ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ АВТОМАТИЗОВАНИХ СИСТЕМ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «ДОСЛІДЖЕННЯ ІНТЕГРАЦІЇ ТЕХНОЛОГІЙ ІЗ ПІДТРИМКОЮ IoT ТА ШТУЧНОГО ІНТЕЛЕКТУ В SMART CITY»

на здобуття освітнього ступеня магістра
зі спеціальності 126 Інформаційні системи та технології
освітньо-професійної програми Інформаційні системи та технології

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ *Віталій ІВКО*

Виконав:
здобувач вищої освіти
група ІСДМ-63 Віталій ІВКО

Керівник:
науковий ступінь, вчене звання Ольга ПОЛОНЕВИЧ
к.т.н., доцент

Рецензент:
науковий ступінь, вчене звання _____
Ім’я, ПРІЗВИЩЕ

Київ – 2023

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут Інформаційних технологій

Кафедра Інженерії програмного забезпечення автоматизованих систем

Ступінь вищої освіти Магістр

Спеціальність Інформаційні системи та технології

Освітньо-професійна програма Інформаційні системи та технології

ЗАТВЕРДЖУЮ
Завідувач кафедрою ІПЗАС

_____Каміла СТОРЧАК
«____» _____20___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ СТУДЕНТУ
Івку Віталію Володимировичу
(прізвище, ім'я, по батькові здобувача)

1.Тема кваліфікаційної роботи: «Дослідження інтеграції технологій із підтримкою IoT та штучного інтелекту в Smart City»

керівник кваліфікаційної роботи Ольга ПОЛОНЕВИЧ, к.т.н., доцент
(ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від «19» жовтня 2023 року № 145.

2. Строк подання кваліфікаційної роботи: 29 грудня 2023 року.

3. Вихідні дані до кваліфікаційної роботи: Методи http GET, POST, PUT, PATCH, DELETE;
Вимоги до бездротових мереж;
Науково-технічна література з питань, пов'язаних з технологіями IoT та AI.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. IoT «розумні» міста.

2. Аналіз технологій штучного інтелекту для Smart City.
3. Інтеграція технологій із підтримкою IoT та штучного інтелекту у Smart City.

5. Перелік ілюстративного матеріалу: *презентація*
   1. Цифрові технології для функціонування «розумних» міст.
   2. Порівняння мережних технологій IoT для Smart City.
   3. Технології штучного інтелекту для Smart City.
   4. Інтеграція IoT-технологій та штучного інтелекту у Smart City.

6. Дата видачі завдання: 19 жовтня 2023 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз наявної науково-технічної літератури	19.10 – 25.10.23	
2	Аналіз IoT-технологій	26.10 – 02.11.23	
3	Аналіз мережних протоколів IoT для «розумних» міст	03.11 – 09.11.23	
4	Використання бібліотеки NumPy	10.11 – 16.11.23	
5	Дослідження технологій штучного інтелекту, які інтегруються у Smart City	17.11 – 21.11.23	
6	Використання штучного інтелекту для системи «розумного» будинку	22.11 – 11.12.23	
7	Оформлення роботи: вступ, висновки, реферат	12.12 – 20.12.23	
8	Розробка демонстраційних матеріалів	21.12 – 28.12.23	

Здобувач вищої освіти _____ Віталій ІВКО
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник роботи
кваліфікаційної роботи _____ Ольга ПОЛОНЕВИЧ
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня магістра: 85 стор., 50 рис., 4 табл., 21 джерело.

Мета роботи – дослідити технології та протоколи Інтернету речей та технології штучного інтелекту, які інтегруються у Smart City.

Об'єкт дослідження – процес інтеграції технологій IoT та AI.

Предмет дослідження – технології IoT та AI у Smart City.

Короткий зміст роботи: У роботі проведено дослідження IoT-технологій, які інтегруються у Smart City. Проведено дослідження технологій штучного інтелекту, які інтегруються у Smart City. Визначені типи автоматизацій для взаємодії системи Smart Home, який є частиною концепції Smart City, та Artificial Intelligence. Проаналізовані алгоритми машинного навчання для Інтернету речей та застосування засобів штучного інтелекту для автоматизації системи «розумного» будинку. Визначені інструменти для інтеграції ресурсів Smart Home та Artificial Intelligence в єдину систему для вдосконалення функціональності та ефективності «розумних» будинків.

КЛЮЧОВІ СЛОВА: ТЕХНОЛОГІЇ, ІНФОРМАЦІЙНА СИСТЕМА, ІНТЕРНЕТ РЕЧЕЙ, ПРОТОКОЛ, ШТУЧНИЙ ІНТЕЛЕКТ, АЛГОРИТМ, СОКЕТ, SMART CITY, ІНТЕРАКТИВНЕ СЕРЕДОВИЩЕ.

ABSTRACT

Text part of the master`s qualification work: 85 pages, 50 pictures, 4 table, 21 sources.

The purpose of the work is to investigate the technologies and protocols of the Internet of Things and artificial intelligence technologies that are integrated into the Smart City.

Object of research is the process of integration of IoT and AI technologies.

Subject of research is IoT and AI technologies in Smart City.

Summary of the work: In the work, a study of IoT technologies, which are integrated into Smart City, was carried out. Research was conducted on artificial intelligence technologies that are integrated into the Smart City. The types of automation for the interaction of the Smart Home system, which is part of the Smart City concept, and Artificial Intelligence were determined. Machine learning algorithms for the Internet of Things and the use of artificial intelligence for the automation of the Smart Home system were analyzed. Tools were defined for the integration of Smart Home and Artificial Intelligence resources into a single system to improve the functionality and efficiency of Smart Homes.

KEYWORDS: TECHNOLOGY, INFORMATION SYSTEM, INTERNET OF THINGS, PROTOCOL, ARTIFICIAL INTELLIGENCE, ALGORITHM, SOCKET, SMART CITY, INTERACTIVE ENVIRONMENT.

ЗМІСТ

ВСТУП

Актуальність теми: Побудова Smart City включає в себе використання різноманітних технологій для оптимізації міських систем та поліпшення якості життя мешканців. Ці технології повинні працювати спільно для створення інтелектуального та ефективного міста, яке відповідатиме потребам мешканців та надавати їм зручність, оптимізуватиме використання ресурсів та сприятиме сталому розвитку. Побудова Smart City вимагає комплексного підходу до інтеграції цих технологій для створення міст, які у майбутньому будуть не лише технологічно передовими, але й забезпечуватимуть підвищену якості життя громадян.

Метою роботи є дослідження технологій та протоколів Інтернету речей та технології штучного інтелекту, які інтегруються у Smart City.

Для досягнення поставленої мети необхідно *виконати наступні завдання*:

- дослідити технології та протоколи Інтернету речей для «розумного» міста;
- дослідити технології штучного інтелекту для «розумного» міста;
- визначити алгоритми машинного навчання для Інтернету речей для застосування в системах «розумного» будинку та «розумного» міста;
- визначити методи інтеграції IoT-технологій та технологій штучного інтелекту у Smart City;
- розробити рекомендації застосування засобів штучного інтелекту для автоматизації системи «розумного» будинку.

Об'єкт дослідження – процес інтеграції технологій IoT та AI.

Предметом дослідження є технології IoT та AI у Smart City.

Апробація результатів магістерської роботи:

Івко В.В. «Інтеграція технологій штучного інтелекту в Smart City». Тези доповіді на І Всеукраїнській науково-технічній конференції «Технологічні горизонти: дослідження та застосування інформаційних технологій для технологічного прогресу України і Світу». – Київ, 28 листопада 2023 року.

РОЗДІЛ 1. IoT «РОЗУМНІ» МІСТА

1.1 Аналіз IoT-технологій

Інтернет речей (IoT) використовує різноманітні технології для з'єднання пристроїв та забезпечення обміну даними між ними. Ключові технології, які використовуються в IoT:

1) Бездротові технології зв'язку:

- *Wi-Fi*: дозволяє підключати пристрої до Інтернету через бездротовий зв'язок;
- *Bluetooth:* використовується для короткодіючого бездротового з'єднання між пристроями;
- *Zigbee та Z-Wave*: протоколи для низькоенергетичних, бездротових мереж для IoT-пристроїв у домашньому оточенні.

2) *RFID (Radio-Frequency Identification):* забезпечує безконтактний обмін даними між пристроями за допомогою радіочастот.

3) *Сенсори та актуатори:*

- акселерометри, гіроскопи, сенсори температури та вологості: використовуються для збору різних типів даних про навколишнє середовище;
- камери та мікрофони: використовуються для збору візуальної та аудіоінформації.

4) *M2M (Machine-to-Machine) Communication:* технологія, що дозволяє пристроям взаємодіяти між собою без прямого втручання людей.

5) *Cloud Computing:* дозволяє зберігати та обробляти великі обсяги даних з IoT-пристроїв.

6) *Edge Computing:* обробка даних на рівні пристрою або локально, що дозволяє зменшити затримки та обсяг даних, які передаються.

7) *Blockchain:* забезпечує безпеку та конфіденційність даних у мережі IoT.

8) *LPWAN (Low Power Wide Area Network):* мережі з низьким споживанням енергії для підключення великої кількості пристроїв на великі відстані.

9) AI та машинне навчання: використовуються для аналізу вивчення великих обсягів даних, які отримані від IoT-пристроїв.

Ці технології разом створюють потужне інфраструктурне середовище для функціонування та розвитку різноманітних застосувань Інтернету речей (рис.1.1).

Рисунок 1.1 – IoT у «розумних» містах

1.2 Компоненти Smart City

«Розумне» місто складається з кількох компонентів, які показано на рис.1.2. Програми розумного міста зазвичай мають чотири аспекти, пов’язані з ними: перший – це збір даних, другий – передавання та прийом даних, третій – зберігання, четвертий – аналіз. Збір даних залежить від програми та є справжнім двигуном для розробки датчиків у різних областях.

Друга частина — це обмін даними, який передбачає передавання даних із блоків збору даних у хмару для зберігання та аналізу. Це завдання було досягнуто декількома способами: багато підприємств розумного міста мають загальноміські мережі Wi-Fi, використовуються технології 4G і 5G, а також різні типи локальних мереж, які можуть передавати дані як на локальному, так і на глобальному рівні. Третій етап — це зберігання в хмарі.

Для впорядкування та організації даних використовуються різні схеми зберігання, щоб зробити їх придатними для четвертого етапу — аналізу даних. Аналіз даних означає вилучення шаблонів і висновків із зібраних даних для

прийняття рішень. У деяких ситуаціях простий аналіз, як от базове прийняття рішень і агрегування, також може працювати. Для більш складного прийняття рішень доступність хмари дозволяє не тільки збирати, зберігати та обробляти гетерогенні дані, але й аналізувати дані за допомогою статистичних методів, а також машинних алгоритмів і алгоритмів глибокого навчання у режимі реального часу.

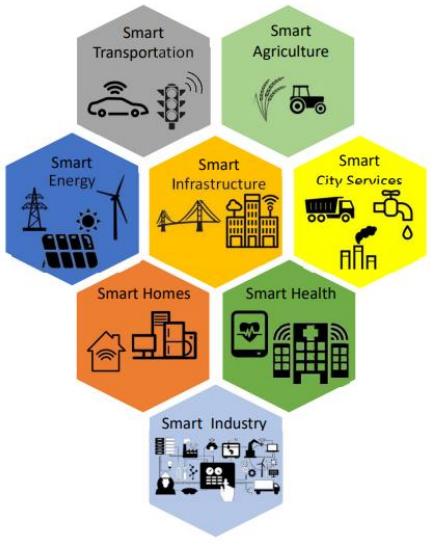


Рисунок 1.2 - Компоненти Smart City

«Розумне» сільське господарство

Продовольча безпека є однією з найважливіших частин цілей сталого розвитку Організації Об'єднаних Націй (ООН) на 2030 рік. Із збільшенням населення світу, погіршенням кліматичних змін, що спричиняє нестабільну погоду у продовольчих центрах світу, змагання за те, щоб виробництво їжі стало стабільним і ефективне використання ресурсів, що скорочуються, таких як вода, є пріоритетом для країн у всьому світі. «*Розумне» сільське господарство* – це використання датчиків, вбудованих у рослини та поля, для вимірювання різних параметрів, які допомагають приймати рішення та запобігати хворобам, шкідникам тощо. Частиною парадигми розумного сільського господарства є точне землеробство, яке передбачає розміщення датчиків у рослинах для забезпечення цільових вимірювань і таким чином дозволяють розгортати механізми цільової допомоги.

Точне землеробство буде необхідним для продовольчої безпеки в майбутньому і, отже, є важливою частиною боротьби за стале виробництво продуктів харчування. Основними застосуваннями штучного інтелекту в IoT для сільського господарства є моніторинг посівів, виявлення захворювань і догляд за посівами на основі даних і прийняття рішень.

Сервіси Smart City

Послуги «розумного» міста охоплюють діяльність, яка підтримує населення міста, це стосується муніципальних завдань, таких як постачання води, утилізація відходів, екологічний контроль, моніторинг тощо. Датчики якості води можуть бути розгорнуті, щоб постійно надавати оновлену інформацію про якість води, що використовується в місті та виявлення витоків. Одним із популярних компонентів ініціатив «розумного» міста є управління відходами, і воно було частиною багатьох ініціатив «розумного» міста від жолобів у Барселоні до баків, обладнаних датчиками та підключених до хмари, щоб не лише інформувати відповідні органи влади про необхідність їх спустошити, а також використовувати штучний інтелект для визначення найкращого маршруту для зниження витрат. Датчики також можна використовувати для моніторингу навколишнього середовища в місті, щоб визначити рівень забруднення і направляти громадян до наступного вільного місця для паркування, щоб заощадити на паливі.

«Розумна» енергія

Типові електричні системи мають односторонній потік енергії від мережі. Джерело генератора, як правило, гідроелектростанція або електростанція на основі викопного палива. Виробництво електроенергії контролюється за допомогою зворотного зв’язку від підстанцій, однак, оскільки немає інформаційного зворотного зв’язку від споживача, схема виробництва електроенергії, яка використовується в цих системах, вимагає, щоб потужність, вироблена цими джерелами, значно перевищувала попит, щоб забезпечити безперервну роботу постачання електроенергії. Процес виявлення несправностей і

виконання коригувальних дій у таких системах також займає багато часу. Крім того, оскільки технології відновлюваних джерел енергії стають дешевшими, сучасний споживач не тільки отримує електроенергію від основної комунальної компанії, але й здійснює власну генерацію. «Розумні» електромережі – це використання інформаційно-комунікаційних технологій (ІКТ) для того, щоб зробити поточні та нові встановлені мережі більш видимими, дозволити розподілене виробництво енергії як на стороні споживача, так і на стороні утиліти та запровадити можливість самовідновлення мережі. Одним із аспектів інтелектуальних мереж є те, що дані про електроенергію в режимі реального часу передаються комунальним підприємствам у різних точках мережі по всіх лініях постачання до споживача. Оскільки інтелектуальні мережі надають дані про використання споживачами у реальному часі, це дозволяє краще управляти виробництвом електроенергії за допомогою моделей прогнозування, розроблених на основі отриманих даних про споживання, інтегруючи різні джерела енергії, а також самовідновлення мережі для забезпечення безперебійного постачання (рис.1.3).

Рисунок 1.3 - Smart Energy System

«Розумне» здоров'я

Розумне здоров’я стосується використання ІКТ для покращення доступності та якості медичної допомоги. У зв’язку із збільшенням населення та зростанням

вартості медичної допомоги, ця сфера привертає увагу дослідників, а також постачальників медичних послуг. Сучасні системи охорони здоров'я переобтяжені, тому не можуть задовольнити зростаючий попит населення. У зв'язку з цим «розумне» здоров'я має на меті забезпечити доступність медичної допомоги якомога більшій кількості людей через телемедичні послуги та покращену діагностичну допомогу лікарям за допомогою штучного інтелекту. Завдяки повсюдному поширенню мобільних телефонів і пристроїв відстеження стану здоров'я, які можуть фіксувати дані про здоров'я людей у режимі реального часу (ЕКГ, температура, насичення тіла киснем та інші біосенсори), а також записувати щоденну активність і виявляти аномальні рухи за допомогою інерційних датчиків, стало можливим використовувати хмарні можливості для обробки цих даних для прийняття кращих рішень щодо охорони здоров'я. Таким чином зменшуються загальні витрати, а також навантаження на заклади охорони здоров'я (рис.1.4).

Рисунок 1.4 – IoT у «розумній» охороні здоров'я

«Розумний» будинок

Одним із основних компонентів «розумних» міст є «розумний» дім, оскільки він займає центральне місце в житті мешканців міста. «Розумні» будинки передбачають використання датчиків, встановлених по всьому будинку людини,

які надають інформацію про будинок, а також про його мешканців. Ці датчики можуть включати монітори активності користувачів, наприклад, датчики навколишнього середовища, засоби відстеження руху та споживання електроенергії.

Комбінована система управління «розумним» будинком приведена на рис.1.5.

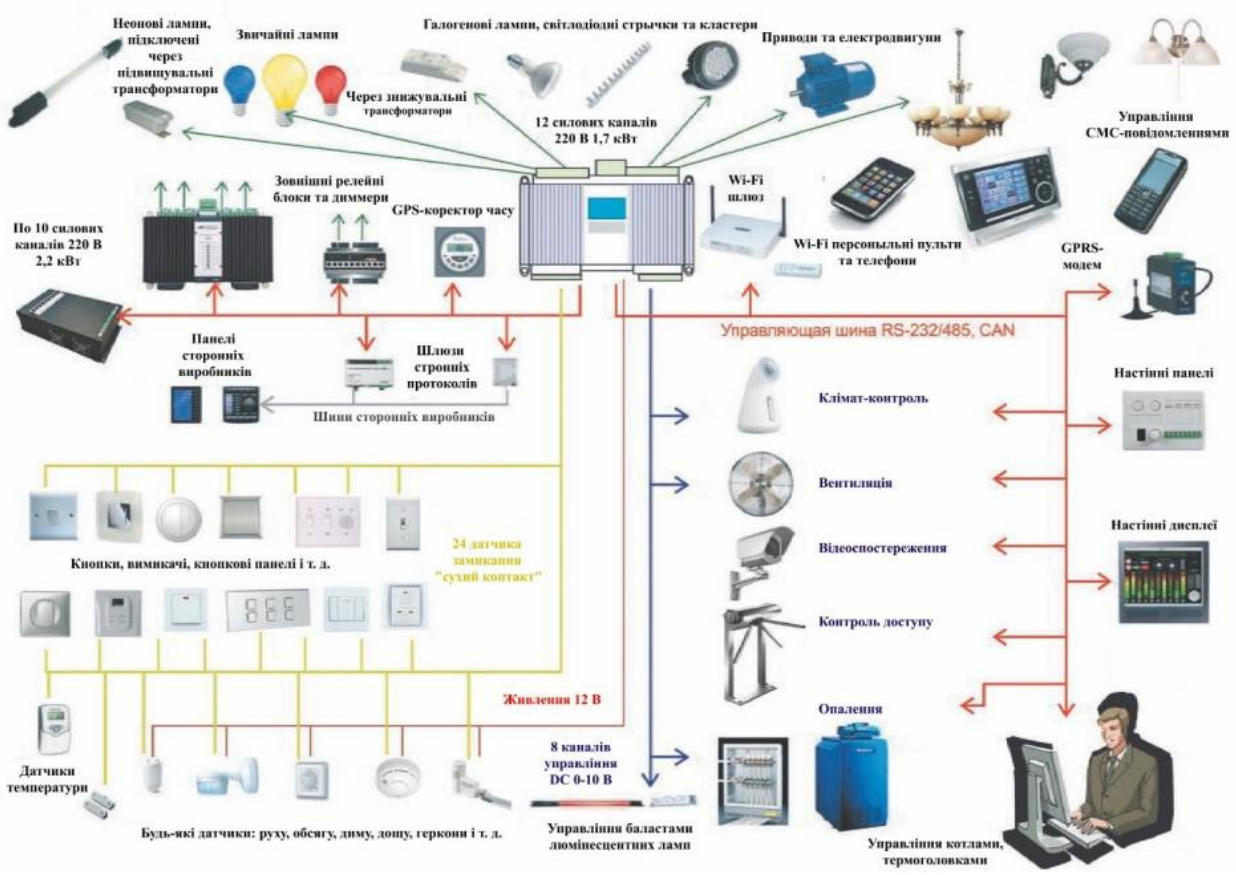


Рисунок 1.5 – Комбінована система управління «розумним» будинком

«Розумна» промисловість

Промисловості в усьому світі зайняті постійним прагненням бути більш ефективними та підвищувати продуктивність при зниженні витрат. Парадигма індустрії 4.0 передбачає бачення пов’язаної фабрики, де всі її посередники бездоганно інтегровані, працюючи в тандемі один з одним. Це стало можливим завдяки технологіям IoT. Використання IoT у виробничих і виробничих процесах, кіберфізичних системах, що об’єднують працівників і машини, призвело до ряду переваг для галузі, таких як швидші та кращі інновації, оптимізація схем

виробництва (ресурсів і процесів), краща якість продукції та підвищена безпека для робітників заводу. Однак інтелектуальні індустрії стикаються з кількома проблемами щодо використання IoT, робота з набором різнорідних пристроїв і машин має свої проблеми та вимагає від кіберфізичних систем гнучкості конфігурації, підключення та швидкого впровадження для використання у додатках IoT для Smart Industry. Штучний інтелект тісно взаємодіє з технологіями IoT для того, щоб стимулювати розробку та розгортання послуг промисловості 4.0. Завдяки датчикам, вбудованим у машини та інші процеси на заводі, дані з цих джерел дають можливість використовувати методи штучного інтелекту для підвищення рівня автоматизації, виконання операцій бізнес-аналітики тощо. Фактично, дослідники запропонували структуру для інтеграції ШІ в IoT для «розумної» промисловості. Основні застосування штучного інтелекту в промисловості – це прогнозне технічне обслуговування, моніторинг та виявлення несправностей (справність машини) і управління виробництвом.

«Розумна» інфраструктура

Інфраструктура міста має вагоме значення для його якості життя, міська влада повинна будувати нові мости, дороги та будівлі для мешканців, а також виконувати технічне обслуговування для безперебійного використання. «Розумна» інфраструктура допомагає містам гарантувати, що їхня інфраструктура знаходиться у формі та придатна для використання, використовуючи датчики для вимірювання структурного стану будівель, мостів для моніторингу стану конструкції за допомогою акселерометрів та «розумних» матеріалів. Дані, які зібрані за допомогою цих датчиків, дозволяють проводити прогнозне технічне обслуговування цих важливих одиниць для підтримки нормальної роботи міста.

«Розумний» транспорт

Багато міських центрів страждають від проблем із дорожнім рухом, зокрема, із заторами, забрудненням, розкладом. Стрімкий розвиток і впровадження нових ІКТ, транспортної інфраструктури та пішохідного зв'язку стали звичним явищем.

Незалежно від того, чи то транспортний засіб – транспортний засіб (V2V), транспортний засіб – інфраструктура (V2I), транспортний засіб – пішохід (V2P) або пішохід – інфраструктура (P2I), такі технології зробили можливим проєктування «розумних» транспортних систем. Багато підходів використовують дані GPS для відстеження поведінки водія та моделей дорожнього руху. Ці дані у реальному часі вже використовуються для картографування маршрутів у таких програмах, як Waze і Google Maps, а також для планування поїздок у громадському транспорті. Паркувальні системи, обладнані датчиками, також можуть направляти водіїв до найближчого вільного місця для паркування.

1.3 Застосування IoT для «розумних» міст. Архітектури IoT для «розумних» міст

В основі ініціатив «розумного» міста є Інтернет речей - технологічна технологія, яка дозволила всеосяжну цифровізацію й породила концепцію «розумних» міст. Інтернет речей забезпечує повсюдне підключення пристроїв до мережі Інтернет, що дозволяє їм надсилати інформацію у хмару та потенційно отримувати вказівки для виконання дій. IoT передбачає збір даних і виконання операцій аналізу даних для отримання інформації з метою підтримки прийняття рішень і розробки політики. За оцінками до 2025 року понад 75 мільярдів пристроїв буде підключено до Інтернету, що сприятиме ще більшій розробці програм. У контексті «розумного» міста IoT дозволяє датчикам збирати та надсилати дані про стан міста в центральну хмару з подальшим витягуванням та обробкою даних для вилучення шаблонів і прийняття рішень.

Інтернет речей об’єднує операції приймання, передавання, обробки та зберігання даних за допомогою хмарних сервісів. Загальна архітектура IoT складається з п’яти рівнів, де послідовні рівні працюють з інформацією з попереднього рівня, як показано на рис.1.6. Також показані три архітектури, для IoT-систем.

Сенсорний рівень складається з датчиків, які можуть отримувати інформацію про фізичні величини, які можуть впливати на фізичні об’єкти, наприклад,

пристрої радіочастотної ідентифікації (RFID). Дані, зчитані сенсорним рівнем, передаються далі на проміжний рівень за допомогою мережного рівня за допомогою бездротових мережних технологій, таких як Wi-Fi, стільниковий Інтернет, Zigbee, Bluetooth. Проміжний рівень забезпечує загальний інтерфейс для апаратного забезпечення сенсорного та прикладного рівень, який використовує дані через різні API та служби управління базами даних, щоб надавати користувачам послуги. Бізнес-рівень приєднується до прикладного рівня та використовується для розробки стратегій і формулювання політик, які допомагають управляти всією системою.

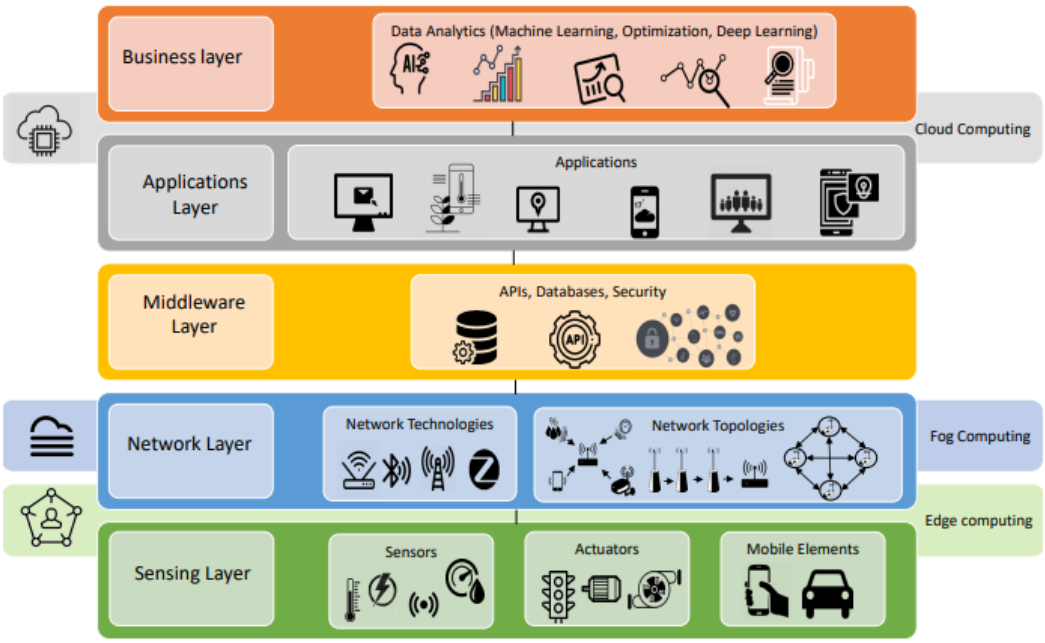


Рисунок 1.6 – Архітектури для IoT-систем

З точки зору архітектур, як правило, архітектури IoT класифікуються на основі операційних завдань, розподілених на частини системи IoT, ця категоризація базується головним чином на обробці даних. Існують три архітектури систем IoT - це хмарна, туманна та гранична моделі. Важливо зазначити, що три архітектури IoT не є взаємовиключними, натомість метою цієї ієрархії є доповнення вищого рівня, надаючи йому лише корисну інформацію, яка робить систему більш продуктивною та надійною. Для будь-якого розробника IoT-системи мета полягає

в тому, щоб встановити баланс між можливостями трьох рівнів, зберігаючи в полі зору вартість системи та вимоги.

1.3.1 Cloud Computing Model в системі Інтернету речей

Cloud Computing Model - архітектура для систем IoT, яка базується на передумові, що обробка даних із різних компонентів системи IoT має відбуватися в хмарі (рис.1.7). Хмарні обчислення дозволяють здійснювати віддалений доступ до безперервних загальних ресурсів (обчислень, сховищ і послуг) через мережу. Він повинен мати можливість динамічно розподіляти ці ресурси без втручання людини, планувати або об'єднувати за потреби, а також мати можливість доступу з різних платформ. Хмара може надавати як апаратні, так і програмні послуги для програм «розумного» міста. Перевага цього методу полягає в тому, що він забезпечує центральну платформу управління, з якої можна спостерігати, контролювати систему IoT, а також розповсюджувати командні дії на основі отриманих даних. Крім того, ця централізація дозволяє хмарним системам мати достатньо великі обчислювальні сховища, що дозволяє їм виконувати складні завдання інтелектуального аналізу даних, вилучення шаблонів і робити висновки з даних датчиків у «розумних» містах, щоб використовувати їх найкращим чином.

Але у використання моделі хмарних обчислень для IoT є й недоліки. По-перше, передавання всіх зібраних даних у хмару збільшує мережний трафік, навіть якщо це може бути невірно для програм, у яких вимірювання проводяться не дуже часто, але в інших випадках це може збільшити мережні витрати. Крім того, накладні витрати на передавання даних можуть зрости через велику кількість даних, які необхідно передати багатьом датчикам, які існують у сценарії розумного міста. Іншим недоліком, від якого страждає модель хмарних обчислень, є затримка даних, оскільки сенсорні блоки існують на сенсорному рівні, а прийняття рішень з обробки даних відбувається в хмарі, що призводить до затримки даних під час передавання сенсорної інформації, особливо коли багато пристроїв починають надсилати дані одночасно. Надійність мережі може бути

проблемою під час використання цієї моделі, оскільки через великий обсяг трафіку даних у мережі може бути неможливо застосувати надійні схеми передавання даних, оскільки системи IoT стають більшими.

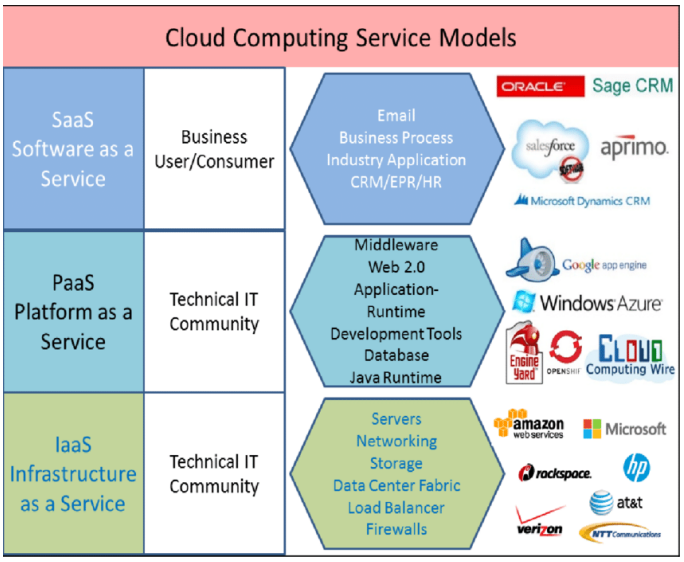


Рисунок 1.7 - Cloud Computing Service Model

Cloud Computing Model в системі Інтернету речей (IoT) визначає спосіб, яким обробляються, зберігаються та обмінюються даними, зібрані від пристроїв IoT. Цей підхід використовує хмарні технології для надання послуг та ресурсів, необхідних для забезпечення функціональності та масштабованості системи IoT. Основні складові Cloud Computing Model в IoT-системі включають:

1) Хмарні сервіси (Cloud Services). Включають в себе різноманітні послуги, такі як обчислення, зберігання даних, бази даних, аналітику, машинне навчання тощо, які надаються через хмарний інфраструктурний сервіс.

2) Хмарна інфраструктура (Cloud Infrastructure) – це апаратне та програмне забезпечення, яке забезпечує можливість розгортання та використання хмарних послуг.

3) Зберігання даних у хмарі (Cloud Storage). Дані, які зібрані від пристроїв IoT, можуть бути збережені у хмарі для подальшої обробки та аналізу.

4) Обчислення у хмарі (Cloud Computing). Обчислювальні ресурси у хмарі можуть бути використані для виконання різноманітних обчислювальних завдань,

таких як аналіз даних, обробка сигналів, виконання алгоритмів машинного навчання тощо.

5) Хмарний доступ (Cloud-based Access) – віддалений доступ до хмарних ресурсів та послуг, що дозволяє взаємодіяти з даними та пристроями IoT з будь-якого місця за допомогою мережі Інтернет.

Цей підхід сприяє ефективному використанню ресурсів, масштабованості, забезпеченню безпеки та доступності даних в IoT-системах. Але важливо також враховувати питання конфіденційності та безпеки даних, оскільки вони передаються та зберігаються у хмарних сервісах.

1.3.2 Fog Computing Model у системі IoT

Fog Computing Model в системі Інтернету речей (IoT) є альтернативним підходом до обробки даних, який використовує розподілені ресурси на рівні мережі, щоб зменшити завантаження на централізовані хмарні інфраструктури (рис.1.8). У Fog Computing обчислення та обробка даних відбуваються ближче до місця збору даних, а не в центральному хмарному середовищі. Проаналізуємо Основні характеристики Fog Computing Model.

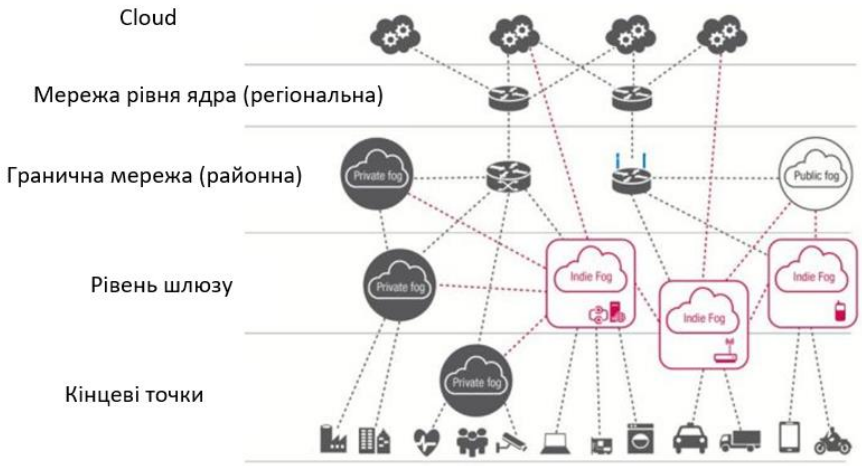


Рисунок 1.8 – Fog IoT

1) Розподілені ресурси. Fog Computing використовує ресурси, розташовані на мережному рівні, такі як мережні вузли, шлюзи та інші проміжні пристрої для

обробки та аналізу даних. Це дозволяє збільшити швидкість обробки та зменшити затримки у передаванні даних до центрального хмарного середовища.

2) Низька затримка та велика пропускна спроможність. Fog Computing дозволяє реагувати на події в реальному часі, оскільки обробка відбувається близько до джерела даних. Це особливо важливо в застосунках, де вимагається миттєва відповідь, наприклад, в системах моніторингу та управління.

3) Ефективне використання мережних ресурсів. Замість відсилання всіх даних на центральний сервер для обробки, Fog Computing дозволяє виконувати обчислення на локальному рівні. Це зменшує трафік в мережі та допомагає зекономити пропускну спроможність.

4) Гнучкість та масштабованість. Fog Computing може бути гнучко налаштованим та легко масштабованим в залежності від потреб конкретної систпеми IoT. Наприклад, можливо використовувати розподілені ресурси відповідно до географічного розташування пристроїв IoT.

Цей підхід може бути особливо ефективним в застосунках, де важлива реакція в реальному часі та обмежена пропускна спроможність мережі, наприклад, у містах, промислових об'єктах або автономних транспортних системах.

1.3.3 Edge Computing Model в IoT-системах

Edge Computing Model в системі Інтернету речей (IoT) - це підхід до обробки та аналізу даних, при якому ці процеси відбуваються ближче до самого джерела даних, а не в централізованій хмарі або на віддалених серверах. Edge Computing розміщує обчислення та обробку даних на "краю" (edge) мережі, неподалік від пристроїв IoT або вузлів мережі (рис.1.9). Основні характеристики Edge Computing Model включають:

1) Локальну обробку даних. У Edge Computing обчислення відбуваються безпосередньо на пристроях IoT або на близьких до них ресурсах, які називаються граничними вузлами (edge nodes). Це дозволяє значно зменшити затримки у відповіді та обробляти дані на місці їх виникнення.

2) Зниження трафіку в мережі. Edge Computing дозволяє виконувати обробку даних локально, уникнувши передавання великої кількості даних до централізованого хмарного сховища. Це може бути ефективним у випадках, коли об’єми даних великі або коли потрібно забезпечити швидку відповідь на події.

3) Забезпечення приватності та безпеки. Edge Computing може допомогти зберегти дані на локальному рівні, що сприяє збереженню приватності та зменшенню ризиків витоку чутливої інформації, оскільки менше даних потрапляє на центральні сервери або у хмару.

4) Робота в умовах обмеженого зв’язку. Edge Computing може бути корисним у випадках, коли мережне з’єднання нестабільне або обмежене, оскільки обробка відбувається локально, а не віддалено.

5) Підвищена масштабованість. Edge Computing може бути масштабованим в залежності від розміру та потреб визначеної системи IoT.

Цей підхід особливо корисний у випадках, де важлива реакція в реальному часі, обмежена пропускна спроможність мережі або коли важливо зберігати та обробляти дані локально з огляду на приватність та безпеку.

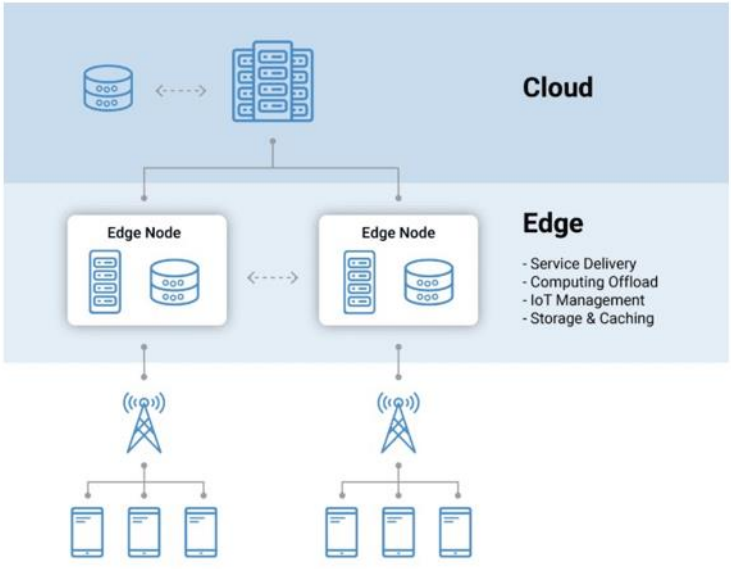


Рисунок 1.9 - Edge Computing Model в системі Інтернету речей

1.4 Мережні технології IoT для «розумних» міст

Інтернет речей у «розумних» містах залежить від агрегації даних, виміряних окремими датчиками, розміщеними в середовищі «розумного» міста. Системи, які можуть використовувати ці вимірювання окремо, давно існують і забезпечують автоматизацію невеликих проєктів. Однак «розумність» у «розумному» місті полягає в колективному використанні даних із цих окремих датчиків для прийняття складних рішень під час надання послуг громадянам. Колективне використання даних дає змогу аналізувати їх у ширшому масштабі порівняно з окремими рівнями, щоб визначити довгострокові закономірності та надати вагому інформацію службам підтримки. Кількість пристроїв IoT, які зараз є у світі, у кілька разів перевищує кількість населення світу. Щоб уможливити обмін даними між цими пристроями, необхідно використовувати бездротові технології, оскільки фізичні з'єднання, по-перше, будуть занадто дорогими (там, де їх можна використовувати), по-друге, не задовольнятимуть вимогам мобільності, типовим для багатьох програм «розумного» міста.

Інтернет забезпечив зв'язок комп'ютерів, смартфонів та інших електронних пристроїв у всьому світі один з одним, дозволяючи миттєво передавати інформацію між ними. Але для IoT Інтернет не обов'язково може бути єдиним способом зв'язку, оскільки багато програм не мають периферійних пристроїв, які можуть підключатися до Інтернету. Додаток може складатися з локальної мережі датчиків, які можуть обмінюватися даними між собою та покладатися на протокол зв'язку з кількома стрибками для надсилання даних до центрального вузла, концентратору або шлюзу. Шлюз може бути фіксованим і підключеним до Інтернету, передаючи будь-які дані, які відстежуються, в хмару для подальшої обробки або використання. Можливо також, що пристрої в програмі можуть використовувати багато різних протоколів, причому центральний вузол має можливість спілкуватися з усіма ними. Звичайним випадком для таких архітектур є «розумний» дім, де виробники виробляють пристрої, використовуючи відповідні або несумісні протоколи, для яких може бути використаний концентратор, і кілька концентраторів комерційно доступні.

Топології мережі

Існує три топології мережі IoT: «точка-точка», «зірка» та mesh-топологія. Перший тип топології — це топологія «точка-точка», де пристрої послідовно з'єднуються один з одним. Мережі «точка-точка» вводять стрибки даних для пакетів, які потрібно надіслати на інші вузли, оскільки дані повинні проходити через кожен вузол на шляху двох вузлів, які хочуть обмінюватися даними. Мережі «точка-точка» не дуже розповсюджені в IoT-системах, оскільки вони мають низький рейтинг стійкості до збоїв і виходять з ладу, якщо у будь-якому з проміжних вузлів станеться збій.

У топології «зірка» всі пристрої в мережі підключені до центрального вузла або шлюзу і не можуть передавати дані один одному напряму. Для того, щоб здійснити обмін даними між собою, пристрої повинні відправити їх через центральний вузол. Мережі із зірковою топологією, з їхньою структурою центрального вузла забезпечують природну схему агрегації для збору даних в IoT, але великі мережі, що складаються з багатьох пристроїв, що може мати місце в більшості програм «розумного» міста, можуть призвести до високої затримки та вузьких місць у сценаріях високої пропускної спроможності інформації. Зіркова топологія використовувалася у різних програмах, включаючи боротьбу зі стихійними лихами і зондуванням навколишнього середовища.

Останній тип мережної топології, яка використовується в Інтернеті речей, — це mesh-топологія, яка дозволяє всім окремим пристроям спілкуватися між собою. Забезпечуючи зв'язок між вузлами в мережі, mesh-топологія пропонує більший діапазон, оскільки дані, що передаються до певного вузла, можуть здійснювати кілька стрибків через мережу, це також підвищує стійкість мережі, оскільки можуть використовуватися альтернативні шляхи, якщо доставка пакетів не вдається через будь-які причини. Вузол стає несправним. Насправді такі топології використовувалися у «розумних» будинках, а також у «розумних» мережах. Існують й інші топології, наприклад, дерево, яке має кілька мереж «зірка», з'єднаних точково.

1.5 Аналіз мережних протоколів IoT для «розумних» міст

Важливо, щоб протокол зв'язку, який використовується у програмі «розумного» міста, відповідав бажаній якості обслуговування (QoS). Проаналізуємо особливості найпопулярніших протоколів бездротових мереж, які використовуються у «розумних» містах.

Радіочастотна ідентифікація (*RFID)* використовує радіочастоти для передавання та отримання даних. Зв'язок RFID складається з двох типів пристроїв, один пристрій – це зчитувач, а інший – тег. Зчитувач зазвичай живиться, і коли тег наближається до зчитувача, після авторизації відбувається обмін інформацією, оскільки тег збирає енергію від зчитувача. Такі теги називаються пасивними, є також активні теги, потужність яких не залежить від зчитувача. Залежно від стандарту, RFID може працювати на різних частотах у радіочастотному спектрі від 125 кГц до 928 МГц і може використовуватися на коротких відстанях. Вони використовуються у таких додатках, як «розумний» транспорт (стягнення податків, паркування), «розумне здоров'я» тощо.

Комунікація ближнього поля (*NFC*) подібна до RFID, проте структура зв'язку NFC не складається з тегів і зчитувачів. На відміну від RFID, обидва пристрої, які хочуть обмінюватися даними за допомогою NFC, потребують живлення, а передавання та отримання даних може відбуватися в обох напрямках, на відміну від RFID. Це дозволяє використовувати NFC для управління та налаштування пристроїв на відміну від RFID, який не можна використовувати для завдань вимірювання або управління. NFC використовує подібні частоти до RFID, але використовується для коротких відстаней. Пристрої NFC популярні для додатків, що включають оплату за допомогою смартфонів, а також використовуються у смарт будинках.

Bluetooth є протоколом з низьким енергоспоживанням, розповсюдженим у додатках IoT, оскільки він може підтримувати необмежену кількість вузлів. Протокол розроблений для короткого діапазону з низькою пропускною спроможністю зв'язку у місці, де пристрої можуть легко виходити або входити в мережу. Bluetooth працює в діапазоні ISM 2,4 ГГц і може мати максимальну швидкість передавання даних 2 Мбіт/с. Bluetooth широко використовується у

«розумному» будинку завдяки тому, що він забезпечує прямий інтерфейс підключення до смартфонів без необхідності будь-якого проміжного пристрою-концентратору.

Z-Wave — це протокол з низьким енергоспоживанням, розроблений для використання у програмах домашньої автоматизації. Це низькошвидкісний протокол з коротким діапазоном, що працює на частотах 868 МГц і 900 МГц. Він працює в режимі головного підлеглого, де головний може мати кілька підлеглих пристроїв, які можуть відповідати на команди від головного вузла. Тому це добре підходить для додатків, де присутній центральний елемент управління, а також потрібно збирати дані з кількох датчиків, таких як «розумні» будинки та «розумні» системи охорони здоров'я.

Li-Fi (Light Fidelity) використовує видиме світло замість радіочастот (RF) для обміну даними. Перевага використання Li-Fi над радіочастотним зв'язком полягає в тому, що він може використовувати вже існуючі системи освітлення, що також призводить до збереження електроенергії. Забезпечує дуже високу швидкість передавання даних на короткі відстані та використовується в системах паркування.

Wi-Fi (Wireless Fidelity) працює на бездротових частотах у діапазонах 2,4 ГГц і 5 ГГц, щоб забезпечити високошвидкісне підключення до мережі Інтернет на обмеженій відстані. Wi-Fi розповсюджений у багатьох додатках для «розумних» міст, оскільки він забезпечує готовий до використання інтерфейс для смартфонів, комп'ютерів та інших переносних гаджетів.

Протокол *ZigBee* розроблений як недорогий протокол з низьким енергоспоживанням для бездротових сенсорних мереж (WSN) і еволюціонував для використання в Інтернеті речей. Протокол ZigBee працює в діапазоні 868 МГц/915 МГц/2,4 ГГц і пропонує помірні швидкості передавання даних із відстанями, подібними до Wi-Fi у схемі передавання даних з кількома стрибками. Радіостанції Zigbee є недорогими пристроями, тому це популярний протокол, який використовують багато виробників пристроїв для «розумного» будинку та охорони здоров'я. Мережа ZigBee має три пристрої: координатор, який є

контролером мережі; маршрутизатор, який відповідає за передавання даних на інші пристрої і кінцевий пристрій ZigBee (сенсори та виконавчі механізми).

Бездротова інтелектуальна мережа комунальних послуг (*Wi-SUN*) - це мережа, схвалена Інститутом інженерів з електротехніки та електроніки (IEEE) і використовується у польових мережах для вимірювання комунальних послуг, автоматизації розподілу комунальних послуг, таких як електроенергія, газ, тощо, а також для системи реагування на попит для комунальних програм. Він підтримує адресацію IPv6 і може використовуватися в зіркоподібній або mesh-конфігурації, де він також забезпечує зв'язок з кількома стрибками.

Cellular. Під стільниковими технологіями розуміють зв'язок 3G, 4G і 5G. Разом із Bluetooth і Zigbee вони є найпопулярнішими технологіями IoT. Стільниковий зв'язок забезпечує високу швидкість передавання даних і підтримує більше програм із багатим вмістом порівняно з іншими протоколами. Завдяки великому радіусу дії вони є кращими для різноманітних застосувань, де потужність не є проблемою. Залежно від технології діапазони стільникового зв'язку варіюються від 600 МГц до 80 ГГц з дуже високою швидкістю передавання даних.

LoRaWAN розшифровується як глобальна мережа великого радіусу дії і це глобальна мережа малої потужності (LPWAN), яка складається з кількох шлюзів і кількох кінцевих пристроїв із шлюзами, підключеними до внутрішнього мережного сервера. Внутрішній сервер забезпечує підключення до хмари. Кінцеві пристрої не мають фіксованого зв'язку з певним шлюзом і можуть надсилати дані на кілька шлюзів, коли їм потрібно передати дані у хмару.

6LoWPAN, що є скороченням від IPv6 через мережі малої потужності, було створено Інженерною робочою групою Інтернету (IETF) спеціально для додатків Інтернету речей з метою забезпечення можливості підключення до мережі Інтернет для невеликих пристроїв. Це мережа на основі IP і використовує зв'язок IPv6. Мережа малого радіусу дії, що працює у промислових, наукових і медичних діапазонах (ISM).

SigFox - власний стандарт, розроблений SigFox Inc., Франція. Він використовує неліцензовані діапазони для виконання надвузькосмугового двонаправленого

зв’язку з низькою швидкістю. SigFox має схожу архітектуру з LoRaWAN і як LoRaWAN, і 6LoWPAN є популярним LPWAN у домені IoT, що пропонує достатньо великі відстані зв’язку до 50 км. SigFox знаходить застосування у безпеці будівель, «розумному» освітленні та моніторингу навколишнього середовища.

NB-IoT (Narrow Band IoT) - тип LPWAN, який працює у діапазонах глобальної системи мобільного зв’язку (GSM) і довгострокового розвитку (LTE). Насправді він може працювати, використовуючи те саме апаратне забезпечення з оновленим програмним забезпеченням, оскільки вважається простою версією LTE.

На рис.1.10 приведені протоколи IoT для «розумних» міст.

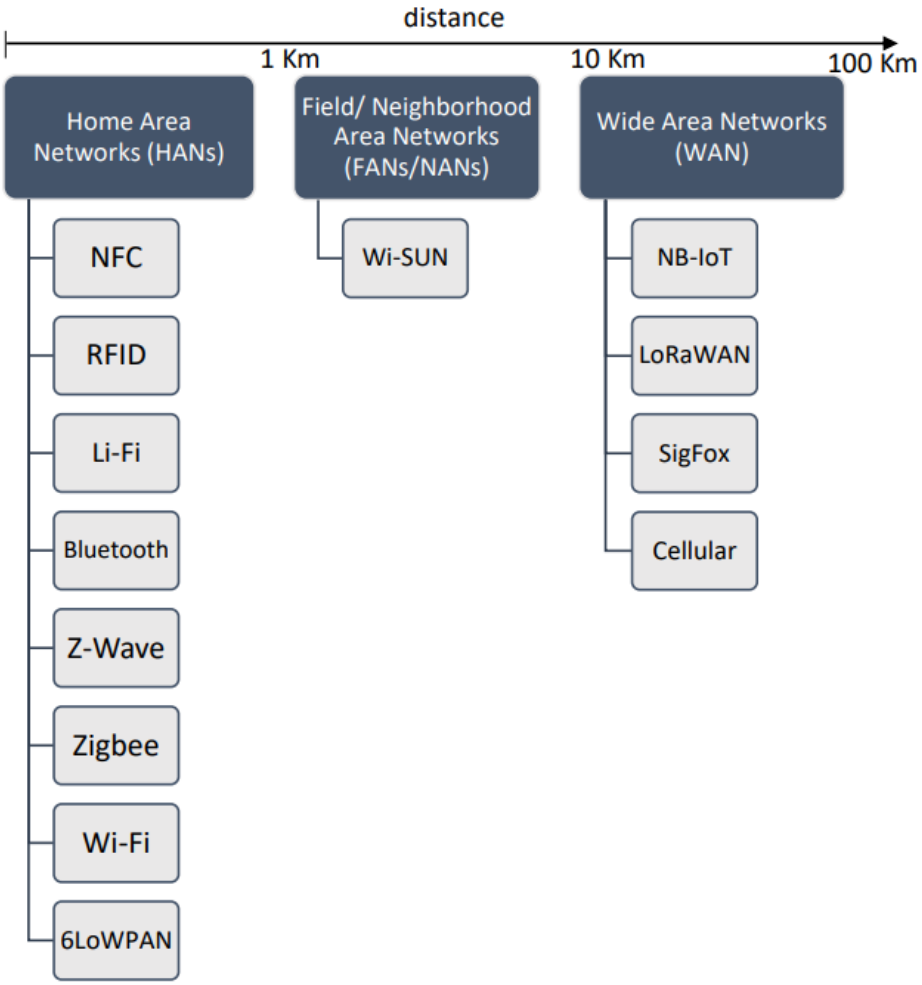


Рисунок 1.10 - Протоколи IoT для «розумних» міст

РОЗДІЛ 2. АНАЛІЗ ТЕХНОЛОГІЙ ШТУЧНОГО ІНТЕЛЕКТУ ДЛЯ SMART CITY

2.1 Аналіз технологій штучного інтелекту

Штучний інтелект (англ. Artificial Intelligence) охоплює широкий спектр технологій, які спрямовані на створення систем, здатних виконувати завдання, які традиційно вимагають людського інтелекту. Ключові технології штучного інтелекту:

1) *Машинне навчання (Machine Learning - ML):* машинне навчання дозволяє системам автоматично вивчати та покращувати свою продуктивність без явного програмування. Алгоритми машинного навчання використовуються для розпізнавання образів, класифікації даних, прогнозування та багатьох інших завдань.

2) *Глибоке навчання (Deep Learning - DL):* підгалузь машинного навчання, яка використовує нейронні мережі з багатьма шарами (глибокими мережами) для вирішення складних завдань, таких як розпізнавання мови або обробка зображень.

3) *Нейронні мережі:* є ключовим елементом глибокого та машинного навчання. Нейронні мережі намагаються моделювати роботу людського мозку для розв’язання завдань розпізнавання та вивчення.

4) *Обробка природних мов (Natural Language Processing - NLP):* технології NLP дозволяють комп’ютерам розуміти, інтерпретувати та взаємодіяти з людською мовою. Вони використовуються у віртуальних асистентах, системах автоматичного перекладу тощо.

5) *Комп’ютерне зорове сприйняття (Computer Vision):* технології комп’ютерного зору дозволяють комп’ютерам розпізнавати та аналізувати зображення та відео. Вони застосовуються у розпізнаванні облич, медичній діагностиці, автономних автомобілях тощо.

6) *Автоматизоване прийняття рішень (Automated Decision Making):* системи штучного інтелекту можуть приймати рішення на основі аналізу даних та

вивчення шаблонів. Це використовується у фінансах, управлінні ланцюгом постачання, медицині та інших галузях.

7) *Розподілене навчання (Distributed Learning):* це підходить до машинного навчання, де моделі навчаються на різних пристроях або вузлах мережі та об'єднуються для отримання загальної моделі.

8) *Голосові технології (Voice Recognition):* системи, які розпізнають та інтерпретують людський голос, застосовуються у віртуальних асистентах, технологіях розпізнавання мови та ін.

Ці технології часто комбінуються для створення більш складних та інтелектуальних систем, які можуть вирішувати різноманітні завдання в різних галузях.

2.2 Дослідження аспектів використання бібліотеки NumPy для штучного інтелекту

Бібліотека NumPy (Numerical Python) – це популярна бібліотека для мови програмування Python, яка надає підтримку для роботи з масивами та математичними функціями для обчислення та аналізу даних. Бібліотека NumPy є важливим інструментом для наукових обчислень, машинного навчання, обробки сигналів, графіки та багатьох інших областей, де необхідна робота з числовими даними та великими масивами даних.

Бібліотека NumPy відіграє важливу роль у штучному інтелекті завдяки своїм можливостям у роботі з числовими масивами та матрицями. Дослідимо аспекти, в яких NumPy є корисним інструментом для розвитку і застосування штучного інтелекту:

- *математичні операції:* NumPy надає ефективні та оптимізовані інструменти для виконання математичних операцій на числових масивах. Це включає в себе операції додавання, віднімання, множення, ділення та багато інших, що є ключовими для алгоритмів штучного інтелекту;
- *векторизація:* NumPy підтримує векторизацію операцій, що дозволяє виконувати операції одночасно для цілих масивів даних, замість використання

циклів. Це забезпечує високий рівень продуктивності та швидкість виконання алгоритмів;

- *масиви:* NumPy використовує масиви, які представляють собою ефективний спосіб зберігання та обробки даних у багатовимірних форматах. Це особливо важливо для обробки даних у вигляді зображень, звуку або інших форматів, які часто використовуються в області штучного інтелекту;

- *лінійна алгебра:* NumPy має розширений набір функцій для роботи з лінійною алгеброю, такими як розв'язання систем лінійних рівнянь, знаходження власних значень та векторів, обчислення детермінанту та інше. Це корисно для багатьох алгоритмів машинного навчання;

- *підтримка рандомізації:* NumPy містить функції для генерації псевдовипадкових чисел, що використовується у багатьох алгоритмах машинного навчання для рандомізації та інших завдань;

- *інтеграція з іншими бібліотеками:* NumPy є частиною екосистеми Python для наукових обчислень, і вона часто використовується в поєднанні з іншими бібліотеками, такими як SciPy, Pandas, Matplotlib тощо для розв'язання різних завдань.

```
import numpy as np

np.__version__
```

```
import scipy

scipy.__version__
```

NumPy оперує масивами. Найрозповсюджений спосіб створення масивів – створення їх із списків. Ми у функцію np.array передаємо наш список і отримуємо потрібний нам масив. В *np.array* є декілька аргументів: *dtype* – у ньому можемо вказати той тип даних, який ми хочемо використовувати. Наприклад, ми хочемо використовувати float64. У нас з'явилося число з плаваючою крапкою (рис.2.1).

```
In [11]: a = np.array([1, 2, 3, 4, 5])
         a
Out[11]: array([1, 2, 3, 4, 5])

In [12]: a = np.array([1, 2, 3, 4, 5], dtype=np.float64)
         a
Out[12]: array([ 1.,  2.,  3.,  4.,  5.])
```

Рисунок 2.1 – Створення масивів

Можна створювати двовимірні масиви. Для цього нам потрібно подати на вхід відповідно двовимірний список або список вкладених списків. Вкладені списки повинні мати однакову довжину – це наша відповідальність. Якщо це буде не так, то ми отримаємо виключення. В *dtype* можна передавати типи даних Python, такі як int, float і т.д., і NumPy їх сам приведе до потрібного внутрішнього типу.

```
In [13]: a = np.array([[1, 2, 3, 4, 5],
                       [6, 7, 8, 9, 0]], dtype=float)
         a

Out[13]: array([[ 1.,  2.,  3.,  4.,  5.],
                [ 6.,  7.,  8.,  9.,  0.]])
```

Рисунок 2.2 – Створення двовимірних масивів

У масивів є декілька атрибутів. Є атрибут *shape*, який говорить про те, яка розмірність у нашого масиву. У нас двовимірний масив 2x5 (рис.2.3).

```
In [14]: a.shape
Out[14]: (2, 5)
```

Рисунок 2.3 – Використання атрибуту *shape*

Є ще атрибут *ndim*, який говорить про те, скільки у нас розмірностей. У даному прикладі їх 2 (рис.2.4).

```
In [15]: a.ndim
Out[15]: 2
```

Рисунок 2.4 - Використання атрибуту *ndim*

Є ще атрибут *dtype*, який повертає нам тип даних, які зберігаються всередині масиву (рис.2.5).

```
In [16]: a.dtype
Out[16]: dtype('float64')
```

Рисунок 2.5 – Використання атрибуту *dtype*

Наш масив ми можемо сконвертувати, привести до іншого типу. Наприклад, ми хочемо привести масив, який в собі зберігає числа з плаваючою крапкою, до типу *int.* Для цього ми викликаємо метод *astype* і вказуємо той тип, який ми хочемо отримати (рис.2.6).

```
In [17]: a.astype(int)
Out[17]: array([[1, 2, 3, 4, 5],
                [6, 7, 8, 9, 0]])
```

Рис.2.6 – Застосування методу *astype*

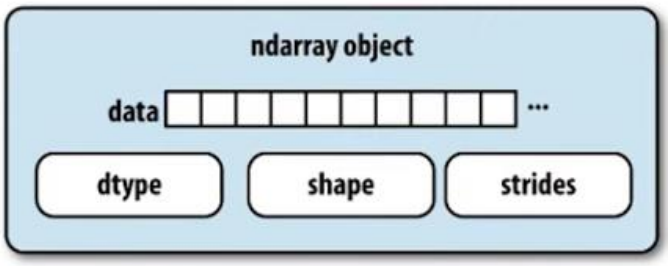


Рисунок 2.7 – Розмірність масивів

Спочатку треба зрозуміти, як багатовимірні масиви зберігаються всередині бібліотеки. Масиви зберігаються як єдиний великий шматок даних, послідовно. Тому, зміна розмірності призводить просто до того, що ми переставляємо вказівники, де в нас починається новий рядок або новий стовпчик, в залежності від того, як нам це зручно, але за замовчуванням – де починається новий рядок.

У NumPy-об’єктів є три атрибути: атрибут *dtype*, який зберігає тип даних, який зберігається всередині масиву; атрибут *shape* – говорить про те, яка розмірність у нашого масиву; атрибут *strides* – показує, на скільки байтів треба відступити, щоб перейти до наступного елементу або у рядку, або у стовпчику.

Наприклад, у нас є двовимірний масив *a* і всередині знаходяться числа з плаваючою крапкою float64 (рис.2.8). Його розмір 8 байтів, а розмірність такого масиву 2x5.

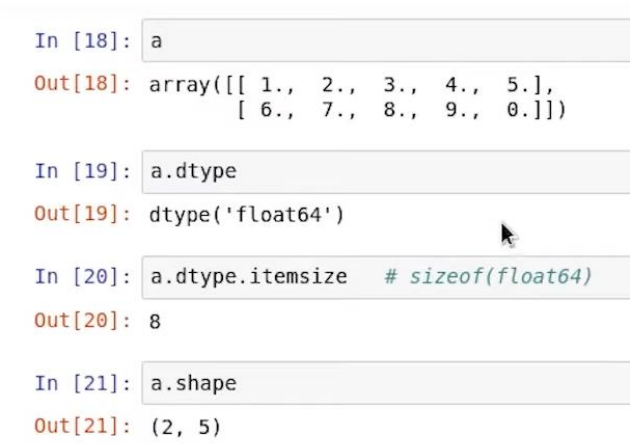


Рисунок 2.8 - Двовимірний масив *a*

Тоді атрибут *strides* має наступний вигляд. Припустимо, ми знаходимося в одиничці і хочемо отримати наступний елемент в рядку. Для цього нам потрібно зрахувати один елемент, розмір цього елементу 8 байтів, відповідно ці 8 байтів ми й зраховуємо (рис.2.9).

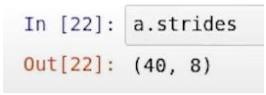


Рисунок 2.9 – Застосування атрибуту *strides*

Щоб перейти до наступного елементу у стовпці, нам потрібно зрахувати весь рядок, тобто 5 елементів: 5*8 = 40. Так як у нас масив зберігається як єдиний шматок пам’яті, то операція по зміні його форми полягає у зміні атрибуту *shape*.

Розглянемо, як змінити форму масиву. Є спеціальний метод *reshape*, в якому ми вказуємо, якої розмірності повинен бути наш новий масив (рис.2.10). У нашому випадку ми хочемо, що у нього була розмірність 5 рядків і 2 стовпці.

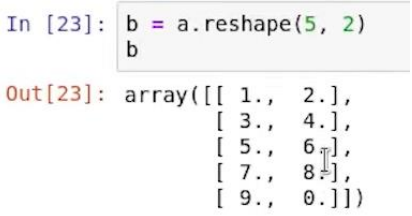


Рисунок 2.10 – Зміна форми масиву

Бачимо, що атрибут *shape* у нас змінився (рис.2.11) і атрибут *strides* також змінився (рис.2.12).

```
In [24]: b.shape
Out[24]: (5, 2)
```

Рисунок 2.11 – Зміна атрибуту *shape*

```
In [25]: b.strides
Out[25]: (16, 8)
```

Рисунок 2.12 - Зміна атрибуту *strides*

Як і в попередньому випадку, для того щоб перейти до наступного елементу в рядку нам потрібно зрахувати всього лише один елемент, тобто 8 байтів, а щоб перейти до наступного елементу у стовпці, нам потрібно зрахувати весь рядок, тобто 2 елементи: 2*8=16.

У деяких випадках у нас може бути багато розмірностей, і ми не хочемо рахувати якого розміру у нас буде кожна розмірність. Наприклад, я хочу отримати масив із двох стовпців, при цьому я не хочу рахувати, що у мене рядків буде 5. Для цього я можу передати -1 і NumPy сам зрозуміє, що рядків у нас буде 5. Аналогічно можна вказати, яка кількість рядків і стовбців, і NumPy сам зрозуміє.

```
In [26]: a.reshape(-1, 2)
Out[26]: array([[ 1.,  2.],
                [ 3.,  4.],
                [ 5.,  6.],
                [ 7.,  8.],
                [ 9.,  0.]])
```

Зміна форми масиву нічого не робить з даними, які зберігаються як єдиний шматок пам’яті. Тому, якщо ми робимо *reshape*, отримуємо новий об’єкт. Новий об’єкт вказує на ті ж самі дані, на які вказував об’єкт *a*. В цьому прикладі ми робимо *reshape* для об’єкту *a* і в об’єкті *a* змінюємо один із елементів (рис.2.13) і як бачимо він у нас змінився і в об’єкті *b*.

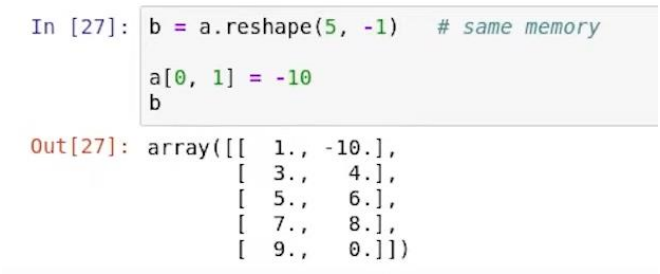


Рисунок 2.13 – Зміна об’єкту *b*

Є декілька способів, як можна багатовимірний масив витягнути в одну лінію, як він зберігається всередині пам’яті. Для цього є метод *flatten*, який і витягує масив в один рядок, але він нам повертає копію.

Витягуємо масив *a* і в масиві *a* змінюємо одне із значень (рис.2.14). Бачимо, що в масиві *b* залишилося старе значення, а в масиві *a* воно змінилося.

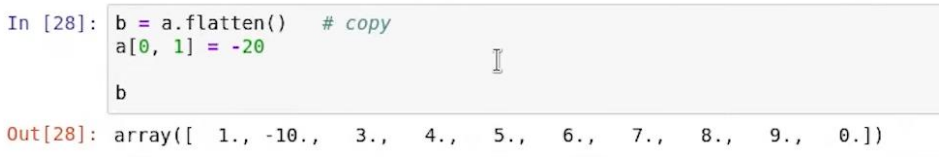


Рисунок 2.14 – Метод *flatten*

Є інший метод *ravel,* який робить схожу операцію, тобто витягує масив в один рядок, але не повертає нам копію. По суті це reshape-1 (рис.2.15). Ми змінюємо в *a* один із елементів, попередньо витягнуту версію ми зберегли в *b*. Ми бачимо, що в *b* теж змінилося.

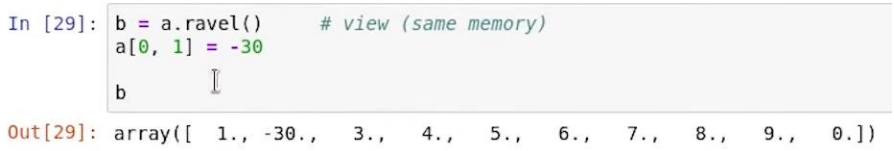


Рисунок 2.15 – Метод *ravel*

Операція транспонування також не призводить до копіювання даних. Щоб транспонувати масив, нам потрібно звернутися до атрибуту *T* і викликати метод *transpose* (рис.2.16). Змінили елемент в a і він змінився в c. Чому так відбувається, тому що в нас масив c вказує на ті самі дані, просто у нас змінився атрибут strides, який показує на скільки нам потрібно відступати (рис.2.17).

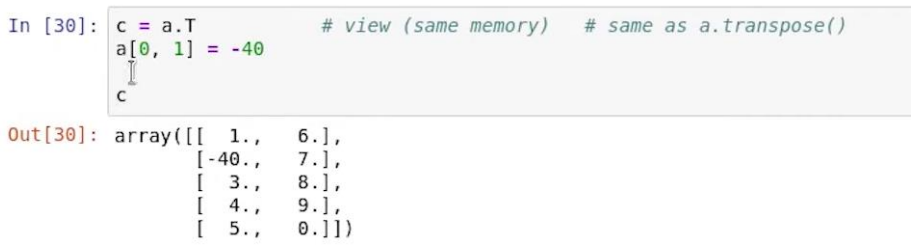
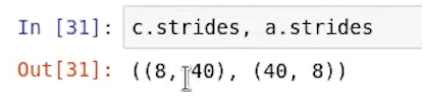


Рисунок 2.16 – Метод *transpose*



Тепер для того, щоб отримати наступний елемент в рядку, нам потрібно прочитати весь стовпчик. 5*8=40. Ми зраховуємо весь стовпчик і переходимо до наступного елементу в рядку. Якщо у стовпчику, то потрібно просто рахувати один елемент: 8 байтів.

Якщо у нас багатовимірний масив – не 2 розмірності, а більше, то метод *transpose* дозволяє нам вказати, які осі ми хочемо поміняти між собою. Це дуже корисно, тому що якщо у нас у масиві, наприклад, 4 розмірності, то операція транспонування не дуже очевидна, що відбувається, тому корисно змінювати осі попарно.

Є ще так називаємі фіктивні осі – це осі, розмірність яких дорівнює 1. Для чого вони потрібні. В деяких випадках потрібно мати уяву, що масив у нас – це рядок або вектор-стовбець.

Рисунок 2.17 – Фіктивна вісь – вісь з розмірністю 1

Якщо ми хочемо, щоб наш масив був вектор-рядком (рис.2.18), потрібно використовувати *np.newaxis* у квадратних дужках і ставити його на першу позицію, тим самим сказавши, що ми хочемо, щоб додалася ще одна фіктивна розмірність на початок. Те ж саме можна зробити за допомогою методу *expand_dims* і явно йому вказати, яку розмірність ми хочемо добавити (рис.2.19).

```
In [33]: b[np.newaxis, :]   # same as b.reshape(1, *b.shape)
Out[33]: array([[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]])
```

Рисунок 2.18 – Використання методу *newaxis*

```
In [34]: np.expand_dims(b, axis=0)
Out[34]: array([[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]])
```

Рисунок 2.19 - Використання методу *expand_dims*

Можна також конвертувати вектор-стовбець. Для цього потрібно добавити нову вісь в кінець і отримуємо масив-стовбець.

```
In [35]: b[:, np.newaxis]   # same as b.reshape(*b.shape, 1)
Out[35]: array([[0],
                [1],
                [2],
                [3],
                [4],
                [5],
                [6],
                [7],
                [8],
                [9]])
```

Дослідимо, чим *np.newaxis* краще, ніж *expand_dims*. Тим, що ми можемо добавляти за допомогою *np.newaxis* декілька фіктивних розмірностей. Наприклад, нам потрібно добавити одну фіктивну розмірність на початок і одну фіктивну розмірність у кінець. Розмірність результуючого масиву буде 1x10x1.

```
In [36]: b[np.newaxis, :, np.newaxis].shape
Out[36]: (1, 10, 1)
```

Цього за допомогою expand_dims зробити неможливо.

Важливо: для операцій над двома масивами потрібно, щоб їх розмірності співпадали.

Масиви можна створювати з деякою визначеною структурою, яка може бути стандартною. Наприклад, можна створювати масив з нулів. Наприклад, нам потрібно створити масив з розмірністю 3 на 2, який заповнений нулями.

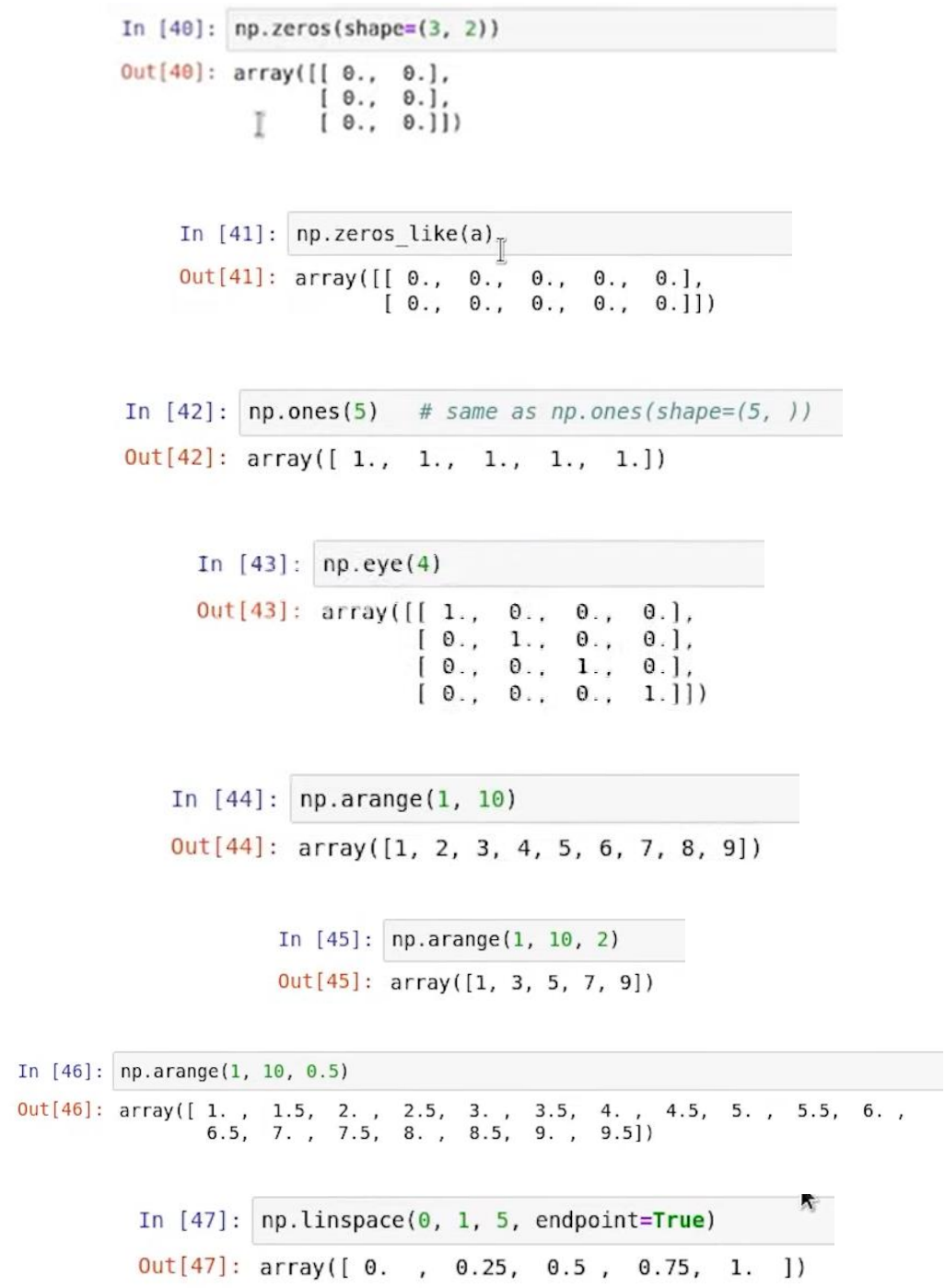


Рисунок 2.20 – Створення масивів з особливими властивостями

Є декілька методів для створення масивів як із списків, так із наявних NumPy масивів. Є 2 таких способи. За допомогою методу *np.array* і за допомогою методу *np.asarray*. *np.asarray* повертає об’єкт, який всередині себе вказує на ті ж самі дані, на які вказував об’єкт *a*. *np.array* створює об’єкт з копією даних із *a*.

```
In [48]: a = np.array([1, 2, 3, 4, 5], dtype=np.float32)
         b = np.asarray(a)
         c = np.array(a)

         b, c

Out[48]: (array([ 1.,  2.,  3.,  4.,  5.], dtype=float32),
          array([ 1.,  2.,  3.,  4.,  5.], dtype=float32))
```

Рисунок 2.21 – Уникнення непотрібного копіювання

Наприклад, всередині *a* ми змінюємо нульовий елемент і дивимося, що відбулося з *b* і *c*:

```
In [49]: a[0] = 0
         b, c

Out[49]: (array([ 0.,  2.,  3.,  4.,  5.], dtype=float32),
          array([ 1.,  2.,  3.,  4.,  5.], dtype=float32))
```

У масиві *b* відбулися зміни, у масиві *c* – не відбулися.

У *np.asarray* можна передавати тип, який ми хочемо використовувати.

```
In [50]: a = np.array([1, 2, 3, 4, 5], dtype=np.float32)
         b = np.asarray(a, dtype=np.int32)
         c = np.array(a)

         a, b, c

Out[50]: (array([ 1.,  2.,  3.,  4.,  5.], dtype=float32),
          array([1, 2, 3, 4, 5], dtype=int32),
          array([ 1.,  2.,  3.,  4.,  5.], dtype=float32))

In [51]: a[0] = 0
         a, b, c

Out[51]: (array([ 0.,  2.,  3.,  4.,  5.], dtype=float32),
          array([1, 2, 3, 4, 5], dtype=int32),
          array([ 1.,  2.,  3.,  4.,  5.], dtype=float32))
```

np.asarray можна використовувати для того, щоб створювати масиви із списків (рис.2.22).

```
In [52]: d = [1, 2, 3, 4, 5]
         a = np.asarray(d)

         d[0] = 0
         a

Out[52]: array([1, 2, 3, 4, 5])
```

Рисунок 2.22 – Створення масивів із списків

При цьому дані будуть скопійовані. Тому всередині списку Python дані зберігаються не так як всередині бібліотеки NumPy.

```
In [53]: a.copy()
Out[53]: array([1, 2, 3, 4, 5])

In [54]: np.copy(a)
Out[54]: array([1, 2, 3, 4, 5])

In [55]: a = np.arange(10).reshape(2, -1)
         a
Out[55]: array([[0, 1, 2, 3, 4],
                [5, 6, 7, 8, 9]])
```

Рисунок 2.23 – Поелементні операції над масивами

Наприклад, у нас є двовимірний масив з 10-ма елементами з розмірністю 2x5. Можна виконувати наступні операції:

```
In [56]: a ** 3   # same as np.power(a, 3)
Out[56]: array([[  0,   1,   8,  27,  64],
                [125, 216, 343, 512, 729]])

In [57]: a + 2    # same as np.add(a, 2)
Out[57]: array([[ 2,  3,  4,  5,  6],
                [ 7,  8,  9, 10, 11]])

In [58]: 2 * a    # same as np.multiply(2, a)
Out[58]: array([[ 0,  2,  4,  6,  8],
                [10, 12, 14, 16, 18]])

In [59]: 2 ** a   # same as np.power(2, a)
Out[59]: array([[  1,   2,   4,   8,  16],
                [ 32,  64, 128, 256, 512]])
```

Рисунок 2.24 – Операції із скалярами та унарні операції

Те саме, що ми робимо з числами, можна виконувати з NumPy-масивами та числами. Операції виконуються поелементно. Для всіх цих операцій є відповідні функції в NumPy, які ми можемо використати.

```
In [60]: np.sqrt(a)
Out[60]: array([[ 0.        ,  1.        ,  1.41421356,  1.73205081,  2.        ],
                [ 2.23606798,  2.44948974,  2.64575131,  2.82842712,  3.        ]])

In [61]: np.exp(a)
Out[61]: array([[  1.00000000e+00,   2.71828183e+00,   7.38905610e+00,
                   2.00855369e+01,   5.45981500e+01],
                [  1.48413159e+02,   4.03428793e+02,   1.09663316e+03,
                   2.98095799e+03,   8.10308393e+03]])

In [62]: np.log(1 + a)
Out[62]: array([[ 0.        ,  0.69314718,  1.09861229,  1.38629436,  1.60943791],
                [ 1.79175947,  1.94591015,  2.07944154,  2.19722458,  2.30258509]])

In [63]: np.log2(1 + a)
Out[63]: array([[ 0.        ,  1.        ,  1.5849625 ,  2.        ,  2.32192809],
                [ 2.5849625 ,  2.80735492,  3.        ,  3.169925  ,  3.32192809]])

In [64]: np.sin(a)
Out[64]: array([[ 0.        ,  0.84147098,  0.90929743,  0.14112001, -0.7568025 ],
                [-0.95892427, -0.2794155 ,  0.6569866 ,  0.98935825,  0.41211849]])
```

Є підтримка порівняння із скалярами. Наприклад, ми хочемо порівняти елемент a з нульом поелементно і ми отримуємо очікуваний результат:

```
In [65]: a > 0   # same as np.greater(a, 0)
Out[65]: array([[False,  True,  True,  True,  True],
                [ True,  True,  True,  True,  True]], dtype=bool)
```

Для цього порівняння також є функція в NumPy, яка виконує ті самі функції.

Є ще підтримка агрегуючих операцій. Створимо одновимірний масив розмірністю 7 (рис.2.25).

```
In [66]: np.random.seed(5656)
         a = np.random.randint(0, 10, size=(7, ))
         a[3] = 10
         a
Out[66]: array([ 4,  6,  7, 10,  0,  3,  5])
```

Рисунок 2.25 – Створення одновимірного масиву розмірністю 7

```
In [67]: a.min(), a.max(), a.argmax(), a.sum(), a.prod(), a.mean()
Out[67]: (0, 10, 3, 35, 0, 5.0)
```

Argmax – позиція, де досягається максимум. Тут ми викликали методи. Те ж саме можна виконати, викликаючи функції із NumPy.

```
In [68]: np.min(a), np.max(a), np.argmax(a), np.sum(a), np.prod(a), np.mean(a)
Out[68]: (0, 10, 3, 35, 0, 5.0)
```

Є можливість використання функції Python min, max, sum:

```
In [69]: # крайне не рекомендуемый вариант
         min(a), max(a), sum(a)   # только для одномерных массивов
Out[69]: (0, 10, 35)
```

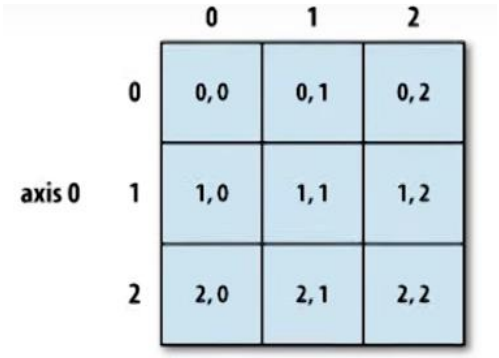


Рисунок 2.26 – *a.agg(axis=axis)* – операція агрегації вздовж осі *axis*

- виконує редукцію (операцію агрегації) по осі axis;
- видаляє вісь axis з вихідного масиву

У двовимірному масиві можна виконувати ті самі операції агрегації. Створимо масив і порахуємо в ньому максимум (рис.2.27).

```
In [70]: np.random.seed(5555)

         a = np.random.randint(0, 10, size=(3, 7))
         a[1, 3] = 15

         a
Out[70]: array([[ 2,  3,  0,  5,  2,  0,  3],
                [ 8,  8,  0, 15,  1,  5,  3],
                [ 0,  1,  6,  2,  1,  4,  5]])

In [71]: a.max()
Out[71]: 15
```

Рисунок 2.27 – Підрахунок максимуму в масиві

Відповідно, якщо мінімум, то 0 (рис.2.28).

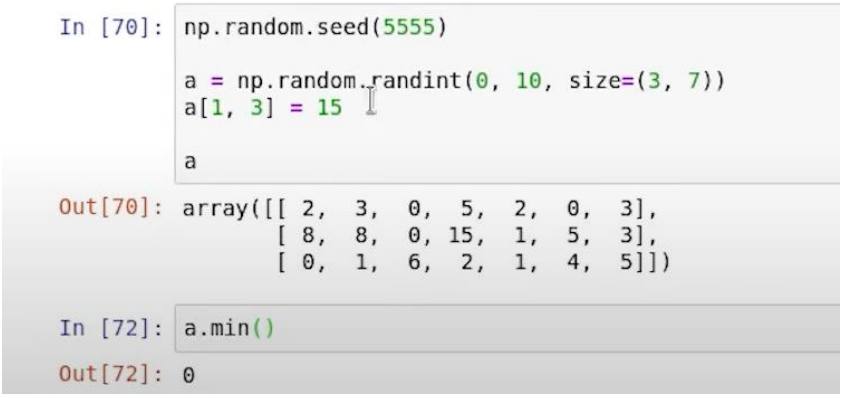


Рисунок 2.28 – Підрахунок мінімуму в масиві

Але тут є можливість виконувати операції агрегації по осям.

Є двовимірний масив. Одна вісь відповідає рядкам (axis 0), інша – стовпцям (axis 1). Ми можемо застосувати агрегуючу операцію вздовж цієї осі. Наприклад, ми хочемо отримати максимум вздовж осі axis 0, тобто для кожного рядка отримати максимум. Або, наприклад, ми хочемо отримати мінімальне значення порядково. Для цього нам потрібно застосувати операцію мінімуму вздовж осі axis 1, тобто порядково. Для цього в агрегуючу операцію ми вказуємо вісь, вздовж якої ми хочемо виконати операцію. Наприклад, вздовж осі 0 ми хочемо порахувати максимальне значення.

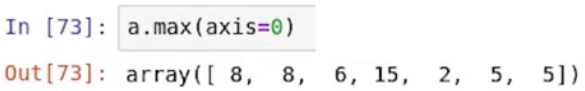


Рисунок 2.29 – Редукція вздовж осі *axis=0* або редукція по стовпцю

Аналогічно по сумі (рис.2.30).

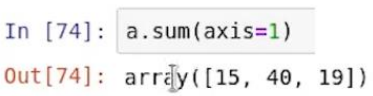


Рисунок 2.30 – Редукція вздовж осі *axis=1* або редукція по рядку

NumPy є важливим інструментом у світі штучного інтелекту, оскільки надає швидкі та потужні засоби для роботи з числовими даними, які часто

зустрічаються у задачах машинного навчання та інших областях штучного інтелекту.

2.3 Алгоритми машинного навчання для Інтернету речей

Інтернет речей (IoT) використовується для збору та обробки даних з різних пристроїв та датчиків, що дозволяє здійснювати різноманітні застосунки в реальному часі. Деякі алгоритми машинного навчання можуть бути використані для аналізу цих даних та виведення корисної інформації. Ось деякі з алгоритмів, які можуть бути застосовані в сфері IoT:

1. Алгоритми класифікації (логістична регресія; метод опорних векторів). *Логістична регресія* використовується для класифікації об'єктів в одному з двох класів. *Метод опорних векторів (SVM)* добре підходить для класифікації великої кількості ознак та об'єктів.

2. Алгоритми кластеризації (k-середні; агломеративна кластеризація). *k-середні* використовуються для групування подібних об'єктів у кластери. *Агломеративна кластеризація* - об'єднання найбільш близьких об'єктів у кластери.

3. Навчання без вчителя (метод головних компонент; автокодувальники). *Метод головних компонент (PCA)* допомагає зменшити розмірність даних, зберігаючи при цьому їхню важливість. *Автокодувальники* використовуються для вивчення представлення вхідних даних та зменшення розмірності.

4. Прогнозування часових рядів (статистична модель ARIMA, довга короткострокова пам'ять LSTM). *Статистична модель ARIMA* використовується для аналізу та прогнозування часових рядів. *Довга короткострокова пам'ять LSTM* - це тип рекурентних нейронних мереж (RNN), розроблений для роботи з послідовностями даних, особливо з короткими та довгими залежностями у послідовностях.

5. Алгоритми для аналізу текстів (моделі векторних представлень слів). *Моделі векторних представлень слів (Word Embeddings)* використовуються для розуміння текстової інформації.

6. Застосування нейронних мереж (згорткові нейронні мережі, рекурентні нейронні мережі). *Згорткові нейронні мережі (CNN)* використовуються для обробки зображень та визначення патернів у великих масивах даних. *Рекурентні нейронні мережі (RNN)* використовуються для роботи з послідовностями даних, такими як часові ряди або текст.

Ці алгоритми можуть бути використані окремо або в комбінації для досягнення бажаного результату в аналізі даних IoT. Вибір конкретного алгоритму залежить від конкретного завдання та характеристик даних.

РОЗДІЛ 3. ІНТЕГРАЦІЯ ТЕХНОЛОГІЙ ІЗ ПІДТРИМКОЮ IoT ТА ШТУЧНОГО ІНТЕЛЕКТУ У SMART CITY

3.1 Дослідження IoT-технологій, які інтегруються у Smart City

Smart City («розумне» місто) - це концепція розвитку міст, яка використовує різноманітні технології та інновації для поліпшення якості життя мешканців, оптимізації використання ресурсів та забезпечення ефективності управління. Основна ідея полягає в тому, щоб використовувати інформаційні та комунікаційні технології для «розумного» управління різними аспектами міського життя. Основні характеристики Smart City включають:

- *інтегровану інфраструктуру*: використання технологій для інтеграції різних аспектів міської інфраструктури, таких як транспорт, енергетика, водопостачання, освітні та медичні системи.
- *ефективне використання ресурсів*: міста мають оптимізувати використання ресурсів, зменшуючи відходи та споживання енергії.
- *забезпечення безпеки*: використання систем відеоспостереження, сенсорів та аналітичних інструментів для підвищення рівня безпеки та реагування на небезпеки.
- *ефективний транспорт*: впровадження технологій для оптимізації транспортної системи, управління трафіком, підтримки громадського транспорту та зменшення забруднення.
- *зручність для мешканців*: забезпечення доступу до інформації для мешканців, покращення сервісів та створення комфортних умов для проживання.
- *електронне управління*: застосування електронних систем для кращого управління міськими послугами та рішенням проблем.
- *активна участь громади*: залучення громадян до управління містом, забезпечення можливості висловлювати свої ідеї та враховувати їх у прийнятті рішень.

Для досягнення цих цілей Smart City використовує різноманітні технології, такі як Інтернет речей (IoT), аналітика даних, штучний інтелект, датчики та інші інноваційні рішення. Основна мета - створити інтелектуальне місто, яке ефективно використовує ресурси та поліпшує якість життя своїх мешканців. Побудова Smart City включає в себе використання різноманітних технологій для оптимізації міських систем та поліпшення якості життя мешканців. Ключові технології, які використовуються для створення Smart City, включають:

1. **Інтеренет речей (IoT).** Використання сенсорів та з'єднань для збору даних з різних джерел, таких як транспорт, енергозабезпечення, водозабезпечення, відходи та інші для розумного управління містом.

2. **Штучний інтелект (ШІ).** Використання алгоритмів машинного навчання та інших технологій штучного інтелекту для аналізу великих обсягів даних, прогнозування, оптимізації систем управління, аналізу безпеки та прийняття рішень.

3. **Системи збірної обробки даних (Big Data).** Обробка та аналіз великих обсягів структурованих та неструктурованих даних для отримання цінної інформації про функціонування міста.

4. **Системи енергоефективності.** Впровадження технологій для моніторингу та управління енергоспоживанням в будівлях та громадських просторах.

5. **Смарт-транспорт та мобільність:**

5.1 *Системи громадського транспорту:* інтеграція технологій для оптимізації графіків, розкладів та платіжних систем;

5.2 *Доступність електротранспорту та автономних транспортних засобів (АТЗ):* розробка інфраструктури та систем навігації.

6. **Системи управління водозабезпеченням:**

- *мережі датчиків для моніторингу якості води*: виявлення забруднень та оптимізація використання води.

7. **Відходозбір та утилізація:**

- *системи маршрутизації для збору відходів*: використання даних для оптимізації маршрутів та підвищення ефективності збору відходів.

8. **Системи громадської безпеки:**

- *мережі відеоспостереження*: використання камер та аналітики для забезпечення безпеки та виявлення надзвичайних ситуацій.

9. **Централізовані системи управління:**

- *панельні управління та інформаційні системи*: забезпечення централізованого управління та надання інформації мешканцям.

Ці технології повинні працювати спільно для створення інтелектуального та ефективного міста, яке відповідатиме потребам мешканців та надавати їм зручність, оптимізуватиме використання ресурсів та сприятиме сталому розвитку. Побудова Smart City вимагає комплексного підходу до інтеграції цих технологій для створення міст, які у майбутньому будуть не лише технологічно передовими, але й забезпечуватимуть підвищену якості життя громадян.

Smart City використовує різноманітні технології Інтернету речей (IoT) для забезпечення покращення інфраструктури, ефективного управління ресурсами, для оптимізації різних аспектів міського життя та поліпшення якості життя мешканців. Ключові технології IoT, які інтегруються у Smart City:

1) Системи моніторингу та управління транспортом:

- смарт-системи світлофорів: регулювання руху транспорту з урахуванням потоків;

- системи паркування: визначення вільних місць та оптимізація паркування.

2) Енергетичний менеджмент:

- смарт-лічильники: забезпечення ефективного використання енергії та моніторинг споживання;

- системи енергозбереження: автоматизація освітлення, опалення та кондиціювання.

3) Управління водоспоживанням:

- системи моніторингу якості води: відстеження рівня забруднення та попередження про аварії;

- розумні системи поливу: оптимізація зрошення з урахуванням погодних умов.

4) Системи відходів:

- сміттєві баки з IoT-системами: моніторинг та оптимізація процесу збору відходів.

5) Громадська безпека:

- відеоспостереження: моніторинг громадських місць для забезпечення безпеки;
- датчики для виявлення надзвичайних ситуацій: попередження про пожежі, повені тощо.

6) Системи для громадської інфраструктури:

- смарт-крісла та смарт-лавиці в парках: забезпечення підзарядки пристроїв та надання інтернет-підключення;
- інтерактивні табло та кіоски: доступ до інформації про громадський транспорт, події тощо.

7) Системи збору та аналізу даних:

- централізовані системи управління: моніторинг та аналіз даних для прийняття рішень щодо розвитку міста;
- аналітика для прийняття рішень: використання штучного інтелекту та аналітичних інструментів для оптимізації управління містом.

8) Датчики та системи моніторингу:

- датчики руху та вимірювання викидів: використовуються для моніторингу та контролю руху транспорту та рівня забруднення повітря;
- датчики водостічних систем: допомагають виявляти та управляти проблемами водозабезпечення та розвитком повеневих ситуацій.

9) Мережі зв’язку:

- LPWAN (Low Power Area Network): дозволяє підключати велику кількість датчиків та пристрої з низьким споживанням енергії та великим радіусом дії;
- 5G мережі: забезпечують високу швидкість передавання даних та низьку затримку, що важливо для реалізації різноманітних IoT-застосувань.

10) Смарт-системи транспорту:

- GPS та сенсори руху: використовуються для моніторингу руху транспорту та оптимізації систем громадського транспорту;

- системи управління транспортним потоком: використовуються дані з датчиків для управління рухом на дорогах.

11) Інтелектуальні системи управління будівлями:

- системи «розумного будинку»: автоматизація систем опалення, освітлення, кондиціонування та безпеки.

Технології Інтернету речей (IoT) відіграють важливу роль у розвитку розумних міст, де різноманітні пристрої і об'єкти пов'язуються мережею для збору, обміну та аналізу даних з метою оптимізації функціонування міської інфраструктури та покращення якості життя громадян. Ці технології спільно використовують дані та забезпечують інтеграцію, допомагають зробити міста більш ефективними, екологічно чистими та зручними для мешканців, а також сприяють збереженню ресурсів та підвищенню якості життя.

У табл.3.1 приведене порівняння мережних технологій IoT для Smart City.

Таблиця 3.1 - Порівняння мережних технологій IoT для Smart City

Architecture	Technology	Frequency/Medium	Data rate	Range	Topology
Home Area Networks (HANs)	NFC	125 KHz, 13.56 MHz/860 MHz	106 Kbps, 212 Kbps or 424 Kbps	10 cm	Point to Point
	RFID	125 KHz, 13.56 MHz/902-928 MHz	4 Mbps [82]	3 - 10 m	Point to Point
	Li-Fi	LED Light	1 – 3.5 Gbps [85]	10 m	Point to point, Star, Mesh
	Bluetooth	2.4 GHz	Up to 2 Mbps	240 m	Star
	Z-wave	868 MHz/900 MHz	40-100 Kbps	30 – 100 m	Mesh
	Zigbee	868 MHz/915 MHz/2.4 GHz	250 Kbps	Up to 100 m	Mesh, Star, Tree
	Wi-Fi	2.4 GHz/5 GHz	54 Mb/s, 6.75 Gb/s	140 m , 100 m	Tree
	6LOWPAN [82]	868 MHz/915 MHz/2.4 GHz	Up to 250 Kbps	10 - 100 m	Mesh, Star
Field/Neighborhood Area Networks (FANs/NANs)	Wi-SUN	868 MHz/915 MHz/2.4 GHz	Up to 300 Kbps	Up to 4 Km	Star, Mesh
Wide Area Networks (WANs)	NB-IOT	Licensed LTE bands	200 Kbps	1 - 10 Km	Tree
	LoRaWAN	433 MHz/868 MHz/915 MHz	50 Kbps	5 - 20 Km	Star of Star (nested star)
	Sigfox	433 MHz/868 MHz/915 MHz	100 bps	10 – 50 Km	One hop star
	3G	1.8 – 2.5 GHz	2 Mbps	-	Tree
	4G	600 – 5.925 GHz	up to 1 Gbps	-	Tree
	5G	600 - 80 GHz	Up to 20 Gbps	-	Tree

3.2 Дослідження технологій штучного інтелекту, які інтегруються у Smart City

Smart City використовує різноманітні технології штучного інтелекту для ефективного управління та оптимізації міських систем. Ключові технології штучного інтелекту, які інтегруються у Smart City:

1) Системи прогнозування і аналітики:

- Прогнозування транспортного потоку: Використання алгоритмів штучного інтелекту для прогнозування обсягу транспортного руху, що допомагає оптимізувати системи громадського транспорту та управління трафіком;
- Аналітика для управління водозабезпеченням: Використання штучного інтелекту для аналізу даних щодо споживання води та виявлення можливостей зменшення втрат та оптимізації систем водопостачання.

2) Системи управління енергозабезпеченням:

- Прогнозування споживанням електроенергії: Використання алгоритмів штучного інтелекту для прогнозування пікового споживання та оптимізації роботи системи енергозабезпечення міста;
- Системи енергоефективності: Використання штучного інтелекту для аналізу та оптимізації енергоефективних будівель та інших об’єктів.

3) Смарт-транспорт та автономні транспортні засоби:

- Автономні транспортні засоби: Використання систем машинного навчання та комп’ютерного зору для навігації та управління;
- Системи моніторингу та діагностики транспорту: Використання штучного інтелекту для прогнозування потреб у технічному обслуговуванні, виявлення несправностей та оптимізації роботи транспортних систем.

4) Системи управління водозабезпеченням:

- Системи виявлення витоків та забруднень: Використання сенсорів та аналітики штучного інтелекту для виявлення можливих проблем у системах водопостачання та забезпечення якості води.

5) Системи громадської безпеки:

- Аналітика відеоспостереження: Використання алгоритмів штучного інтелекту для виявлення небезпек та реагування на події у режимі реального часу;
- Прогнозування злочинності: Використання аналітики штучного інтелекту для прогнозування можливих зон збільшення злочинності та оптимізації патрулювання.

6) Системи управління відходами:

- Маршрутизація збору сміття: Використання штучного інтелекту для оптимізації маршрутів та часу збору відходів.

Ці технології штучного інтелекту спільно використовують дані для прийняття інформованих рішень, підвищення ефективності та поліпшення якості життя в містах. Штучний інтелект забезпечує можливість збору, аналізу та обробки великих обсягів даних для прийняття інформованих рішень щодо розвитку міста, планування інфраструктури та вдосконалення послуг для мешканців.

3.3 Інтеграція IoT-технологій та штучного інтелекту у Smart City

IoT-технології та технології штучного інтелекту інтегруються у Smart City для оптимізації різних міських систем та поліпшення якості життя. Дослідимо, яким чином ці технології співпрацюють у Smart City:

1) **Збір та аналіз даних**:

- ***IoT***: Датчики розташовані в різних частинах міста, таких як транспорт, енергетика, водопостачання, датчики якості повітря та інші, збирають великі обсяги даних;
- ***штучний інтелект***: Алгоритми машинного навчання та аналітичні інструменти використовуються для обробки цих даних, виявлення закономірностей, розпізнавання шаблонів та прогнозування тенденцій.

2) **Оптимізація транспортних систем**:

- ***IoT***: Датчики на дорогах, у транспортних засобах та громадському транспорті забезпечують потрібну інформацію про транспортний потік та рух машин;

- ***штучний інтелект***: Аналітика використовується для прогнозування транспортних заторів, оптимізації графіків громадського транспорту та підвищення ефективності транспортних систем.

3) **Ефективне управління енергетикою**:

- ***IoT***: Смарт-лічильники та сенсори моніторять використання електроенергії та інших ресурсів у будівлях та інфраструктурі;
- ***штучний інтелект***: Алгоритми визначають ефективність та оптимізують використання енергії, забезпечуючи економію та зниження викидів.

4) **Системи управління водозабезпеченням**:

- ***IoT***: Датчики виявляють витоки, моніторять рівень та якість води;
- ***штучний інтелект***: Аналітика використовується для оптимізації водозабезпечення, реагування на аварійні ситуації та підтримки сталого управління водними ресурсами.

5) **Громадська безпека**:

- ***IoT***: Відеоспостереження та датчики безпеки реєструють події та стан громадських місць;
- ***штучний інтелект***: Аналітичні системи використовуються для розпізнавання небезпек, виявлення аномальних ситуацій та попередження можливих загроз.

6) **Системи управління відходами**:

- ***IoT***: Датчики у контейнерах для сміття моніторять рівень наповнення;
- ***штучний інтелект***: Системи аналізують дані, щоб оптимізувати маршрути та час збору відходів.

Ці технології працюють разом, використовуючи дані, щоб розумно управляти різними аспектами міського життя, підвищувати ефективність та забезпечувати більш зручні умови для мешканців (рис.3.1).

AIoT може обробляти величезний обсяг даних, зібраних від IoT-пристроїв, і витягати з них значущі знання. Це допомагає в управлінні бізнес-процесами, прийнятті рішень та забезпеченні ефективності виробництва. Інтеграція AI та IoT дозволяє автоматизувати ряд задач і процесів. Наприклад, в промисловості це може бути відслідковування та оптимізація виробничих процесів.

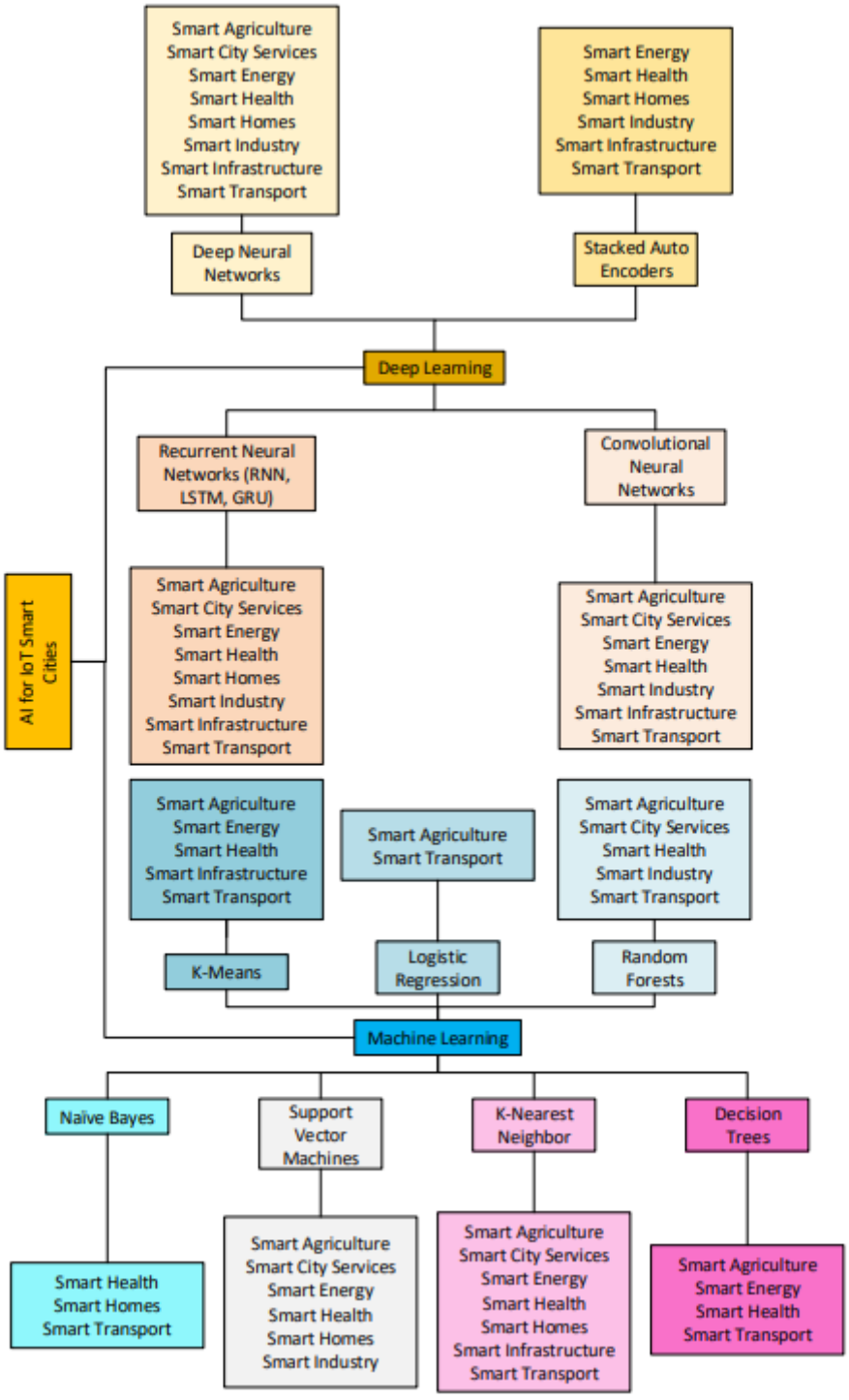


Рисунок 3.1 - AI для «розумних» міст IoT

AIoT дозволяє виявляти аномалії та потенційні загрози на ранніх етапах, забезпечуючи високий рівень кібербезпеки для систем Інтернету речей. Штучний інтелект може допомагати оптимізувати використання енергії, наприклад, регулюючи роботу енергоефективних систем опалення, кондиціювання повітря

тощо. Впровадження технології AIoT може дати нові можливості для бізнесу через розумний аналіз даних, персоналізовані послуги та інші інноваційні підходи. У сферах, таких як медицина та охорона здоров'я, AIoT може сприяти розробці продуктів та послуг, які покращують якість життя, забезпечуючи персоналізовані та ефективні рішення. AIoT відіграє ключову роль у впровадженні концепції Індустрії 4.0, де підприємства стають більш підключеними, автоматизованими та інтелектуалізованими.

Штучний інтелект може стимулювати розробку нових та інноваційних продуктів, що може призвести до нових ринків та вищого рівня конкурентоспроможності. Але, важливо враховувати етичні та безпекові аспекти при впровадженні технології AIoT, також звертати увагу на захист приватності та обробку особистих даних.

3.4 Використання штучного інтелекту для системи «розумного» будинку

Досліджуючи систему «розумного» будинку, можна визначити наступні основні методи автоматизації такої системи: метод автоматизації з використанням GSM; з використанням Bluetooth; на базі телефону; з використанням технології ZigBee; з використанням бездротової мережі; метод автоматизації змішаного типу, який поєднує в собі технології GSM, Bluetooth та ZigBee.

Система «розумного» будинку на базі технології GSM використовує мобільні мережі для забезпечення комунікації та управління різними пристроями в будинку. Основні елементи такої системи включають в себе: мобільний інтерфейс; системи віддаленого управління; системи безпеки; управління енергозбереженням; моніторинг параметрів; інтеграцію з іншими системами. Система використовує мобільний зв'язок (GSM, 3G, 4G, або навіть 5G) для взаємодії з різними пристроями та серверами системи. Це дозволяє нам отримувати відомості про стан будинку та виконувати команди з використанням смартфону або іншого пристрою з підтримкою мобільного зв'язку. Загалом, це

дозволяє власникам будинків ефективно управляти та моніторити свій будинок в реальному часі, навіть якщо вони далеко від нього.

Методи автоматизації системи розумного будинку на базі технології Bluetooth можуть включати в себе використання спеціальних пристроїв та програмного забезпечення, які підтримують цю технологію. Важливо відзначити, що Bluetooth може використовуватися для певних аспектів автоматизації, але не завжди є єдиною технологією, що використовується у «розумних» будинках. Зазвичай вона комбінується з іншими технологіями, такими як Wi-Fi, Zigbee, Z-Wave та іншими. Існує багато різних рішень та пристроїв, які можна інтегрувати у систему «розумного» будинку на базі технології Bluetooth, і вони можуть керуватися за допомогою спеціальних додатків або голосових команд.

Автоматизація системи «розумного» будинку на базі телефону може використовувати різні технології та підходи. Основна ідея полягає в тому, щоб використовувати телефон як центральний контрольний пункт для управління різними пристроями та системами вдома. Розробка спеціальних мобільних додатків, які дозволяють користувачеві взаємодіяти з різними пристроями в будинку через телефон. Це може бути основним інтерфейсом для управління освітленням, опаленням, кондиціонуванням, системою безпеки. Використання пристроїв IoT, які можуть взаємодіяти між собою та з телефоном через хмарні сервіси. Це може включати сенсори, камери, датчики руху, які передають дані через мережу Інтернет. Можливість налаштовувати сценарії та розклади автоматизації за допомогою мобільного додатку. Наприклад, можна налаштовувати, щоб світило вмикалося автоматично під час нашого повернення додому або щоб термостат знижував температуру під час нашої відсутності. Ці підходи можуть варіюватися в залежності від виробників обладнання та програмного забезпечення, а також від конкретних можливостей телефонів та їх операційних систем.

Системи «розумного» будинку, які використовують технологію ZigBee для автоматизації, володіють кількома характеристиками, які роблять їх ефективними та зручними. ZigBee спроектований для роботи на батареях

протягом тривалого часу. Це дозволяє підключати бездротові датчики, вимикачі та інші пристрої до системи «розумного» будинку, не переймаючись частою заміною батарей. ZigBee базується на відкритих стандартах, що робить його привабливим для виробників різноманітних пристроїв. Це сприяє створенню систем, де пристрої різних виробників можуть взаємодіяти між собою. ZigBee оптимізований для відправки невеликих порцій даних, що робить його ідеальним для сенсорів, вимикачів та інших пристроїв IoT. Системи ZigBee можуть динамічно вирішувати проблеми, такі як інтерференція або втрата зв'язку, шляхом вибору оптимального шляху для передавання даних.

Способи автоматизації системи «розумного» будинку на базі бездротової мережі можуть включати в себе використання різних технологій та пристроїв для забезпечення комунікації та управління різними аспектами будинку. Основні елементи автоматизації системи розумного будинку на базі бездротової мережі включають: бездротові мережі (Wi-Fi, ZigBee, Z-Wave), сенсори, актуатори, хмарні сервіси, які використовуються для зберігання та обробки даних, а також для дистанційного керування системою «розумного» будинку через мережу Інтернет, системи автоматизації та управління, протоколи комунікації, які визначають стандарти та правила для обміну інформацією між різними пристроями. Завдяки цим елементам і їх бездротовій взаємодії користувач може з легкістю управляти та моніторити різні аспекти свого будинку, такі як освітлення, опалення, кондиціювання повітря та безпеку.

Метод автоматизації системи «розумного» будинку змішаного типу передбачає використання комбінацій різних технологій та пристроїв для створення ефективної та інтелектуальної системи управління будинком. Повинна бути забезпечена інтеграція різних типів пристроїв з використанням центральної системи управління, яка може взаємодіяти з різними типами пристроїв. Важливим аспектом такого підходу є використання хмарних технологій для збереження та обробки даних у хмарі для забезпечення доступу до системи з будь-якого місця за допомогою мережі Інтернет. Важливим аспектом даного методу автоматизації системи «розумного» будинку є використання алгоритмів машинного навчання та

штучного інтелекту. Підхід до автоматизації системи «розумного» будинку змішаного типу намагається об'єднати різноманітні технології та функціонал в єдину інтегровану систему для забезпечення зручності, ефективності та безпеки мешканців будинку.

Користувач повинен мати доступ до системи «розумного» будинку в режимі реального часу із будь-якої точки світу.

У табл.3.2 на основі проведених досліджень вказані особливості різних методів автоматизації системи «розумного» будинку.

Таблиця 3.2 - Особливості різних методів автоматизації системи «розумного» будинку

Система	GSM	Bluetooth	Телефон	Zigbee	Wireless
Первинна комунікація	SMS-повідомлення	Команди Bluetooth та AT	Телефонні лінії	Команди Zigbee та AT	Радіо, інфрачервоні або інші хвилі
Віддаленість пристрою	Доступ з будь-якої точки світу	Обмежений до 100 метрів	Будь-де з телефонної лінії	До 10 метрів	Залежно від дальності та спектру хвиль, що використовуються
Кількість пристроїв	Необмежена	Необмежена	12 приладів	Необмежена	Необмежена
Вартість	Висока вартість через SMS-платежі	Залежить від дальності пристрою	Дешева	Дешева	Висока вартість через ліцензування та інші проблеми спектру
Пропускна спроможність	Низька	Залежить від віддаленості пристрою	Висока	Висока	Залежно від стандарту
Управління в реальному часі	-	+	-	+	+

На рис.3.2 представлено тенденцію розвитку технологій Smart Home та Artificial Intelligence.

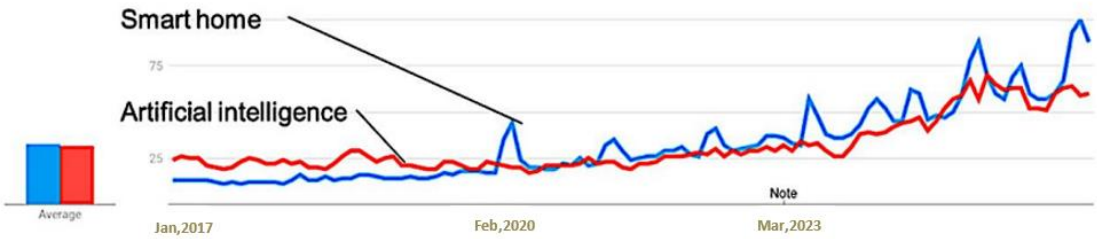


Рисунок 3.2 – Тенденція розвитку технологій Smart Home та Artificial Intelligence

Проаналізувавши джерела, тенденції розвитку Smart Home та Artificial Intelligence, опишемо основні функції Smart Home та Artificial Intelligence. Результати представимо у табл.3.3.

Таблиця 3.3 - Основні функції Smart Home та Artificial Intelligence

№ п/п	Функції Smart Home	Функції Artificial Intelligence
1	Управління пристроями	Розпізнавання діяльності
2	Управління енергією	Обробка даних
3	Охорона здоров'я	Прийняття рішень
4	Інтелектуальна взаємодія	Розпізнавання зображень
5	Безпека	Прогнозування
6		Розпізнавання голосу

Станом на сьогоднішній день відбувається стрімке зростання інформаційних технологій як у нашій країні, не зважаючи на воєнні дії, так і у всьому світі. Відповідно збільшується кількість приладів у будинках. Відбувається впровадження штучного інтелекту у системи «розумного» будинку для керування ресурсами, контролю. Інтелектуальне керування в системі «розумного» будинку може бути реалізоване аналізом даних сенсорної мережі, вивченням попередньої поведінки користувачів або моделей користувачів, використовуючи алгоритм логістичної класифікації з використанням бібліотеки TensorFlow. TensorFlow – комплексна платформа для машинного навчання з відкритим кодом, розроблена

командою Google Brain. Використовується в системі розпізнавання мови DeepSpeech, нейромережі BERT для NLP-задач (рис.3.3).

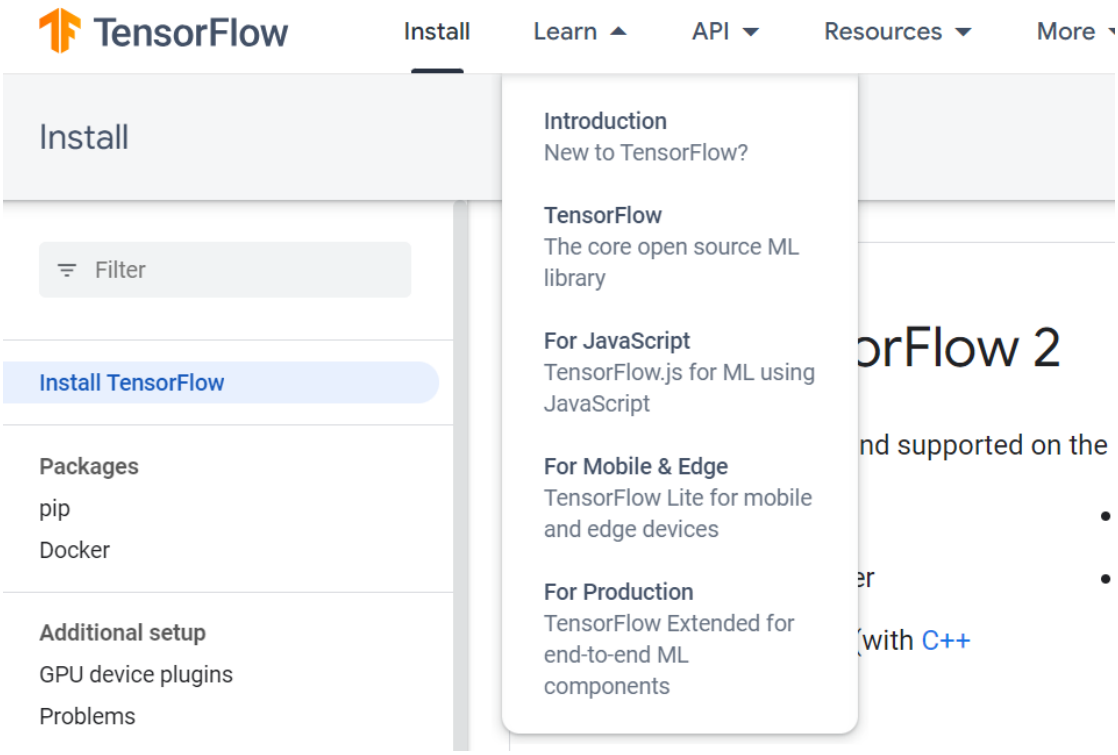


Рисунок 3.3 - Фреймворк TensorFlow

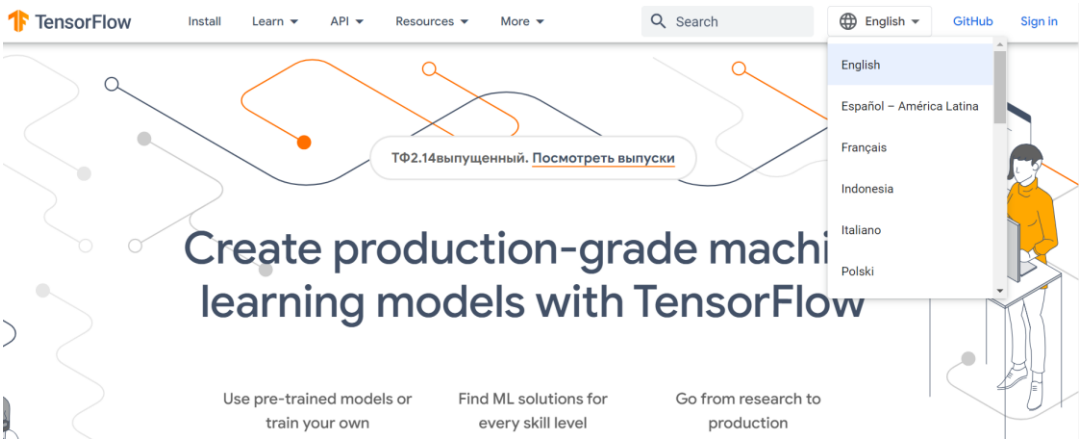


Рисунок 3.4 - Багатомовна документація фреймворку TensorFlow

На рис.3.5 показані способи взаємодії між користувачами, Smart Home та Artificial Intelligence. На рис.3.5 (а) показано, що користувачі дають команди пристроям, а Artificial Intelligence, який вбудований у кожний пристрій. На рис.3.5 (б) показано, що користувачі дають інструкції Artificial Intelligence, який виконує контроль кожного пристрою.

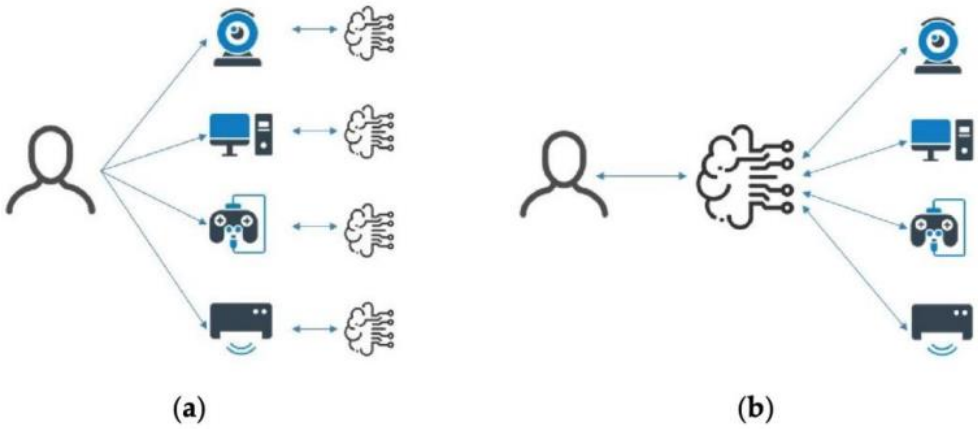


Рисунок 3.5 - Способи взаємодії між користувачами, Smart Home та Artificial Intelligence

Невід'ємною складовою Artificial Intelligence є штучні нейронні мережі (рис.3.6). TensorFlow надає гнучкість для визначення та тренування широкого спектру моделей машинного навчання, включаючи нейронні мережі, глибокі нейронні мережі та інші. TensorFlow має великий та активний користувацький та розробницький комунітет. Це дозволяє легко знаходити підтримку, документацію та різноманітні розширення для фреймворку.

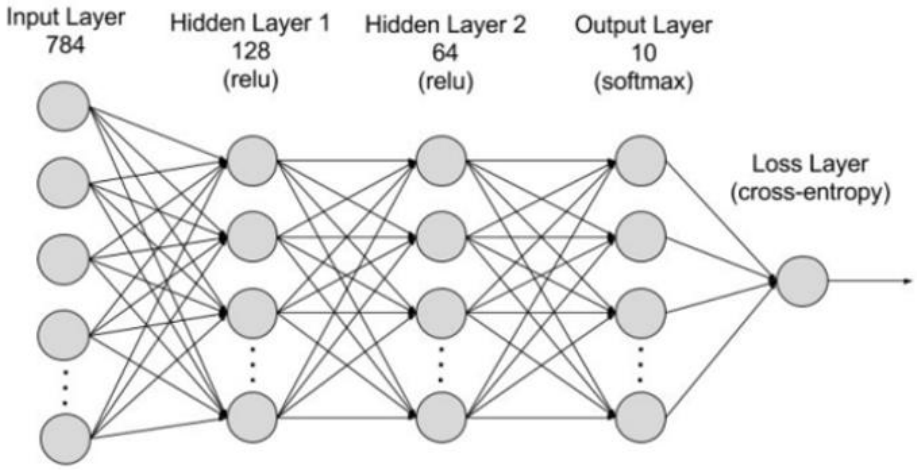


Рисунок 3.6 - Структура штучної нейронної мережі

Нейронні мережі відіграють ключову роль для «розумних» будинках, де вони використовуються для різноманітних завдань автоматизації, оптимізації та підвищення ефективності систем. Основні ролі нейронних мереж у розумних

будинках включають: розпізнавання голосу та обробку природної мови; комп’ютерне зорове спостереження; аналітику енергоспоживання; автоматизацію системи кондиціювання повітря та терморегуляцію; системи безпеки та виявлення вторгнень; прогнозування та оптимізацію роботи системи; розпізнавання пристроїв IoT. У цілому, нейронні мережі відіграють важливу роль у забезпеченні інтелектуальної, ефективної та зручної роботи систем «розумних» будинків.

Використання нейронних мереж для розпізнавання голосу дозволяє системам «розумного» будинку розуміти команди та інструкції, які надходять від користувачів через голосові асистенти, такі як Siri, Alexa або Google Assistant. Використання нейронних мереж для аналізу відеопотоків з камер для виявлення рухів, розпізнавання обличь, розпізнавання об'єктів та інших завдань комп'ютерного зору для забезпечення безпеки та контролю за приміщенням.

Використання нейронних мереж для аналізу та прогнозування енергоспоживання у будинку. Моделі можуть оптимізувати роботу систем опалення, кондиціювання повітря та освітлення, щоб зменшити споживання електроенергії. Нейронні мережі можуть аналізувати історію та звички користувачів для оптимізації роботи систем кондиціювання повітря та терморегуляції, забезпечуючи комфортні умови та економію енергії. Нейронні мережі використовуються для аналізу відеопотоків та виявлення аномальних подій, таких як вторгнення або пожежа. Відповідно це підвищує рівень безпеки у будинку.

Використання нейронних мереж для аналізу даних про роботу систем у будинку та прогнозування оптимальних параметрів для забезпечення оптимальної ефективності та комфорту. Використання нейронних мереж для ідентифікації та управління підключеними пристроями у «розумному» будинку, що дозволяє автоматизовано управляти різними аспектами домашнього середовища.

3.5 Способи автоматизації дій користувача за допомогою штучного інтелекту

Пошукова оптимізація

Багато проблем AI можна вирішити розумним пошуком з багатьох можливих рішень, тобто результат може бути зведеним до пошуку. Наприклад, логічне твердження можна розглядати як пошук шляху, де на кожному кроці застосовуються певні умови. Алгоритми планування пошуку по деревах – це спроба знайти шлях до кінцевої мети. Алгоритми робототехніки для переміщення пристроїв і різних предметів використовують локальний пошук у просторі. Багато алгоритмів навчання в AI використовують алгоритми пошуку, які основані на оптимізації. Рішення для багатьох проблем полягає у використанні "евристики" або "великих принципів", які визначають пріоритетність вибору умови на користь тих, хто швидше досягає мети, і робить це за меншу кількість кроків. Класичні еволюційні алгоритми включають генетичні алгоритми, програмування експресії генів та генетичне програмування. Крім того, розподілені пошукові процеси можуть координуватись за допомогою алгоритмів розвідувальної групи.

Така пошукова оптимізація може бути застосована для пошуку найбільш повторювальних дій користувача у системі «розумного» будинку.

Логіка

Логіка використовується для представлення знань та вирішення проблем, але також може бути застосована і до інших аспектів. Наприклад, алгоритм satplan використовує логіку для планування, а індуктивне логічне програмування – метод для навчання.

У дослідженні AI використовується кілька різних форм логіки. Пропозиційна логіка передбачає функції істини, такі як "або" і "ні".

Нечітка логіка успішно використовується у системах управління, щоб дозволити експертам внести розпливчасті правила, такі як "якщо ви знаходитесь поруч із станцією призначення та рухаєтесь швидко, підвищуйте гальмівний тиск

поїзда". Ці нечіткі правила можуть бути чисельно уточнені в системі. Нечітка логіка погано масштабується в базу знань; багато дослідників AI ставлять під сумнів справедливість ланцюжків нечітко-логічних висновків.

Логіка за замовчуванням, немонотонна логіка та опис – це форми логіки, розроблені для допомоги у визначенні значень за замовчуванням та у проблемах з кваліфікацією. Кілька розширень логіки були розроблені для обробки конкретних областей знань, таких як: логіка опису, обчислення ситуації, обчислення подій і вільне числення (для відображення подій і часу, каузальне обчислення, обчислення віри (перегляд переконань) та модальна логіка). Також були розроблені логіки для моделювання суперечливих або непослідовних тверджень, що виникають у багатоагентних системах, таких як параконсистентна логіка.

Дана логістична функція найкраще підходить для застосування планування послідовності дій користувача у «розумному» будинку. Чи потрібно додавати у сценарій ту чи іншу дію користувача та або вимкнення/включення пристрою до певних дій.

Імовірнісні методи невизначених міркувань

Дослідники AI розробили ряд потужних інструментів для вирішення проблем неповної або невизначеної інформації за допомогою методів теорії ймовірностей та економіки.

Байєсівські мережі – загальний інструмент, який можна використовувати для різних проблем: міркування (за допомогою байєсівського алгоритму виведення), навчання (за допомогою алгоритму максимізації очікування), планування (використовуючи рішення мереж) та сприйняття (з використанням динамічних байєсівських мереж). Імовірнісні алгоритми можуть також використовуватися для фільтрації, прогнозування, згладжування та пошуку пояснень потоків даних, допомагаючи системам сприйняття аналізувати процеси, що відбуваються з часом (наприклад, приховані моделі Маркова або фільтри Калмана).

Такі системи можна застосувати для оцінки системи «розумного» будинку: наскільки вона надійна, безвідмовно працює та який має загальний стан системи.

Класифікатори та статистичні методи навчання

Найпростіші програми штучного інтелекту можна розділити на два типи: класифікатори та контролери. Але контролери також класифікують умови перед виведенням дій, і тому класифікація є центральною частиною багатьох систем AI. Класифікатори – це функції, які використовують відповідність шаблону для визначення найближчої відповідності. Їх можна нлаштувати за прикладами, що робить їх дуже гнучкими для використання в AI. Ці приклади відомі як спостереження або закономірності. При контрольованому навчанні кожен шаблон належить до певного заздалегідь визначеного класу, який можна розглядати як рішення, яке має бути прийняте. Всі спостереження в поєднанні з мітками класів відомі як набір даних. Коли отримано нове спостереження, це спостереження класифікується на основі попереднього досвіду.

Штучні нейронні мережі

Для використання штучного інтелекту у «розумному» будинку використовуються штучні нейронні мережі, які застосовуються у штучному інтелекті. Нейронна мережа – одна із реалізацій моделей глибокого навчання (рис.3.6).

Вхідний рівень приймає деяку вхідну інформацію. Тобто, якщо ми, наприклад, передаємо на нейронну мережу якусь картинку, то нам її потрібно примітизувати до розміру цього самого вхідного рівня. На прихованому рівні відбувається обчислення вхідної інформації. Ця інформація може стискатися або навпаки розтискатися і за рахунок вже існуючих там коефіцієнтів набувати певних змін або проходити певну обробку. І коли у нас вхідна інформація проходить через систему прихованих рівнів вона у нас якраз і набуває тих властивостей, які нам потрібні. Роль вихідного рівня полягає в тому, щоб він нам видавав оброблену інформацію.

Штучні нейронні мережі (ШНМ) відіграють важливу роль у розвитку систем «розумного» будинку, допомагаючи автоматизувати та оптимізувати різні аспекти побутового життя. Використовуються для створення систем розпізнавання голосу

та обличчя. Це дозволяє розпізнавати осіб та використовувати їхній голос для автоматизації різних функцій, таких як управління освітленням, системами опалення, кондиціонування та іншими. Можуть аналізувати дані про використання освітлення та електроприладів у будинку, розуміючи звички мешканців. Це дозволяє оптимізувати витрати енергії та автоматизувати управління освітленням та побутовими приладами.

ШНМ можуть аналізувати дані про енергоспоживання у будинку та передбачати оптимальний графік використання енергії для забезпечення ефективності та економії. Використовуються для реалізації систем відеоспостереження та аналізу відеопотоків. Вони можуть розпізнавати неправильні ситуації або підозрілі дії та сповіщати власників про це. Можуть аналізувати дані про температуру, погодні умови та звички мешканців для ефективного управління системами опалення та кондиціювання повітря.

Мережі даного типу використовуються для створення систем автоматичного управління водопостачанням та системами поливу, щоб ефективно використовувати водні ресурси. Аналізуючи дані про звички та привички мешканців, можуть прогнозувати їхні потреби та надавати рекомендації для оптимізації побутового середовища. *ШНМ виступають у ролі "мозку" систем «розумного» будинку, дозволяючи їм вчитися та адаптуватися до змін у середовищі та потребах користувачів.*

Оцінка прогресу

Ігри дають можливість побачити, як швидко розвивається прогрес. AlphaGo в 2016 року завершив еру класичних орієнтирів настільних ігор. Існує безліч змагань та призів, таких як Imagenet Challenge, щоб сприяти науковому дослідженню штучного інтелекту. Найбільш поширені сфери конкуренції включають загальну машинну розвідку, розмовну поведінку, обмін даними, роботизовані машини та футбол-робот, а також звичайні ігри. Ці сфери також можуть бути реалізовані у «розумному» будинку, наприклад, розмовна поведінка

між людьми у будинку може призвести до замовлення піци або іншої їжі, система може проаналізувати це та запропонувати замовити доставку їжі до будинку.

Розглянемо типи автоматизацій, які можуть бути використані для взаємодії системи «розумного» будинку та штучного інтелекту.

Застосування автоматизації має метою зменшення робочої сили та часу виконання певних задач. Автоматизація впровадила систему комп'ютерів та машин і замінила систему, яка була створена, де одночасно працюють людина і машина.

Існують різні види автоматизацій, які можуть бути використані до взаємодії «розумного» будинку та AI.

1) Числове управління. Свердла, 3D-друк, різання скла та ін. Підпадають під цю категорію пристрої, де машини запрограмовані на виконання повторюваних завдань.

2) CAM (Computer-aided manufacturing). Для цього прикладу автоматизації використовується комп'ютерне програмне забезпечення, схоже на автоматизований дизайн (CAD) на комп'ютерному рівні проєктування та складання.

3) FMS (Functional Movement Systems). Складна система автоматизації, де роботи і інші сучасні засоби автоматизації використовуються для забезпечення гнучкості та налаштування для користувачів.

4) Промисловий робот. Роботи використовуються для зварювання, складання та обробки матеріали, де роботів можна запрограмувати та маніпулювати у трьох і більше осях.

Для автоматизації дій користувача використовуються згорткові нейронні мережі, серед яких LSTM (Long Short-Term Memory) - найбільш потужна і добре відома серед інших функцій машинного навчання - тип штучної нейронної мережі, розробленої для розпізнавання шаблонів у послідовностях даних, таких як числові дані періодичного ряду, що надходять від датчиків, актуальність застосування якої доведена у системах «розумного» будинку.

3.6 Вибір засобів програмування

Для реалізації Smart Home та Artificial Intelligence необхідно створити web-додаток та серверну частину, яка матиме API та Artificial Intelligence, якому потрібно навчатися, обраховувати та надсилати користувачеві пропозиції щодо автоматизаційних процесів.

JavaScript

JavaScript широко використовується для розробки веб-додатків та веб-сайтів, які можуть інтегруватися з системами «розумного» будинку. JavaScript використовується для програмування підключених пристроїв у «розумному» будинку, таких як «розумні» лампи, термостати, датчики руху тощо. Зазвичай використовуються фреймворки, такі як Node.js для розробки серверної частини додатків для IoT. JavaScript може використовуватися для взаємодії з API «розумних» систем управління будинком. Наприклад, взаємодія з API платформи для роботи з даними сенсорів, змінною стану пристроїв або викликом відповідних команд. У цілому JavaScript використовується для створення користувацького досвіду, автоматизації та взаємодії в різних аспектах «розумного» будинку. Дана мова програмування використовується з технологіями HTML, CSS (рис.3.7).

Рисунок 3.7 – HTML, CSS, JavaScript

REST

Технологія REST (Representational State Transfer) використовується у «розумних» будинках для побудови API (інтерфейсу програмування застосунків),

який дозволяє різним пристроям та системам взаємодіяти та обмінюватися даними. REST використовує простий та легко зрозумілий інтерфейс, що робить його зручним для розробки та використання. Взаємодія між пристроями та системами стає більш доступною. Принцип розділення клієнтської та серверної частин, який підтримується REST, дозволяє «розумному» будинку мати гнучку архітектуру, де клієнтські та серверні компоненти можуть розвиватися незалежно один від одного. REST є універсальним інтерфейсом, що може бути використаний для взаємодії між різнорідними системами та пристроями. Це сприяє створенню сумісних та інтегрованих систем «розумного» будинку. REST використовує стандартні HTTP-методи (GET, POST, PUT, DELETE), що спрощує реалізацію та розуміння взаємодії між пристроями та серверами (табл.3.4).

Таблиця 3.4 – Методи HTTP

№ п/п	Назва методу	Призначення
1	GET	Отримати наявний ресурс
2	POST	Створити новий ресурс
3	PUT	Оновити існуючий ресурс
4	PATCH	Часткове оновлення існуючого ресурсу
5	DELETE	Видалити ресурс

React

Бібліотека React призначена для створення інтерактивних інтерфейсів користувача. React використовується для розробки односторінкових веб-додатків (SPA), де сторінка не перезавантажується при взаємодії користувача з додатком. Щодо системи «розумного» будинку, React має наступні застосування:

- дозволяє легко створювати інтерактивні та динамічні веб-інтерфейси для управління системою «розумного» будинку;
- працює на основі компонентів, які можуть бути перевикористані та розширювані. Це дозволяє створювати компактний та модульний код, що корисно

для розробки складних систем «розумного» будинку, які можуть містити багато різних компонентів та функцій;
- використовує концепцію «стану» (state), що дозволяє ефективно відстежувати та оновлювати дані в залежності від подій або змін. Це корисно для систем «розумного» будинку, де стан системи постійно може змінюватися;
- легко взаємодіє з зовнішніми API, що є важливим для отримання та відправки даних між веб-інтерфейсом та серверною частиною системи «розумного» будинку.

Таким чином, React служить потужним інструментом для створення ефективних та інтерактивних веб-інтерфейсів для систем «розумного» будинку, забезпечуючи гнучкість та модульність у розробці.

Node.js

Node.js може використовуватися для розробки серверної частини системи «розумного» будинку. Дозволяє створювати ефективні та масштабовані серверні додатки для обробки запитів від різних пристроїв у будинку. Node.js може використовуватися для взаємодії з різними пристроями у системі «розумного» будинку. Може взаємодіяти з сенсорами, пристроями IoT, «розумними» розетками та іншими пристроями через різні протоколи зв'язку, такі як MQTT або WebSocket. Node.js побудований на асинхронному програмуванні, що дозволяє ефективно використовувати ресурси сервера та підтримує велику кількість одночасних з'єднань, що важливо для систем «розумного» будинку з великою кількістю пристроїв та сенсорів.

Postgres

PostgreSQL - це система управління базами даних (СУБД), яка може мати різноманітні застосування у системах «розумного» будинку. Основні призначення PostgreSQL для систем «розумного» будинку включають: зберігання та управління даними; аналітику та запити; інтеграцію з іншими системами; підтримку геоданих; забезпечує можливість управління користувачами та

встановлення прав доступу до даних; масштабованість. PostgreSQL використовується у системах «розумного» будинку для забезпечення надійного зберігання та управління даними, забезпечення ефективної обробки інформації та сприяння інтеграції різноманітних пристроїв та систем.

Socket.io

Socket.io дозволяє встановлювати бідекілька зв'язків між клієнтом та сервером, і в обидва напрямки. Це робить можливим відправлення та отримання даних у реальному часі. У системі «розумного» будинку це може бути використано для негайного оновлення стану пристроїв, обміну даними між пристроями або побудови реактивних інтерфейсів. Використання Socket.io дозволяє легко синхронізувати стан системи між різними користувачами та пристроями. Це важливо в системі «розумного» будинку, щоб всі пристрої та інтерфейси відображали однаковий стан системи. Socket.io стає дуже корисним інструментом у випадках, коли потрібна швидка та ефективна взаємодія у реальному часі між різними компонентами системи «розумного» будинку.

Synaptic

Synaptic — це JavaScript-бібліотека для навчання штучних нейронних мереж. Її можна використовувати для розв'язання різних завдань у «розумному» будинку, таких як розпізнавання голосу, комп'ютерне зорове спостереження, прогнозування енергоспоживання та інше.

Homey Athom

Homey від Athom - це пристрій для створення системи «розумного» будинку, який об'єднує в собі різноманітні пристрої та технології для забезпечення зручності та автоматизації домашнього середовища. Homey спроектований для роботи з широким спектром пристроїв «розумного» будинку, таких як «розумне» освітлення, термостати, системи безпеки, аудіо- та відеосистеми, «розумні» пристрої для дому, тощо. Він може інтегрувати пристрої, які використовують різні технології зв'язку, такі як Zigbee, Z-Wave, Wi-Fi, Bluetooth і багато інших.

Homey служить центром управління для всіх підключених пристроїв. Користувачі можуть використовувати мобільний додаток або голосові команди для управління різними аспектами свого «розумного» будинку через Homey. Homey дозволяє користувачам створювати складні автоматизовані сценарії та правила. Наприклад, можна налаштувати автоматичне вимкнення світла, коли ми виходимо з дому, або розпочати відтворення музики, коли повертаємося.

Homey може бути інтегрований з різними голосовими асистентами, такими як Amazon Alexa або Google Assistant, що дозволяє нам управляти нашим «розумним» будинком за допомогою голосових команд. Homey має велику кількість додатків (аплікацій), які розширюють його функціоналість. Ці додатки надають підтримку для різноманітних пристроїв та сервісів. Athom регулярно випускає оновлення для Homey, що додають нові функції та покращують його продуктивність. Крім того, користувачі можуть розширювати можливості пристрою додаванням нових пристроїв та додатків.

Homey Athom служить як центральний елемент для інтеграції та управління різноманітними пристроями у «розумному» будинку, надаючи зручність та можливість автоматизації.

Загальне управління мережею ресурсів домашнього будинку здійснюється за допомогою смарт-шлюзу. Шлюз взаємодіє із зовнішніми системами, такими як хмарні сервіси та різними інтернет-сервісами. Для забезпечення мережних служб з необхідною користувачам якістю обслуговування користувачам смарт-шлюз повинен збирати велику кількість даних для аналізу, наприклад, у хмарному середовищі. Аналіз даних зосереджений на вимірюванні даних про якість обслуговування автоматизаційних процесів, безпеці та QoS (поведінці користувачів «розумного» будинку).

Рекомендовано використовувати архітектуру збору даних з HGU (смарт-шлюзу), передавання інформації між «розумними» будинками показано на рис.3.8. Дана архітектура використовується для створення екосистеми «розумних» будинків, якщо їх декілька. Home Gateway Unit або смарт-шлюз у

«розумному» будинку використовується як центральний вузол для збору даних від різних пристроїв та передавання цих даних до віддалених серверів або інших систем.

Архітектура складається з трьох шарів: інфраструктурного рівня «розумного» будинку, рівня смарт-шлюзу та рівня хмарного середовища.

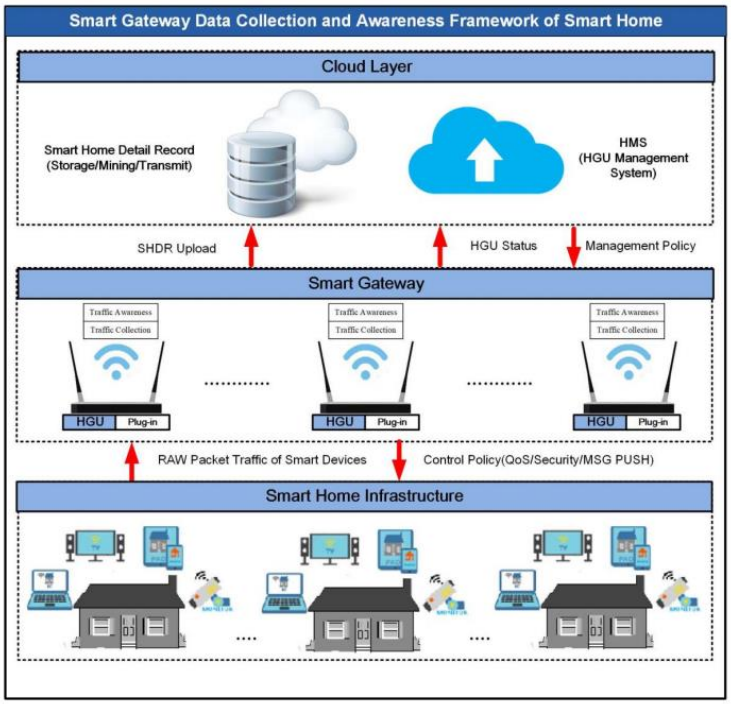


Рисунок 3.8 – Збір даних інтелектуального шлюзу та поінформованість для «розумних» домашніх мереж

Інфраструктурний рівень «розумного» будинку складається з «розумних пристроїв», таких як датчики, комп’ютери, домашні пристрої, IoT тощо. «Розумні» пристрої потребують доступу до зовнішньої мережі, тобто до мережі Інтернет, через смарт-шлюз.

Рівень смарт-шлюзу складається з HGU, який виконує основні функції збору даних та контролю ресурсів «розумного» будинку. Зокрема, прості фрагменти програмного модуля можуть бути реалізовані на рівні операційної системи.

Хмарний рівень в HGU забезпечує зв'язок та взаємодію із хмарними послугами та додатками. Хмарний рівень дозволяє власникам будинків віддалено отримувати доступ до системи «розумного» будинку через Інтернет. Це дозволяє

користувачам взаємодіяти з системою, навіть якщо вони знаходяться поза межами свого будинку. Хмарний рівень може забезпечувати зберігання та обробку даних, зібраних від датчиків та пристроїв у «розумному» будинку. Це дозволяє аналізувати та використовувати ці дані для отримання корисної інформації. Хмарний рівень в HGU сприяє покращенню ефективності, безпеки та функціональності систем «розумного» будинку, забезпечуючи одночасно доступ до різних хмарних ресурсів і послуг.

Реалізована архітектура програмного забезпечення для взаємодії пристроїв розумного «будинку» та збереження даних HGU зображена на рис.3.9. HGU забезпечує функціонування мережі для різних «розумних» пристроїв у домашній мережі. Він також підключається до зовнішньої мережі, тобто до Інтернету. Архітектура програмного забезпечення HGU включає такі частини: ОС HGU, основну платформу обслуговування, плагін збору трафіку, плагін MDA та інтерфейс звітів про дані.

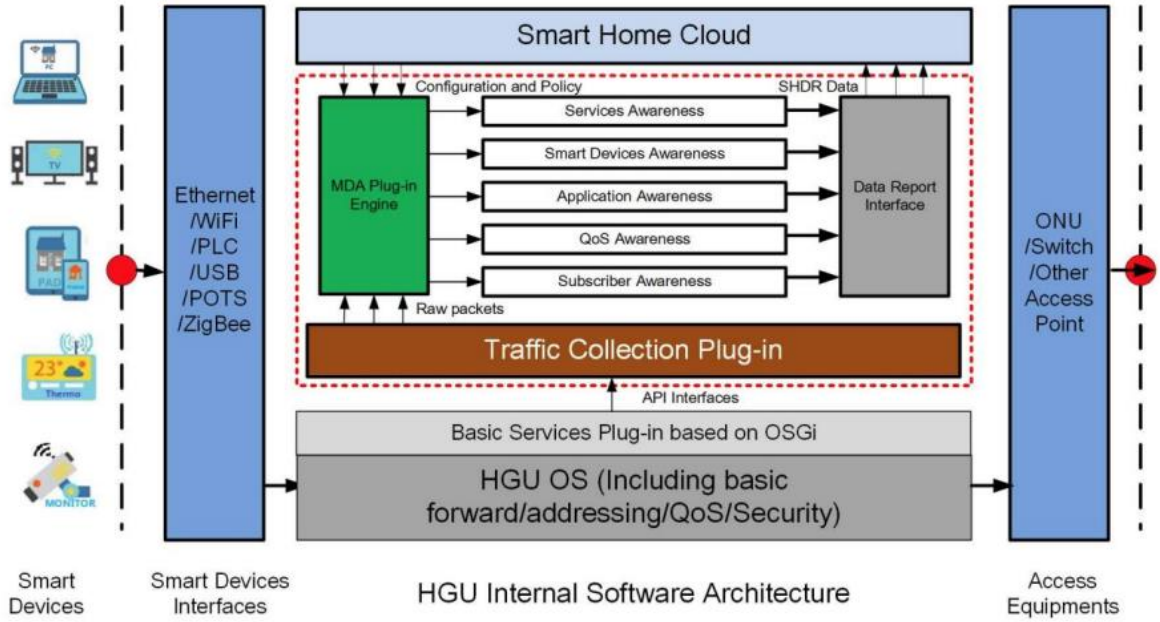


Рисунок 3.9 - Архітектура програмного забезпечення HGU

Проаналізуємо особливості архітектури збору даних та передавання інформації між «розумними» будинками через HGU.

1. Інтеграція різних пристроїв. HGU взаємодіє з різними «розумними» пристроями у будинку, такими як сенсори, камери, термостати, освітлення. Цей

шлюз повинен бути здатен взаємодіяти з пристроями різних виробників та використовувати різні технології зв'язку, такі як Zigbee, Z-Wave, Wi-Fi, Bluetooth.

2. Проміжний рівень обробки. HGU може виконувати проміжну обробку даних, використовуючи вбудовані алгоритми чи машинне навчання. Це дозволяє нам використовувати деякі локальні рішення, оптимізувати використання ресурсів та забезпечувати більш швидку реакцію на події в будинку.

3. Забезпечення безпеки даних. HGU може виконувати функції шифрування та забезпечення безпеки даних для захисту конфіденційності інформації, яку він передає між пристроями та іншими системами.

4. Передавання даних на віддалені сервери. HGU може взаємодіяти з віддаленими серверами або хмарними обчисленнями для збереження даних та отримання додаткових функцій, таких як аналітика, дистанційне управління та моніторинг.

5. Стандартизація протоколів. Для ефективної роботи з різноманітними пристроями HGU може використовувати стандартизовані протоколи комунікації, щоб забезпечити сумісність та інтеграцію.

6. Масштабованість. Архітектура повинна бути масштабованою, щоб легко взаємодіяти з різноманітними пристроями та обробляти збільшення обсягу даних з розвитком «розумного» будинку.

7. Функції маршрутизації та мережної організації. HGU може включати в себе функції маршрутизації та мережної організації для ефективного передавання даних між різними пристроями в межах «розумного» будинку.

У майбутньому HGU буде відігравати ключову роль у зборі, обробці та передаванні даних у системі «розумного» будинку, сприяючи її функціональності та взаємодії між пристроями.

Збір та аналіз даних на основі HGU здійснюватиме обробку та аналіз даних в хмарі для виявлення вузьких місць продуктивності мережі та налаштування мережевих служб, щоб підвищити QoS користувача. Процес збору та обробка даних показана на рис.3.10.

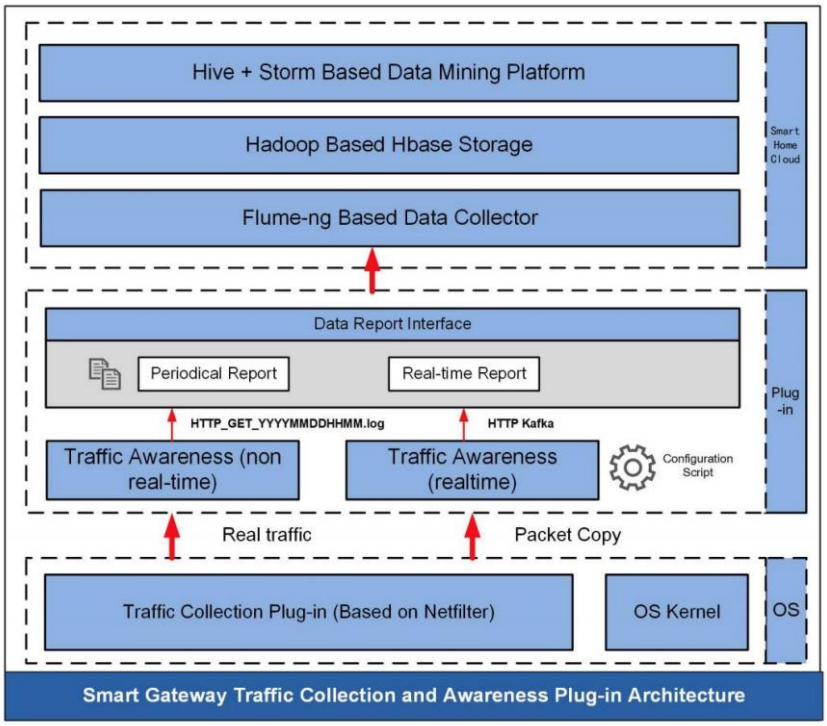


Рисунок 3.10 – Процес збору та обробки даних про трафік

Оскільки більшість програм у «розумних» будинках використовують HTTP як основний протокол, запити користувачів в основному здійснюються за допомогою запитів HTTP. Повідомлення HTTP-запита складається з двох типів: перший це – Get-запит, який може містити параметри для доступу до хмарних служб або ресурсів домашнього середовища, другий – Post-запит, що містить дані, які створив користувач. Інформація про повідомлення зберігається у форматі SHDR. Цей запис зазвичай містить інформацію про службу або тип програми, MAC-адресу, призначення, IP-адресу, довжину пакету, часову мітку прибуття пакету, HTTP-заголовок GET (а саме: URL, ім'я хоста, агент користувача) тощо. Плагін шлюзу спочатку запише SHDR у файл, а потім періодично завантажуватиме його у хмарне середовище.

Плагін для аналізу трафіка у режимі реального часу декодує повідомлення HTTP GET відповідно до специфікації протоколу HTTP після отримання необроблених пакетів із плагіну для трафіку на основі мережевого фільтра в ядрі ОС. Потім він витягує інформацію з ключа запиту HTTP GET. Відфільтровані

дані будуть повідомлятися через інтерфейс звіту даних. Процес фільтрації забезпечує ефективність та точність даних у хмарі.

Плагін для аналізу про рух у реальному часі працює по-іншому. Плагін розшифровує HTTP GET повідомлення, відправляє дані до хмарного середовища в режимі реального часу. Наприклад, інформація про дим повинна повідомлятися у режимі реального часу користувачу.

Впровадження технології штучного інтелекту у «розумний» будинок призводить до численних переваг, які покращують комфорт, ефективність та безпеку життя мешканців. Штучний інтелект надає можливість легко додавати нові функціональності та пристрої до системи «розумного» будинку. Він є модульним і може адаптуватися до різних потреб та змін у використанні приміщення. Застосування штучного дозволяє мешканцям віддалено управляти різними функціями «розумного» будинку через мобільні додатки або віддалені платформи. Штучний інтелект може оптимізувати використання різноманітних систем «розумного» будинку, забезпечуючи більш ефективну роботу та менше споживання ресурсів. Впровадження технології штучного інтелекту у «розумний» будинок призводить до вдосконалення функціональності, зручності та безпеки, забезпечуючи мешканцям більше інтелектуальних можливостей та комфорту. За допомогою вбудованого програмного забезпечення у смарт-шлюз можна досягти ефективності збору даних, інформативності та звітування.

ВИСНОВКИ

Побудова Smart City включає в себе використання різноманітних технологій для оптимізації міських систем та поліпшення якості життя мешканців. Ці технології повинні працювати спільно для створення інтелектуального та ефективного міста, яке відповідатиме потребам мешканців та надавати їм зручність, оптимізуватиме використання ресурсів та сприятиме сталому розвитку. Побудова Smart City вимагає комплексного підходу до інтеграції цих технологій для створення міст, які у майбутньому будуть не лише технологічно передовими, але й забезпечуватимуть підвищену якості життя громадян.

Технології Інтернету речей (IoT) відіграють важливу роль у розвитку «розумних» міст, де різноманітні пристрої і об'єкти пов'язуються мережею для збору, обміну та аналізу даних з метою оптимізації функціонування міської інфраструктури та покращення якості життя громадян. Ці технології спільно використовують дані та забезпечують інтеграцію, допомагають зробити міста більш ефективними, екологічно чистими та зручними для мешканців, а також сприяють збереженню ресурсів та підвищенню якості життя. Ключовими технології IoT, які інтегруються у Smart City є системи збору та аналізу даних, а саме: централізовані системи управління - моніторинг та аналіз даних для прийняття рішень щодо розвитку міста; аналітика для прийняття рішень - використання штучного інтелекту та аналітичних інструментів для оптимізації управління містом.

Технології штучного інтелекту спільно використовують дані для прийняття інформованих рішень, підвищення ефективності та поліпшення якості життя в містах. Штучний інтелект забезпечує можливість збору, аналізу та обробки великих обсягів даних для прийняття інформованих рішень щодо розвитку міста, планування інфраструктури та вдосконалення послуг для мешканців.

IoT-технології та технології штучного інтелекту інтегруються у Smart City для оптимізації різних міських систем та поліпшення якості життя. Ці технології співпрацюють у Smart City для збору та аналізу даних; оптимізації транспортних

систем; ефективного управління енергетикою; систем управління водозабезпеченням; громадської безпеки; системи управління відходами.

IoT-технології та технології AI повинні працювати спільно для створення інтелектуального та ефективного міста, яке відповідатиме потребам мешканців та надавати їм зручність, оптимізуватиме використання ресурсів та сприятиме сталому розвитку. Побудова Smart City вимагає комплексного підходу до інтеграції цих технологій для створення міст, які у майбутньому будуть не лише технологічно передовими, але й забезпечуватимуть підвищену якості життя громадян.

Технологія штучного інтелекту допомагає «розумним» будинкам в управлінні пристроями, управлінні енергією, охороною здоров'я, інтелектуальній взаємодії, безпеці, розважальних системах та персональних роботах, використовуючи розпізнавання діяльності, обробку даних, прийняття рішень, розпізнавання зображень, прогнозування та розпізнавання голосу.

Для інтеграції ресурсів в єдину систему доцільно використовувати такі інструменти як універсальний контролер Homey Athom та інші різні ресурси, які вимірюють, аналізують та обробляють дані про навколишнє середовище. LSTM є популярним вибором для обробки послідовностей даних через їхню здатність ефективно управляти довготерміновою та короткотерміновою інформацією. На основі проведеного аналізу показано, що для реалізації єдиної екосистеми «розумного» будинку з використанням штучного інтелекту найбільш доцільним є використання наступних інструментів: HTML/CSS; JavaScript/TypeScript; Python; React; Node.js; Postgres; REST API; Socket.io; Synaptic; LSTM; Homey Athom.

Athom зможе зробити будинок зручним через те, що більше не потрібно використовувати різні хаби. Homey з'єднує всі домашні пристрої, навіть різних виробників і брендів. Він – мозковий центр «розумного» будинку. Athom реалізує можливість голосового управління, потрібно тільки зв'язати потрібне обладнання з контролером і налаштувати правила.

Впровадження технологій штучного інтелекту у «розумні» міста має безліч переваг, оскільки це дозволяє ефективніше управляти ресурсами, забезпечує

безпеку та комфорт мешканців і взагалі сприяє сталому розвитку. Технології штучного інтелекту можуть стати ключовим елементом для створення більш ефективних, безпечних та стало розвинених «розумних» міст.

В цілому, технології повинні працювати разом, використовуючи дані, щоб «розумно» управляти різними аспектами міського життя, підвищувати ефективність та забезпечувати більш зручні умови для мешканців.

ПЕРЕЛІК ПОСИЛАНЬ

1. http://leetcode.com/problemset/all/

2. http://perso.limsi.fr/pointal/

3. https://numpy.org/

4. W. Yu, W. Cheng, C. Aggarwal, K. Zhang, H. Chen, and Wei Wang. NetWalk: A flexible deep embedding approach for anomaly Detection in dynamic networks, ACM KDD Conference, 2018.

5. Lu, Y., Papagiannidis, S. and Alamanos, E. (2018). Internet of things: A systematic review of the business literature from the user and organisational perspectives. Technological Forecasting and Social Change 136: 285–297.

6. Liyanage, M., Ahmad, I., Abro, A.B. et al. (2018). A Comprehensive Guide to 5G Security. New York: John Wiley & Sons.

7. Zaidan, A.A., Zaidan, B.B., Qahtan, M. et al. (2018). A survey on communication components for iot-based technologies in smart homes. Telecommunication Systems 69 (1): 1–25.

8. Adapa, S. (2018). Indian smart cities and cleaner production initiatives– integrated framework and recommendations. Journal of Cleaner Production 172: 3351–3366.

9. Alavi, A.H., Jiao, P., Buttlar, W.G. and Lajnef, N. (2018). Internet of things-enabled smart cities: State-of-the-art and future trends. Measurement 129: 589–606.

10. Reka, S.S. and Dragicevic, T. (2018). Future effectual role of energy delivery: A comprehensive review of internet of things and smart grid. Renewable and Sustainable Energy Reviews 91: 90–108.

11. Sun, H., Zhang, Z., Hu, R.Q. and Qian, Y. (2018). Wearable communications in 5g: challenges and enabling technologies. IEEE Vehicular Technology Magazine 13 (3): 100–109.

12. Porambage, P., Manzoor, A., Liyanage, M., et al. (2019). Managing mobile relays for secure e2e connectivity of low-power IoT devices. IEEE Consumer Communications & Networking Conference, 2019, Las Vegas, USA (11–14 January 2019). IEEE.

13. Munster, G. and Bohlig, A. Auto Outlook 2040: The Rise of Fully Autonomous Vehicles. Loupventures. Режим доступу до ресурсу: https://loupventures.com/auto-outlook-2040- the-rise-of-fully-autonomousvehicles/ (accessed 16 July 2019).

14. LoRAWAN specifications, LoRa Alliance Technology. Режим доступу до ресурсу: https://lora-alliance.org/resource-hub/lorawanrspecification-v11 (accessed 16 July 2019).

15. Shin, S.-W., Crawford, M., and Mellor, S. (eds) (2017). The Industrial Internet of Things Volume G1: Reference Architecture. 1–58. Industrial Internet Consortium (IIC). Режим доступу до ресурсу: https://www.iiconsortium.org/IIC_PUB_G1_V1.80_2017-01-31.pdf (accessed 16 July 2019).

16. Jake VanderPlas Python Data Science Handbook: Essential Tools for Working with Data, 2022, 551p.

17. Heller, Nicholas et al. (2020). The KiTS19 Challenge Data: 300 Kidney Tumor Cases with Clinical Context, CT Semantic Segmentations, and Surgical Outcomes. arXiv: 1904. 00445 [q-bio.QM].

18. Hollo, Kaspar (2019). Exploring the Value of Weakly-Supervised Deep Learning Approaches for Artefact Segmentation in Brightfield Microscopy Images. URL: https://comserv. cs . ut . ee / home / files / hollo _ softwareengineering _ 2021.

19. Wang, Haofan et al. (2020). Score-CAM: Score-Weighted Visual Explanations for Convo-lutional Neural Networks. arXiv: 1910.01279 [cs.CV].

20. Yang, Guanyu et al. (2020). “Weakly-supervised convolutional neural networks ofrenal tumor segmentation in abdominal CTA images”. In: BMC Medical Imaging 20.1, p. 37. ISSN: 1471-2342. DOI: 10.1186/s12880- 020- 00435- w. URL: https://doi.org/10.1186/s12880-020-00435-w.

21. Selvaraju, Ramprasaath R. et al. (2019). “Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization”. In: International Journal of ComputerVision 128.2, pp. 336–359. DOI: 10 . 1007 / s11263 - 019 - 01228 - 7. URL: https://doi.org/10.1007\%2Fs11263-019-01228-7.

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

(Презентація)