушій Unity як основа з обгрунтуванням вибору.
4. Проведено тести, модульне тестування, інтеграційне тестування

АПРОБАЦІЯ РОБОТИ

Статті:

Боровик Р.П. Використання ECS методології при розробці в юніті, Електронний науковий журнал «Приазовський вісник» робота ще не опублікована, але подана до друку

Тези доповідей:

ДЯКУЮ ЗА УВАГУ!