ТИТУЛЬНИЙ ЛИСТ

ЗМІСТ

ВСТУП

Інформаційні технології активно використовуються в метеорології і роботі синоптиків, так як дозволяють не тільки автоматизувати їх роботу, а також підвищити зручність її виконання і значно підвищити точність результату.

Метеорологія - це наука, яка вивчає процеси, явища, що відбуваються в атмосфері.

Сучасна метеорологія не змогла б розвинутися без впровадження в неї інформаційних технологи і особливо суперкомп'ютерів, пов'язано це з тим що робота синоптиків, зокрема побудова прогнозу погоди, вимагає великих як тимчасових так і людських ресурсів. Починаючи від простого побудови прогнозу погоди закінчуючи попередженням про насування погодних катастроф, які могли б привести до великих людських жертв і масштабних руйнувань.

Використання обчислювальної техніки і відповідних програм дозволяє вирішувати завдання метеорології з великою точністю, для значно більших територій, при цьому за більш короткий термін і з залученням менших людських ресурсів. У зв'язку з цим дана галузь вкрай сильно потребує розробки зручних і високопродуктивних програмних засобах.

Тема більш ніж актуальна так як розвинена метеорологія дозволить поліпшити важливі аспекти життя людей, наприклад: збільшити ефективність сільськогосподарської галузі, дозволить визначати безпечні для життя людей місця, прокладати більш оптимальні маршрути кораблям і літакам, мінімізувати наслідки природних катастроф.

Зв'язок роботи з програмами наукових досліджень кафедри ПІ. Для поліпшення загальних процесів автоматизації та створення єдиного інформаційного простору для сфери метеорології повинні використовуватися сучасні інформаційні технології, які дозволяють розробити сучасний

програмний засіб, здатний виконати поставлене завдання за менший проміжок часу і забезпечити більш високу точність, ніж людина.

Щодо алгоритмів застосовуваних у цій сфері можна виділити проблеми складності алгоритму, довгого часу його виконання, а також гнучкості алгоритму. Складність алгоритму виявляється у тому що сформована математична модель велика і має оптимальну складність алгоритму виконання. Проблема гнучкості виникає внаслідок того, що складена математична модель пов'язана на певних константах прив'язаних до конкретної території або регіону.

Використання нейронних мереж може вирішити обидві ці проблеми, нейронна мережа простіше у використанні, має універсальний інтерфейс отримання вхідних даних і легко адаптується до змін завдяки своїй структурі. У результаті можна отримати універсальний алгоритм, який зможе обробляти дані з будь-яких регіонів.

Розширення погляду залежність набору вхідних даних. Поточні алгоритми, що використовуються в метеорології, використовують для своєї роботи набір вхідних параметрів, що складається з температури повітря, його вологості, напряму вітру та його швидкості. Ці параметри просто збирати, але аналіз погоди на їх підставі може мати серйозну похибку через "прогалини" в моделі. В даному випадку під “пробілами” маються на увазі метеорологічні змінні, які не враховуються в даній моделі, але можуть значно впливати на результат. Нейронні мережі дозволяють легко оцінити значущість своїх вхідних параметрів на результат і могли б допомогти у вирішенні цього завдання у сфері метеорології.

Мета і задачі дослідження. Метою роботи є дослідження ефективності роботи різних варіантів нейронних мереж на задачах обробки та аналізу метеоданих. Для виконання поставленої задачі, необхідно виконати:

– аналіз особливостей метеорологічних задач;
– аналіз специфіки машинного навчання;

– аналіз можливостей сучасних засобів розрабки програмного забезпечення з використанням алгоритмів машинного навчання;

– постановка завдання регресії;

– порівняльний аналіз та вибір алгоритму регресії;

– розробка концепції попередньої обробки та аналізу даних.

Об’єкт дослідження. Об'єктом дослідження є різні варіації нейромереж та набори вхідних параметрів для них.

Методи дослідження. В якості наукової основи дослідження використовують: алгоритми машинного навчання, методи оцінки якості моделей прогнозування та класифікації даних.

1 АНАЛІЗ ПРЕДМЕТНОЇ ТЕМАТИКИ ДОСЛІДЖЕННЯ

1.1 Аналіз особливостей метеорологічних задач

Метеорологія є комплексом наук все атмосферні явища, такі як: погода, клімат, хмари, склад атмосфери, її будова, тепловий режим і влагообмін, оптичні та акустичні, поведінка фронтів, вітру і електричного поля. Всі ці дослідження допомагають знайти і зрозуміти фізичні залежності які відбуваються в атмосфері, що в кінцевому рахунку наблизить людство до розуміння життєвих циклів планети.

Сучасних метеорологів можна розділити по виду діяльності на наступні категорії або наукові напрямки [1]:

– Фізична метеорологія – займаються розробкою радіолокаційних і космічних методів дослідження атмосферних явищ.

– Динамічна метеорологія – вивчення фізичних механізмів атмосферних процесів.

– Синоптична метеорологія – наука про закономірності зміни погоди.

– Кліматологія – наука вивчаюча клімат як сукупність погодних характеристик за багаторічний період і їх зміна властивих певної території.

– Аерологія – наука вивчає верхні шари атмосфери для декількох десятків кілометрів від поверхні землі.

Прикладні напрямки сучасної метеорології:

– Авіаційна метеорологія – наука вивчає вплив погоди на діяльність авіації.

– Агрометеорологія – наука яка вивчає вплив погоди на сільське господарство.

– Біометеорологія – наука вивчає вплив атмосферних процесів на людину та інші живі організми.

– Ядерна метеорологія – наука вивчає природну і штучну радіоактивність, поширення в атмосфері радіоактивних домішок, вплив ядерних вибухів.

– Радіометеорологія – наука вивчає поширення радіохвиль в атмосфері.

Якщо узагальнити те кожен розділ займається пошуком деяких погодних залежностей і вивчення впливу погодних явище на деяку галузь. Для виконання цієї роботи необхідно взаємодіяти злагоджено і задіяти безліч фахівців по всьому світу, які кожні три години збирають інформацію про погоду зі всіх доступних джерел, далі проводять їх обробку і будують різні карти описують графічним методом поточний погодний стан атмосфери в різних регіонах. Далі за отриманими картками проводиться аналіз виявляє необхідну інформацію, наприклад це може бути аналіз погодних аномалій для розуміння того чи можуть вони перерости в природні катастрофи і якщо це так варто попередити населення цих регіонів.

Метеорологічні данні які найчастіше збирають для досліджень [3]:

– Місце та час коли були зняті метеодані, базова інформація яка потрібна для прив’язки метеоданих на карті.

– Висота хмар від рівня моря.

– Дальність видимості. Кількість метрів далі котрих не можна побачити наземні об’єкти.

– Вид хмар, якій переважає в даному секторі.

– Напрям вітру.

– Швидкість вітру.

– Температура повітря в час збору даних.

– Точка роси, це температура повітря при якій починає випадати роса.

– Атмосферний тиск на рівні метеостанції.

– Атмосферний тиск на рівні моря.

– Максимальна та мінімальна температура за добу.

– Погодне явище.

– Кількість опадів.

На базі цих метеоданих будуються прогнози погоди та більшості з усіх метеорологічних розрахунків.

Основою для прогнозу погоди є облік періодичних і неперіодичних змін метеорологічних величин і явищ погоди [4]. Періодичні зміни тієї чи іншої метеорологічної величини обумовлюються добовим і річним ходом цієї величини, неперіодичні – еволюцією і переміщенням синоптичних об'єктів: циклонів і антициклонів, повітряних мас і атмосферних фронтів. Ось чому прогнозом погоди завжди передує прогноз синоптичного положення. Найбільшу трудність і практичний інтерес представляє облік імені не періодичних змін.

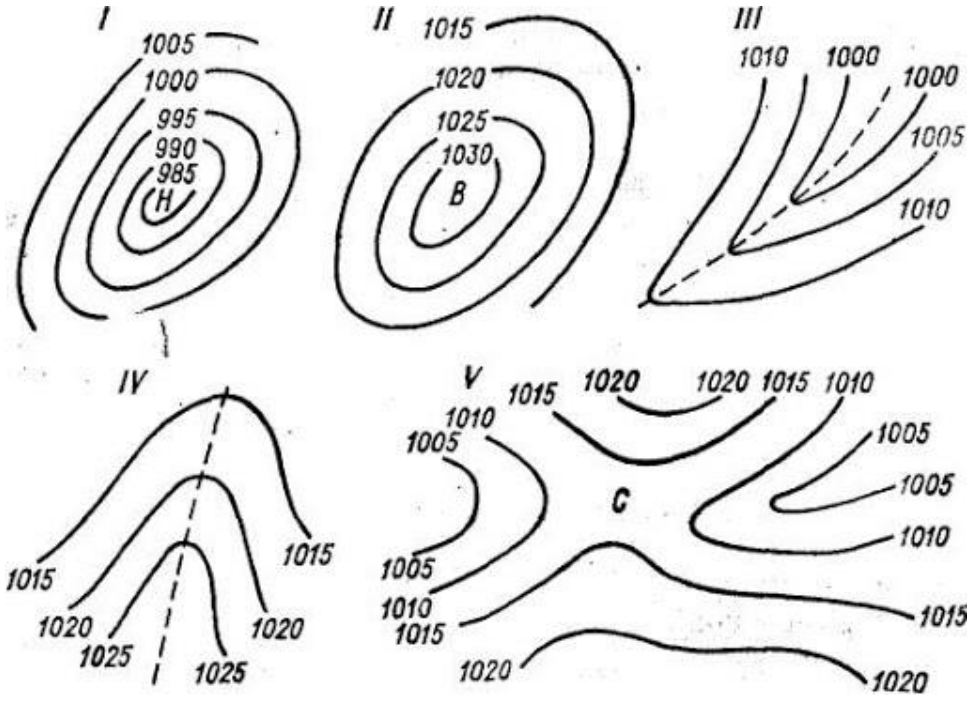

Рисунок 1.1 – Приклад відображення баричних систем на картах.

Синоптичний метод в даний час є основним при розробці довгострокового прогнозу погоди.

Суть методу в тому, що на підставі аналізу карт погоди за кілька послідовних термінів складають прогноз синоптичного положення, який заклю чає в прогнозі виникнення, переміщення і еволюції повітряних мас, атмосферних фронтів, баричних систем, приклад відображення баричних систем на картах рис 1.1.

Карта, на яку наносять передбачуване положення синоптичних об'єктів, називається прогностичною, рис 1.2.

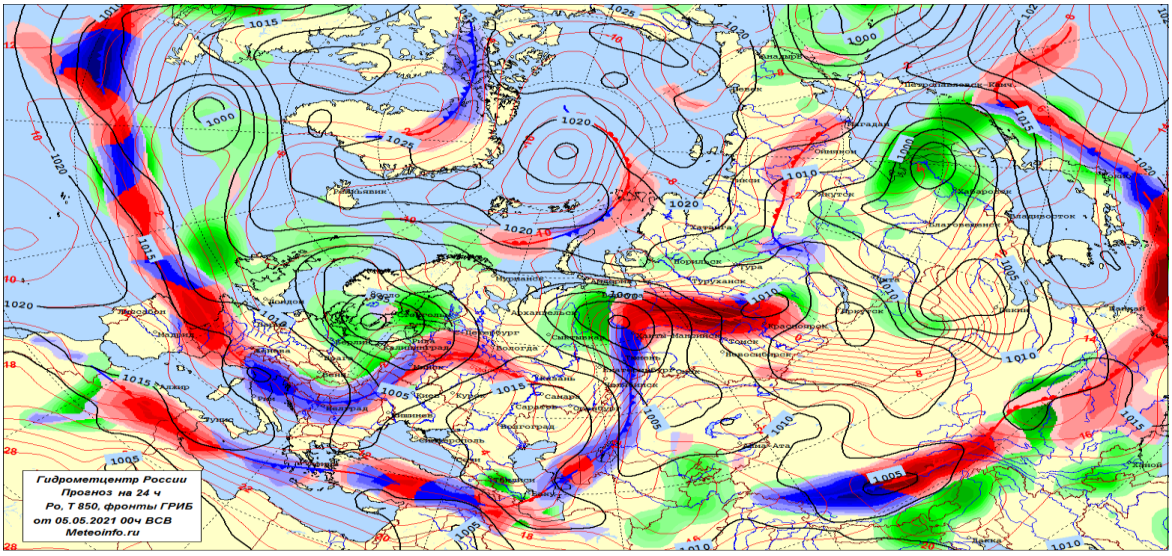

Рисунок 1.2 – Приклад прогностичної метеорологічної карти

Прогностичну карту складають на певний момент часу наступної доби, а іноді на двоє або більше доби вперед.

На підставі прогностичної карти складають прогноз погоди в тому чи іншому районі [4].

Причому прогноз погоди є логічним продовженням прогнозу синоптичного положення і виходить з основного принципу, що полягає в припущенні, що з переміщенням і еволюцією синоптичних об'єктів переносяться з певними змінами і властиві їм умови погоди.

Тому за прогностичне значення метеорологічної величини в першому наближенні приймаються їх значення в районі, звідки очікується переміщення

синоптичного об'єкту, в район, для якого складається прогноз погоди. При такому великому обсязі роботи та даних необхідних до зберігання дана галузь не може обійтися без використання інформаційних технологій. Особливо новітніх програмних засобів, які би спростили роботу з розрахунками та потужних комп’ютерів, які би ці алгоритми оброблювали.

Інформаційні технології щільно увійшли і намертво закріпилися в житті сучасних метеорологів.

У своїй щоденній роботі їм необхідно використовувати масивні програмні комплекси які виконують завдання збору, зберігання і перегляду інформації з джерел метеоданих, програми для побудови погодних карт, для визначення і відстеження баричних систем і атмосферних фронтів, побудови прогностичних карт, виявлення погодних аномалій, аналізу погодних даних, побудови графіків залежностей, отримання статистики і графічного відображення метеоданих, а також прогнозування катастроф і інші.

Вплив інформаційних технологій на галузь досить велике, так як без них виконання більшості завдань сучасної метеорології було б неможливим. Якби метеорологи використовували старі або "класичні" методи, то всьому людству довелося б освоїти цю професію і спільно виконувати її завдання. Сам процес збору даних залучає більше кількість людей, а її аналіз зажадав би все доступне в добі час.

Для вирішення та прискорення цих процесів людство прийшло до використання суперкомп'ютерів [5].

Суперкомп'ютером називається комплекс електронних обчислювальних машин працюють як одне ціле, що дає всій системі велику обчислювальну потужність.

Однак навіть суперкомп'ютерів необхідно достатньо часу для розрахунків еволюції баричних систем і прогнозу погоди.

Наприклад, для розрахунку короткострокового прогнозу погоди для всієї території України, з точністю в п'ять квадратних кілометрів, середньостатистичному суперкомп'ютера знадобиться близько години.

На даний момент процес побудови прогнозу запускається не частіше ніж раз на три години, цього цілком достатньо для більшості територій.

Винятками ж є території з високою ймовірністю появи атмосферних катаклізмів. Як приклад таких областей можна привести

Японію, з їх штормами і цунамі, південні штати Америки, з місцевими повенями і торнадо і т.д.

З кожним днем необхідних метеорологічних розрахунків стає все більше, через те що все більше галузей виявилися залежні від інформації про стан атмосфери або про її вплив на свою галузь і отже отримали необхідність в аналізі метеоданих.

Завдяки цій тенденції сильно збільшилося фінансування метеорології що в кінцевому рахунку дозволило метеорологам почати безліч досліджень, які в майбутньому допоможуть краще розуміти процеси атмосфери і їх вплив на наше життя.

Основна частина фінансової підтримки необхідна для придбання обладнання та розробки програмного забезпечення для всіх галузей даної сфери [5].

При активному розвитку інформаційних технологій в сфері метеорології для людства можуть відкритися можливості:

– Ідеальна організація і оптимальний розподіл вирощуваних культур. При точному знанні того які метео-параметри були раніше і які будуть в кожній точці світу люди зможуть розподілити висаджуються культури ідеально по необхідним умовам. Дане нововведення дозволить збільшити обсяги вирощуваних культур, але також пристосуватися до змін клімату і вчасно реагувати на випадки коли земля стає менш придатною для вирощування конкретної культури.

– Оптимізація і полегшення маршрутів суден і літаків. Дане нововведення дозволяє запобігти небезпеці в дорозі і потенційно врятували б

життя людей, мінімізували б витрати на ремонтні роботи транспорту, а також не допустити забруднення природи від можливих аварій.

– Можливість побудови різних архітектурних споруд. Для побудови різних споруд важливо знати які погодні умови переважають в районі будівлі щоб зрозуміти як сильно треба зміцнити конструкцію будівлі для її стійкості.

– Ремонтні роботи. Частково відноситься в попередньому пункту, для деяких ремонтних робіт важливі певні погодні умови з чим і можуть допомогти нові системні комплекси метеорологів, точно вибравши проміжок часу з ідеальною погодою для ремонтних робіт.

Не менш значущий вплив інформаційні технології привнесли в системи для вимірювання метеоданих.

Раніше використовувані аналогові прилади, хоч і відрізнялися дешевизною і примітивністю в роботі з ними, проте були недостатньо точними і досить громіздкими [5].

Рисунок 1.3 – Схема сучасної метеостанції

Сучасні прилади компактні, рис 1.3, мають можливість передавати результати вимірювань по лініях зв'язку і при цьому мають велику точність. Це дозволяє прискорити процес збору даних, а їх автономність дає можливість вільно масштабувати мережу метеостанцій, рис 1.4.

Подібні системи мають основний стовп на якому закріплені різні датчики, наприклад датчики напряму вітру, кількості опадів, атмосферного тиску та різні датчики для аналізу повітря.

Система автономного живлення представляє собою джерело енергії, систему стабілізації енергії та систему накопичування енергії.

Як джерело енергії може бути сонячні панелі чи вітряки. Система накопичування енергії це акумуляторна батарея яка буди живити станцію в час коли енергії системи живлення не буде достатньо енергії [5].

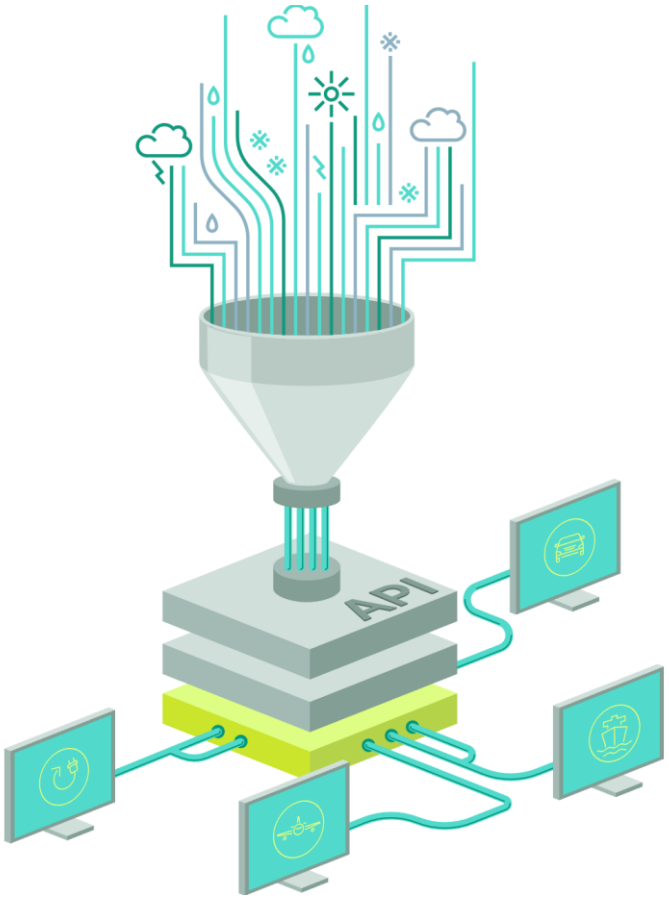

Рисунок 1.4. – Приклад схеми сучасної логіки роботи метеокомплексів

1.2 Аналіз специфіки машинного навчання

Проаналізувавши завдання метеорології можна дійти висновку, що в ній можна застосувати майже всі види та варіації нейромереж та алгоритмів машинного навчання.. Щодо алгоритмів застосовуваних у цій сфері можна виділити проблеми складності алгоритму, довгого часу його виконання, а також гнучкості алгоритму. Складність алгоритму виявляється у тому що сформована математична модель велика і має оптимальну складність алгоритму виконання. Проблема гнучкості виникає внаслідок того, що складена математична модель пов'язана на певних константах прив'язаних до конкретної території або регіону.

Використання нейронних мереж може вирішити обидві ці проблеми, нейронна мережа простіше у використанні, має універсальний інтерфейс отримання вхідних даних і легко адаптується до змін завдяки своїй структурі. У результаті можна отримати універсальний алгоритм, який зможе обробляти дані з будь-яких регіонів.

Розширення погляду на залежність набору вхідних даних. Поточні алгоритми, що використовуються в метеорології, використовують для своєї роботи набір вхідних параметрів, що складається з температури повітря, його вологості, напряму вітру та його швидкості. Ці параметри просто збирати, але аналіз погоди на їх підставі може мати серйозну похибку через "прогалини" в моделі. В даному випадку під “пробілами” маються на увазі метеорологічні змінні, які не враховуються в даній моделі, але можуть значно впливати на результат. Нейронні мережі дозволяють легко оцінити значущість своїх вхідних параметрів на результат і могли б допомогти у вирішенні цього завдання у сфері метеорології [5].

Починаючи від найскладнішого і найкомплекснішого штучного інтелекту, для обробки загальної картини поведінки метеорологічних об'єктів, до простих математичних моделей для аналізу даних конкретного датчика на вузькому участі землі. Для завдань прогнозування використовуються відповідні нейронні мережі, навчені на сотнях тисяч записів метеоданих з різних точок світу, завдання визначення погодних явищ вирішуються алгоритмами класифікації. Також значний вплив на результат надає процес підступу та аналізу аномалій. Аномаліями називають будь-яке значне відхилення в даних, їх поява може будувати висновки про різного роду сторонньому вплив на метеоситуацію в місці їх збору. У цьому контексті, в першу чергу варто уточнити значущість і види сил, які можуть мати вплив.

Безліч різних факторів може впливати на метеорологічну ситуацію, наприклад, це можуть бути різні природні катастрофи, наприклад, потопи і пожежі або техногенні катастрофи у вигляді тих же пожеж або витоків різних речовин. Подібні явища безпосередньо впливають на метеоситуацію в певній

області, безпосередньо впливає на температуру, швидкість і напрям вітру, тиск, вологість. Залежно від масштабів це може вплинути і на загальну метеорологічну картину у світі [6]. Приклад метеостанції домашнього типу наведено на рис.1.5.

Рисунок 1.5 – Приклад метеостанції домашнього типу

Якщо природні катастрофи відрізнити досить просто, то техногенні можна виділити тільки завдяки датчикам аналізу склад повітря і різні домішки в ньому. Подібні станції сильно відрізняються від професійних аналогів але можуть збирати дані про температуру, вологість повітря і атмосферний тиск, рис 1.5. Деякі моделі дозволяють аналізувати якість повітря, конкретно його склад і кількість домішок. Питання відстеження та аналізу стану повітря не менш важливий для метеорології. У повітрі можуть міститися різні домішки які можуть той чи інший ефект на погоду в регіоні [6]. Таким чином вивчаючи склад повітря можна в майбутньому передбачити згубні явища в регіоні по зміні складу його повітря і навпаки, припустити склад повітря через погодні явища в регіоні, рис 1.6.

Рисунок 1.6 – Карта забруднення повітря і ґрунту України

Факторами зміни можуть бути, як досить очевидні, наприклад заводи або аварії, так і менш очевидні і внаслідок чого більш небезпечні, наприклад розкриття покладів метану внаслідок землетрусів. Аналіз складу повітря більш актуальний для другого випадку, конкретно для систем запобігання катастроф.

Система вивчає аномалії в повітрі і може виявити виниклі проблеми, після чого попередити про це відповідні служби. Існують приклади місць які вимагають постійного моніторингу ситуації, наприклад озера в парку Йеллоустоун, які є потенційно небезпечною зоною і можуть завдати серйозної шкоди регіону [6]. Також подібні системи фахівці можуть відслідковувати так звані «мертві зони» в морях і океанах, це зони в яких сталася якась аномалія в слідстві якої живі організми в ній вимерли або покинули її, рис. 1.7. Найчастіше подібний ефект викликаний забрудненнями вод і подібні місця відслідковуються за складом повітря.

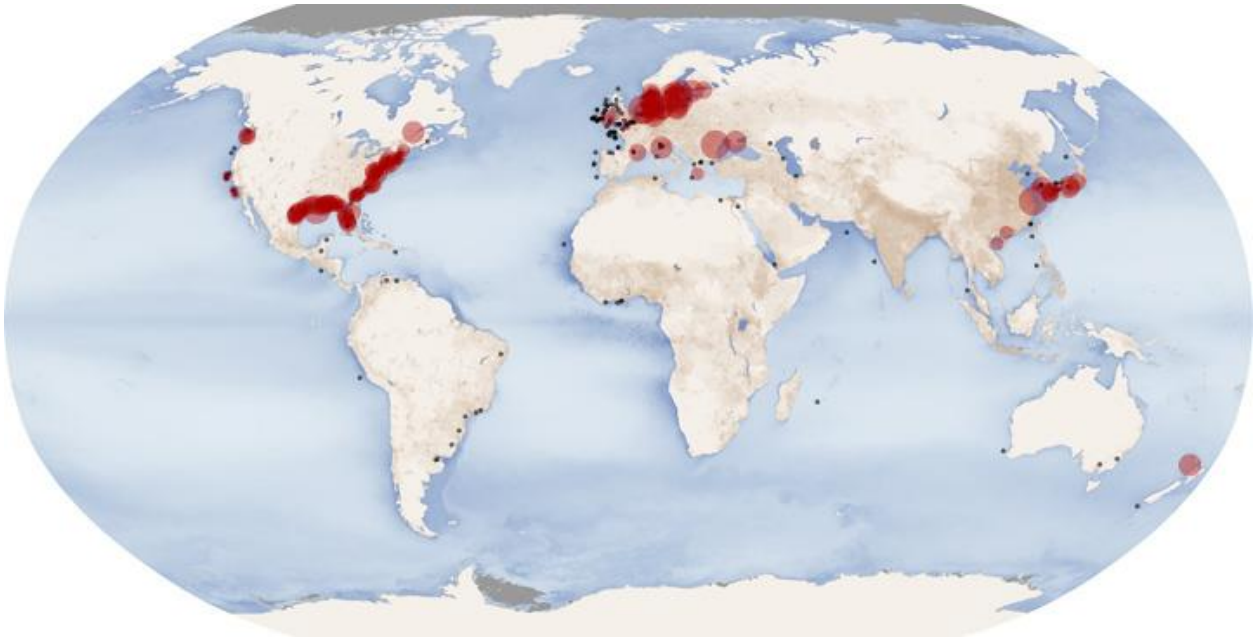

Рисунок 1.7 – Карта світового океану із зазначеними на ній мертвими зонами

Все це створює велику кількість параметрів та їх комбінацій, які треба враховувати при аналізі метеоданих, а про багатьох важливих метеорологічних змінних можуть навіть не підозрювати. У вирішенні цих завдань використання нейронних мереж є вкрай ефективним. Подібні алгоритми можуть визначати різні види аномалій залежно від вилучених відхилень у даних. У деяких випадках навіть стабільні дані можуть бути ознакою аномалії. На планеті відбувається багато внутрішніх циклів, які мають свій вплив на метеоситуацію, наприклад такі як зміна пір року та цикли кліматичних періодів.

Ефективність розвитку сфери сильно залежить від стандартизації її елементів, наприклад розробки визначених стандартів для метеостанцій, що дозволить максимально спростити процес розробки ПЗ для них, вибрати найоптимальніші алгоритми роботи [7]. Створення специфічних програмних бібліотек і фреймворків дозволить полегшити написання програм, оскільки базові алгоритми будуть вже готові, це зменшить шанс помилки і збільшить швидкість розробки шляхом перевикористання коду.

Щодо циклів кліматичних періодів, рис. 1.8, варто описати докладніше. Так як кругообіг циклів кліматичних періодів, як наприклад льодовиковий період, тривають велику кількість років, то варто також враховувати факти еволюції метеорологічних змінних. Під еволюцією у разі мається на увазі природне зміни очікуваних значень метеорологічних змінних протягом тривалого часу. Враховуючи це, можна зрозуміти, що доведеться регулярно адаптувати алгоритми обробки та аналізу під відповідні норми метеоданих, щоб уникнути випадкового визначення їх як аномальних.

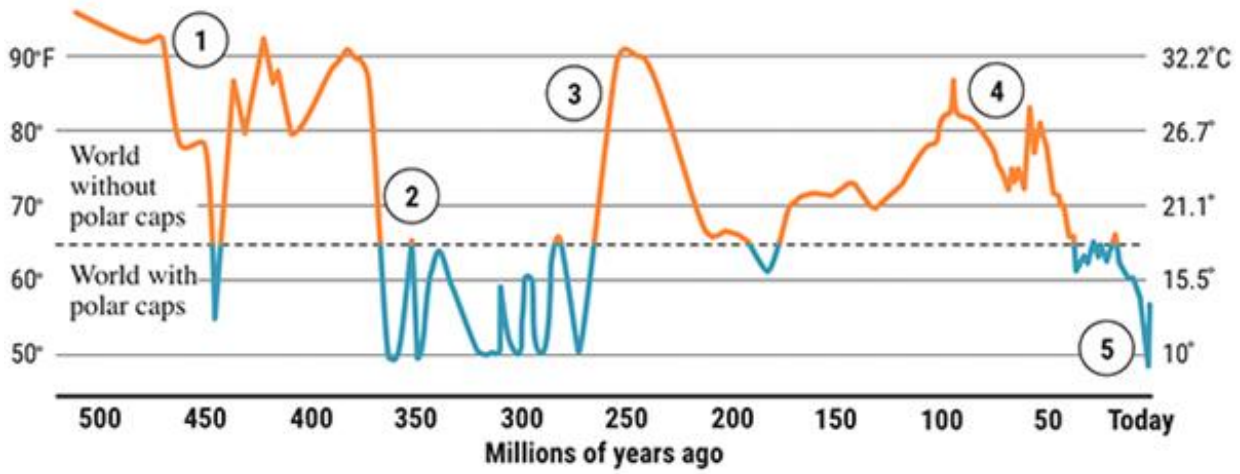

Рисунок 1.8 – Цикли леднікового періоду

Існують методики для визначення значущості вхідних параметрів на результат роботи мережі, для цього можна запитати коефіцієнти, що склалися у нейронів при навчанні і на їх підставі можна зробити висновок. Необхідно це для того, щоб дати оцінку тому, наскільки ефективно складено набір вхідних даних [7].

За допомогою цього можна або виключити параметри, які мінімально впливають на результат, що вкрай важливо, так як це дає можливість прискорити алгоритм і зменшити обсяг даних, що зберігаються. Також завдяки цій можливості можна збільшити точність роботи свого алгоритму, для цього необхідно виділити всі теоретично можливі метеорологічні змінні, які можуть

брати участь у процесі, який ви аналізуєте, після чого пробувати підставляти їх комбінації у вигляді вхідних параметрів.

Дана методика лізу даних за допомогою нейронних мереж, з подальшим аналізом ваг нейронів у них дозволяє виявити навіть найбільш неочевидні залежності.

Наприклад, у 2018 році нейромережа виявила залежність між парою людських генів і схильністю людини до раку. Це відкриває можливість до нових досліджень і може в майбутньому зробити значний внесок у лікування раку.

Подібні взаємозв'язки можуть бути виявлені і у сфері метеорології, наприклад взаємозв'язок виду поверхні на те, як поводиться температура повітря.

Іноді дана методика може застосовуватися для аналізу впливу будь-яких сучасних розробок на загальну метеорологічну ситуацію. У серпні 2022 року був розроблений матеріал, що відображає 97.9% світла, що дозволяє знизити нагрівання покритих нею об'єктів.

У кінцевому підсумку це світло розсіюється, отже впливає навколишні об'єкти, отже нагріває їх, в такий спосіб загальна картина змінюється і може призвести до несподіваних наслідків [7].

1.3 Аналіз можливостей сучасних засобів розробки програмного забезпечення з використанням алгоритмів машинного навчання

Сучасні засоби розробки систем з використанням алгоритмів машинного навчання рясніють різними функціями на вирішення будь-яких задач. До таких систем належать TensofFlow, PyTourch, sklearn.

TensorFlow – це бібліотека для машинного навчання, групи технологій, яка дозволяє навчати штучний інтелект вирішенню різних завдань. Бібліотека спочатку розроблена для Python і найчастіше використовується з ним.

Існують продаж TensorFlow для інших мов: C#, C++, Go, Java, Swift і так далі. Вони використовуються рідше за основну — головним чином для написання коду під специфічні платформи. Сама бібліотека написана мовою Python із використанням швидкого та продуктивного C++ для вирішення математичних завдань. Тому вона ефективно працює зі складними обчисленнями. Бібліотека розроблена Google як продовження внутрішньої бібліотеки компанії. TensorFlow безкоштовна, вона має відкритий вихідний код, який можна переглянути на GitHub, її активно підтримує спільнота ентузіастів. Назва читається як «тензор флоу» та утворена від двох понять: тензор та потік даних [7].

Для чого використовується TensorFlow Сама бібліотека включає безліч інструментів для різних напрямків ML, але найчастіше використовується для роботи з нейронними мережами. Це структури, натхненні пристроєм мереж нейронів у людській нервовій системі. Нейронні мережі складаються із програмних елементів-«нейронів» та зв'язків між ними, і такий пристрій дозволяє їм навчатися. TensorFlow працює зі звичайними та глибокими нейронними мережами різних типів: рекурентними, згортковими тощо. Також вона використовується для машинного та глибокого навчання.

Приклади використання технологій — розпізнавання природної мови, зображень та рукописних текстів, різноманітні завдання класифікації чи кластеризації, обробка великих даних, як-от метеоданих.

Можна виділити такі особливості TensorFlow:

– У TensorFlow моделі представлені за допомогою графів математичних абстракцій, які складаються з вершин і шляхів між ними. Граф можна порівняти із схемою доріг між різними точками. У програмуванні це зазвичай потрібно при вирішенні «маршрутних» завдань та при створенні нейронних мереж.

– TensorFlow працює з тензорами - багатовимірними структурами даних у векторному, тобто спрямованому просторі. Вони використовуються в

лінійній алгебрі та фізиці. Звідси походить назва бібліотеки. За допомогою тензорів описуються шляхи графа, а вершини це математичні операції.

– Обчислення TensorFlow виражаються як потоки даних через граф. Це означає, що інформація «рухається» по графу, передається шляхами від вершини до вершини.

– Бібліотека може працювати на потужностях звичайного центрального процесора (CPU) або використовувати потужності графічного процесора (GPU). Режим перемикається у коді. Існує спеціальний тензорний процесор TPU, створений розробниками бібліотеки, - ним можна скористатися через хмарні сервіси Google.

Перевагами TensorFlow можна виділити високий рівень абстракції. Бібліотека написана так, що не потрібно думати про технічну реалізацію абстрактних понять. Можна зосередитись на описі логіки програми та на математиці, а спосіб реалізації обчислень – завдання TensorFlow, а не програміста. Це полегшує розробку та дозволяє сконцентруватися на важливих завданнях [7].

Інтерактивна розробка TensorFlow дозволяє працювати з компонентами моделі окремо і створювати її на ходу, при цьому окремо перевіряти кожен елемент. Це зручніше, ніж описувати граф як єдину монолітну структуру. Підхід робить розробку більш інтерактивною — структуру можна гнучко налаштовувати та змінювати.

Гнучкість TensorFlow можна використовувати для створення нейронних мереж, для глибокого навчання та інших напрямів ML. Його функції різноманітні на вирішення широкого спектра завдань. Завдяки графам та тензорам у TensorFlow можна легко зобразити складну математичну структуру. Гнучкість стосується як функцій, а й технічної боку питання: бібліотека працює і з центральним, і з графічним процесором, її легко використовувати з іншими інструментами для машинного навчання. Наприклад, TensorFlow застосовують із API Keras.

Кросплатформенність TensorFlow, як і сам Python, працює в популярних операційних системах, локально або в хмарі. Вона має розширення для мобільних пристроїв, IoT та браузерних додатків. Для мобільних пристроїв та інтернету речей можна скористатися середовищем TensorFlow Lite. А якщо модель машинного навчання має працювати у браузері, підійде TensorFlow.js - версія для використання з JavaScript та Node.js.

Велика спільнота TensorFlow — популярна бібліотека, тому на багато питань легко можна знайти відповідь у спільноті. Воно розвиває технологію, створює нові продукти та доповнення, пов'язані з TensorFlow, пише документацію та туторіали. Все це полегшує старт і дозволяє отримати з бібліотеки максимум користі [8].

Власні стандарти TensorFlow – продукт Google. Компанія відома своїми стандартами для технологій. Перша версія бібліотеки була призначена для внутрішнього використання. Досі в TensorFlow зустрічається неочевидна поведінка, через яку код може бути складніше налагоджувати. Складнощі можна компенсувати, якщо уважно вивчати документацію та користуватися додатковими інструментами для налагодження.

Високе споживання пам'яті, якщо TensorFlow використовується з графічним процесором, вона забирає всю його пам'ять. Це знижує продуктивність. Наприклад, якщо моделей кілька і вони написані на різних фреймворках, TensorFlow забере відеопам'ять у інших – виникне помилка. Щоб цього не було, споживання пам'яті бібліотекою треба обмежувати вручну. Складність у вивченні машинного навчання загалом. З TensorFlow складності можуть виникнути через її специфічні стандарти — це не найприязніша до новачків бібліотека.

TensorFlow встановлюється за допомогою pip, пакетного менеджера Python. Версії для роботи з CPU та GPU завантажуються окремо. Також для коректної роботи потрібна Anaconda – спеціальний дистрибутив Python для машинного навчання [8].

Більш детально про внутрішній пристрій TensorFlow можна сказати що TensorFlow — це наскрізна платформа для машинного навчання. Він підтримує наступне:

– Числове обчислення на основі багатовимірного масиву (подібно до NumPy.)

– GPU і розподілена обробка

– Автоматична диференціація

– Побудова моделі, навчання та експорт

Тензори, TensorFlow працює з багатовимірними масивами або тензорами, представленими як об’єкти tf.Tensor. Найважливішими атрибутами tf.Tensor є його форма і dtype: Tensor.shape: повідомляє розмір тензора вздовж кожної з його осей. Tensor.dtype: повідомляє вам тип усіх елементів у тензорі.

TensorFlow реалізує стандартні математичні операції над тензорами, а також багато спеціалізованих операцій для машинного навчання.

Змінні, звичайні об’єкти tf.Tensor є незмінними. Щоб зберегти вагові коефіцієнти моделі (або інший змінний стан) у TensorFlow, використовуйте tf.Variable [8].

Градієнтний спуск і пов’язані з ним алгоритми є наріжним каменем сучасного машинного навчання. Щоб увімкнути це, TensorFlow реалізує автоматичне диференціювання (autodiff), яке використовує обчислення для обчислення градієнтів. Зазвичай ви використовуєте це для обчислення градієнта помилки або втрати моделі відносно її ваг.

Хоча можно використовувати TensorFlow в інтерактивному режимі, як і будь-яку іншу бібліотеку Python, TensorFlow також надає інструменти для:

– Оптимізація продуктивності: для прискорення навчання та висновків.

– Експорт: щоб ви могли зберегти свою модель після завершення навчання.

Для цього потрібно використовувати tf.function, щоб відокремити чистий код TensorFlow від Python [9]. Під час першого запуску функці tf.function, хоча вона виконується на Python, вона фіксує повний оптимізований графік, що представляє обчислення TensorFlow, виконані в межах функції. Під час наступних викликів TensorFlow виконує лише оптимізований графік, пропускаючи будь-які кроки, не пов’язані з TensorFlow. Графік може бути непридатним для повторного використання для вхідних даних із іншою підписом (форма і dtype), тому натомість створюється новий графік. Ці отримані графіки дають дві переваги:

– У багатьох випадках вони забезпечують значне прискорення виконання (хоча це не тривіальний приклад).

– Можна експортувати ці графіки за допомогою tf.saved_model для запуску в інших системах, як-от сервер або мобільний пристрій, встановлення Python не потрібне.

Модулі, шари та моделі. tf.Module – це клас для керування вашими об’єктами tf.Variable та об’єктами tf.function, які з ними працюють. Клас tf.Module необхідний для підтримки двох важливих функцій:

Можна зберігати та відновлювати значення своїх змінних за допомогою tf.train.Checkpoint. Це корисно під час навчання, оскільки можна швидко зберегти та відновити стан моделі.

Можна імпортувати та експортувати значення tf.Variable і графіки tf.function за допомогою tf.saved_model. Це дозволяє запускати вашу модель незалежно від програми Python, яка її створила.

Отримана SavedModel не залежить від коду, який її створив. Ви можете завантажити SavedModel з Python, інших мовних прив’язок або TensorFlow Serving. Ви також можете конвертувати його для запуску з TensorFlow Lit або TensorFlow JS. Класи tf.keras.layers.Layer та tf.keras.Model побудовані на tf.Module забезпечуючи додаткову функціональність і зручні методи для

створення, навчання та збереження моделей. Деякі з них демонструються в наступному розділі.

Однак альтернативою TensorFlow можна представити PyTourch. PyTorch визначається як бібліотека машинного навчання з відкритим кодом для Python. Він використовується для програм, таких як обробка природної мови. Спочатку він був розроблений дослідницькою групою зі штучного інтелекту Facebook та програмним забезпеченням Uber Pyro для ймовірнісного програмування, яке ґрунтується на ньому [9].

Спочатку PyTorch був розроблений Х'ю Перкінсом як оболонка Python для LusJIT, заснована на платформі Torch. Є два варіанти PyTorch.
PyTorch перепроектує та впроваджує Torch у Python, спільно використовуючи ті ж основні бібліотеки C для внутрішнього коду. Розробники PyTorch налаштували цей внутрішній код для ефективної роботи з Python. Вони також зберегли апаратне прискорення на основі графічного процесора, а також функції розширення, які зробили Torch на базі Lua.

Простий інтерфейс – PyTorch пропонує простий використання API; отже, він вважається дуже простим у роботі та працює на Python. Виконання коду у цьому середовищі досить просто. Використання Python – це бібліотека вважається Pythonic, яка плавно інтегрується зі стеком даних Python. Таким чином, він може використовувати всі сервіси та функціональні можливості, які пропонують середовищем Python. Обчислювальні графи PyTorch надає відмінну платформу, яка пропонує динамічні обчислювальні графи. Таким чином користувач може змінити їх під час виконання. Це дуже корисно, коли розробник не знає скільки пам'яті потрібно для створення моделі нейронної мережі [9].

У таблиці 1.1 наведено порівняння між TensorFlow та PyTorch

Таблиця 1.1 – Порівняння бібліотек PyTorch та TensorFlow

PyTorch	TensorFlow

PyTorch тісно пов'язаний із заснованим на lua фреймворком Torch, який активно використовується у Facebook.	TensorFlow розроблено Google Brain і активно використовується в Google.
PyTorch є відносно новим у порівнянні з іншими конкурентними технологіями.	TensorFlow не є новим і розглядається багатьма дослідниками та професіоналами як інструмент, необхідний для роботи.
PyTorch включає все в імперативній і динамічній манері.	TensorFlow включає статичні та динамічні графіки у вигляді комбінації.
Граф обчислень у PyTorch визначається під час виконання.	TensorFlow не включає опцію часу виконання.
PyTorch включає можливість розгортання для мобільних і вбудованих платформ.	TensorFlow краще працює для вбудованих фреймворків.

Переваги PyTorch:

– Код легко налагоджувати та розуміти.

– Він включає багато шарів, як Torch.

– Включає безліч функцій втрати.

– Це можна як розширення NumPy для графічних процесорів.

– Це дозволяє будувати мережі, структура яких залежить самих обчислень.

Чим швидше виходить обчислювати свою функцію і чим гнучкіші можливості для її визначення, тим краще. Тепер, коли кожен фреймворк вміє

використовувати всю потужність відеокарт, перший критерій перестав відігравати значну роль [9].

Тензорні обчислення - основа PyTorch, каркас, навколо якого збільшується вся інша функціональність. На жаль, не можна сказати, що міць і виразність бібліотеки в даному аспекті збігаються з такою у NumPy. Щодо роботи з тензорами, PyTorch керується принципом максимальної простоти та прозорості, надаючи тонку обгортку над викликами BLAS. Цей фреймворк є низькорівневим у сенсі реалізації інтерфейсу його використання. Однак його більш ніж достатньо для вирішення майже будь-яких задач машинного навчання.

Ще однією цікавою бібліотекою для розв'язання задач машинного навчання є sklearn. Це бібліотека яка допомагає створювати прості моделі для машинного навчання, вони застосовуються для обчислення моделей, що легко збираються, для алгоритмів де потрібна швидкості і мала витрата ресурсів пам'яті.

Які завдання вирішує бібліотека, наприклад, препроцессинг. термін "препроцессинг" (preprocessing) означає попередню обробку даних. Вони приводяться до виду, необхідного для подачі на вхід будь-якого алгоритму. Наприклад, зображення підганяються під один розмір та колірну схему. З даних вилучаються ключові ознаки, якими буде вчитися модель, і також призводять до потрібного формату.

Зменшення розмірності. Часто у даних міститься надмірна інформація. Наприклад, деякі ознаки можна отримати з інших. Щоб зробити подальший аналіз ефективнішим, розмірність вибірки зменшується — так, щоб зберегти максимум корисних даних. І тому використовуються спеціальні методи, наприклад метод головних компонент.

Виявлення аномалій. Алгоритми «відсікають» з набору даних помилкові чи дивні записи, які лише додають зайві похибки. Це потрібно, щоб аналіз та навчання працювали точніше.

Вибір датасету. Якщо потрібно познайомитися з бібліотекою, можна скористатися одним із готових навчальних наборів. Вони теж є у бібліотеці.

Вибір моделі. Функції та алгоритми допомагають оцінити ефективність різних моделей для вирішення задачі. Їх можна порівнювати один з одним, проводити валідацію результатів, вибирати точніші. Це потрібно для покращення якості навчання [9]. Приклад регресії наведено на рис.1.9.

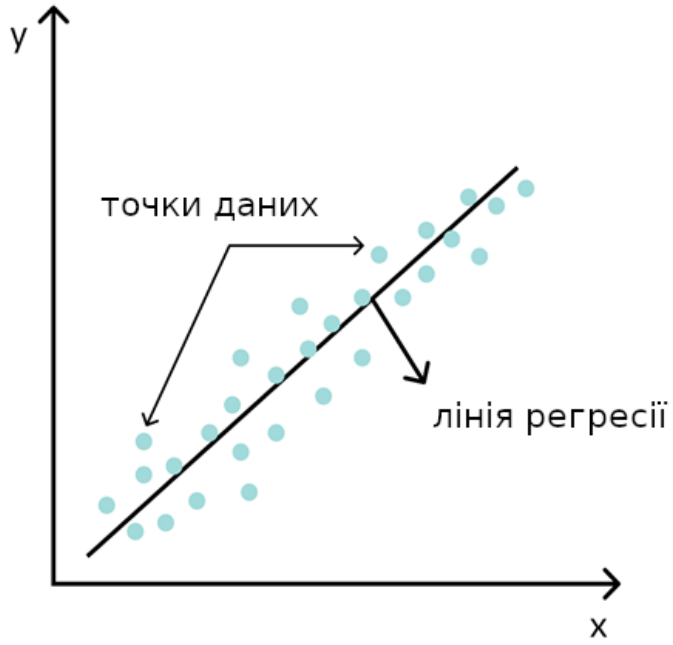

Рисунок 1.9 — Приклад регресії

Регресія. Це прогнозування показників за наявними даними, які можуть приймати нескінченну кількість різних значень. Ці показники би мало бути пов'язані з будь-яким об'єктом, тобто. бути його атрибутами. Наприклад, прогноз кількості користувачів на сайті у різні дні – це завдання регресії.

Класифікація - прогнозування показника з кінцевою кількістю значень. Простіше визначення — прогнозування категорії, до якої належить об'єкт. Розпізнавання жанру тексту чи об'єктів на зображенні – це завдання класифікації [10].

Кластеризація. Це розподіл даних з датасету за великими групами — кластерами, тобто їх угруповання. Схожі об'єкти об'єднуються у класи. Критерії, якими визначається схожість, залежить від моделі та умов завдання.

1.4 Висновки до першого розділу

Результатом проведеного аналізу предметної області, було визначено що сучасна метеорологія є добрим напрямком для впровадження сучасних інформаційних технологій. Це пов’язано з актуальним використанням засобів комунікації та інформаційного обміну між метеостанціями та метеоцентрами [10]. Проведений аналіз інформаційних матеріалів показав, що сфера потребує нові алгоритми аналізу метеоданих, які будут більш гнучкими та будуть аналізувати більше різноманітниї метеорологічниї змінних. Результати аналізу існуючих алгоритмів прогнозу метеоданих з ціллю найти серед них більшт оптимальний та точний. Також потрібно провести достідження з ціллю знайти найкращу комбінацію метеорологічних смінних для розрахунку прогнозу погоди. Розглянувши найпопулярніші засоби розробки, що використовуються в галузі машинного навчання, можна дійти висновку, що найбільш зручною для вирішення задач роботи є мова Python та фреймворк для створення нейромереж TensorFlow 2, адже саме в ній є всі необхідні функції. Отримані в рамках розділу результати будуть використані під час обґрунтування вибірку напрямку дослідження та формалізації завдання з розробки програмного забезпечення у подальших розділах роботи.

2. ОБГРУНТУВАННЯ ВИБОРУ НАПРЯМКУ ДОСЛІДЖЕННЯ

2.1 Постановка завдання регресії

З метою визначення усіх функціональних можливостей інформаційних систем для метеорологів необхідно проаналізувати існуючі рішення та сформувати головні вимоги до таких систем. Для об’єктивного порівняння, необхідно використовувати тільки web-рішення, які автоматизують робочі задачі метеорологів, або спрощують роботу метеорологічного центру.

Відсутність системи збору метеоданих це критичний недолік для майбутнього метеорології, який не дасть їй можливості розвиватися.

Основним завданням системи є аналіз метеорологічних даних з різних джерел і побудова короткострокового прогнозу погоди з прогнозуванням можливих надзвичайних погодних явищ. Гроза і сильна злива, град і шквальний вітер, сніговий буран або екстремальна температура часто формуються протягом короткого часу і часто мають відносно невеликі розміри, що не дозволяє створення прогнозу погоди звичайними інструментами.

Система короткострокового прогнозу погоди і попередження про надзвичайні погодні умови аналізує метеорологічні дані в режимі реального часу в рамках заданої місцевості обмеженою відносно невеликою територією і будує прогноз виникнення надзвичайних погодних умов в рамках декількох годин, що дозволяє надати більш точний прогноз [11].

Починається весь процес роботи системи зі збору даних з якими далі доведеться працювати, після чого вони обробляються та зберігаються у системі. Після цього в дію вступає модуль аналізу цих даних, він складається з алгоритму розрахунку регресії та алгоритму класифікації. На даний момент розглянемо алгоритм розрахунку регресії, адже саме це завдання вирішує система під час побудови короткострокового прогнозу погоди.

Розрахунок алгоритму регресії починається з побудови математичного рівняння, яке оцінює лінію простої (парної) лінійної регресії:

$$Y = a + bx, \tag{2.1}$$

де x називається незалежною змінною чи предиктором. Y – залежна змінна або змінна відгук. Це значення, що ми очікуємо для y (у середньому), якщо знаємо величину x, тобто. це «передбачене значення y» a – вільний член (перетин) лінії оцінки; це значення Y коли x=0, рис 2.1. b – кутовий коефіцієнт чи градієнт оціненої лінії; вона є величину, яку Y збільшується загалом, якщо ми збільшуємо x однією одиницю. a та b називають коефіцієнтами регресії оціненої лінії, хоча цей термін часто використовують тільки для b [11].

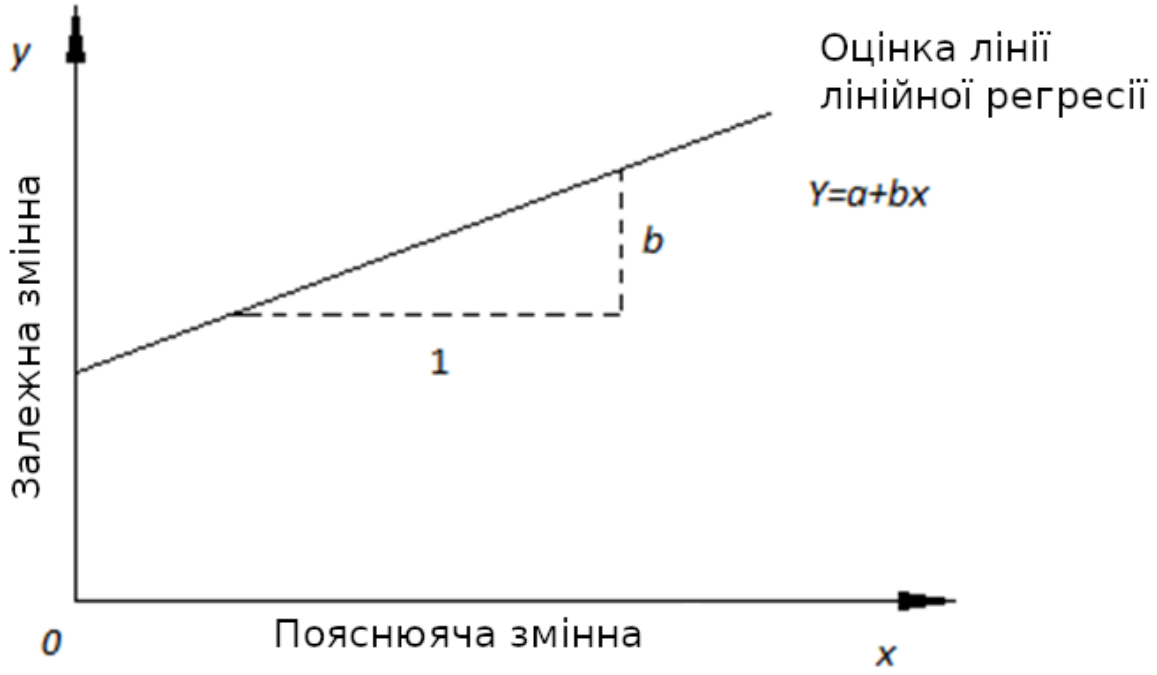

Рисунок 2.1 - Лінія лінійної регресії, що показує перетин a та кутовий коефіцієнт b (величину зростання Y при збільшенні x на одну одиницю)

Парну лінійну регресію можна розширити, включивши до неї більше однієї незалежної змінної; у цьому випадку вона відома як множинна регресія. Аномальні значення (викиди) та точки впливу. "Впливове" спостереження, якщо воно опущене, змінює одну або більше оцінок параметрів моделі (тобто кутовий коефіцієнт або вільний член). Викид (спостереження, що суперечить більшості значень у наборі даних) може бути "впливовим" спостереженням і може добре виявлятися візуально, під час огляду двовимірної діаграми розсіювання або графіка залишків. І для викидів, і для "впливових" спостережень (крапок) використовують моделі як з їх включенням, так і без них звертають увагу на зміну оцінки (коефіцієнтів регресії). При проведенні аналізу не варто відкидати викиди або точки впливу автоматично, оскільки звичайне ігнорування може вплинути на отримані результати [11].

Гіпотеза лінійної регресії при побудові лінійної регресії перевіряється нульова гіпотеза про те, що генеральний кутовий коефіцієнт лінії регресії β дорівнює нулю. Якщо кутовий коефіцієнт лінії дорівнює нулю, між x ***і*** y немає лінійного співвідношення: зміна x не впливає на y. Для тестування нульової гіпотези про те, що дійсний кутовий коефіцієнт β дорівнює нулю використовується наступний алгоритм:

Обчислити статистику критерію, що дорівнює відношенню $blSE(b)$, яка підпорядковується $t-$ розподілу з $(n-2)$ ступенями свободи, де SE(b) - стандартна помилка коефіцієнта b.

$$b = \frac{\sum(x-\bar{x})(y-\bar{y})}{\sum(x-\bar{x})^2} \tag{2.2}$$

$$SE(b) = \frac{s_{rez}}{\sum(x-\bar{x})^2} \tag{2.3}$$

Зазвичай, якщо досягнутий рівень значущості P < 0.05 нульова гіпотеза відхиляється. Можна розрахувати 95% довірчий інтервал для генерального кутового коефіцієнта β:

$$b \pm t_{0.05} SE(b) \tag{2.4}$$

де $t_{0.05}$ - відсоткова точка t - розподілу зі ступенями свободи (n - 2) що дає можливість двостороннього критерію 0.05. Це той інтервал, який містить генеральний кутовий коефіцієнт з ймовірністю 95%.

Для великих вибірок, де, n ≥ 100 можно апроксимувати $t_{0.05}$ значенням 1,96 (тобто статистика критерію прагнутиме до нормального розподілу).

Оцінка якості лінійної регресії: коефіцієнт детермінації R^2. Через лінійне співвідношення y і x очікується, що y змінюється, у міру того як змінюється x, і називаємо це варіацією, яка обумовлена чи пояснюється регресією. Залишкова варіація має бути якнайменше.

Якщо це так, то більшість варіації y пояснюватиметься регресією, а точки лежатимуть близько до лінії регресії, тобто. лінія добре відповідає даним [11].

Частка загальної дисперсії y, яка пояснюється регресією називають коефіцієнтом детермінації, зазвичай виражають через відсоткове співвідношення та позначають R^2(у парній лінійній регресії це величина r^2, квадрат коефіцієнта кореляції), що дозволяє суб'єктивно оцінити якість рівняння регресії.

Різниця (100 - R^2) є відсотком дисперсії який не можна пояснити регресією.

Немає формального тесту для оцінки R^2 потрібно покластися на суб'єктивне судження, щоб визначити якість припасування лінії регресії.

2.2 Порівняльний аналіз та вибір алгоритму регресії

У разі під алгоритмом регресії мається на увазі збірний образ всіх алгоритмів прогнозування будь-яких даних [11]. Далі розглядаються різні варіанти архітектур нейронних мереж, які застосовуються в алгоритмах прогнозування.

Першою розглянутою архітектурою буде багатошаровий перцептрон, рис 2.2.

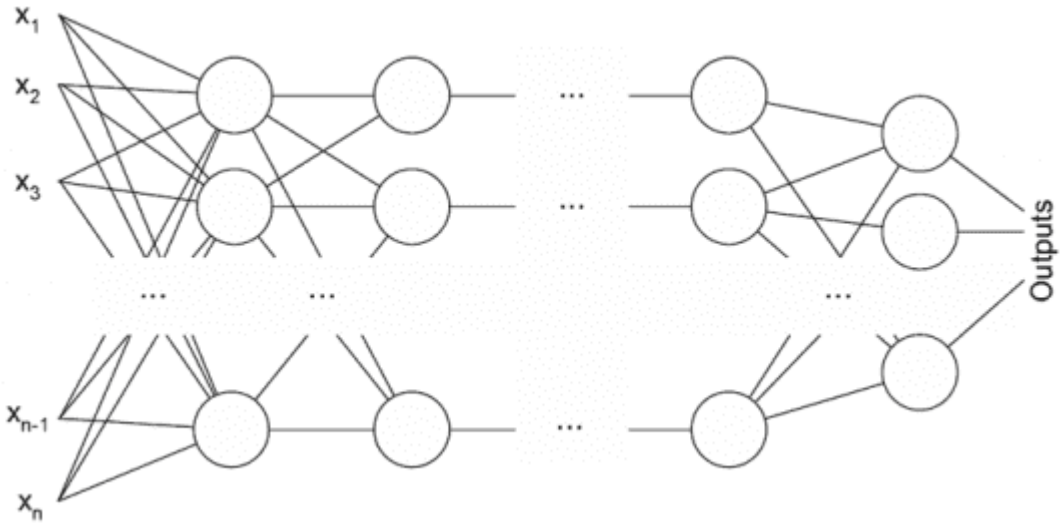

Рисунок 2.2 - Багатошаровий перцептрон

Багатошаровий перцептрон складається з трьох або більше шарів. Кожен вузол у шарі з'єднаний з кожним вузлом у наступному шарі, що робить мережу повністю пов'язаною. Приховані шари містять вузли мережі (одиниці), що не піддаються спостереженню. Кожна прихована одиниця є функцією виваженої суми вхідних даних. Ця функція є функцією активації, при цьому значення ваги визначаються алгоритмом оцінки. Мережа містить другий прихований рівень, кожна прихована одиниця у якому є функцією від виваженої суми одиниць у першому прихованому рівні. В обох рівнях використовують

однакову функцію активації. Кількість прихованих шарів, багатошаровий перцептрон може мати один або два приховані рівні [11].

Функція активації "зв'язує" шар зважених сум об'єктів із наступним шаром значень даних об'єктів.

Гіперболічний тангенс. Це така функція: $y(c) = \tanh(c) = \frac{(ec-e-c)}{(ec+e-c)}$. Вона визначена для дійсних змінних та переводить їх у діапазон (–1, 1). При використанні автоматичного вибору архітектури це функція для всіх нейронів у прихованих шарах.

Сігмоїд. Це така функція: $y(c) = \frac{1}{(1+e+c)}$. Вона визначена для дійсних змінних та переводить їх у діапазон (0, 1).

Кількість нейронів. Кількість одиниць у кожному прихованому шарі може бути задано явно або автоматично визначено алгоритмом оцінки. Вихідний шар містить цільові (залежні) змінні. функція активації. Функція активації "зв'язує" шар зважених сум об'єктів із наступним шаром значень даних об'єктів.

Тотожність. Це функція γ(c) = c. Вона визначена всім аргументів і повертає їх незміненими. При використанні автоматичного вибору архітектури це функція активації для нейронів у прихованих шарах, якщо не існує кількісних залежних змінних.

Софтмакс. Це така функція: $y(ck) = \frac{\exp(ck)}{\sum jexp(cj)}$. Вона приймає вектор дійсних аргументів і перетворює їх на вектор, компоненти якого укладені в діапазоні (0, 1), а їх сума дорівнює 1. Функція софтмакс доступна тільки в тому випадку, якщо всі залежні категоріальні змінні. При використанні автоматичного вибору архітектури це функція активації для нейронів у вихідному шарі, якщо всі залежні категоріальні змінні.

Така архітектура частіше знаходить застосування у завданнях розпізнавання мови та машинному перекладі [12].

Рекурсивні нейронні мережі - вид нейронних мереж, що працюють з даними змінної довжини. Моделі рекурсивних мереж використовують ієрархічні структури зразків під час навчання. Наприклад, зображення, що складаються зі сцен, що поєднують підсцени, що включають багато об'єктів. Виявлення структури сцени та її деконструкція-нетривіальне завдання. При цьому необхідно ідентифікувати окремі об'єкти, так і всю структуру сцени.

У рекурсивних мережах нейрони з однаковими вагами активуються рекурсивно відповідно до структури мережі. У процесі роботи рекурсивної мережі виробляється модель для передбачення як структур змінної розмірності, так і скалярних структур через активацію структури відповідно до топології. Мережі RvNNs успішно застосовуються при навчанні послідовних структур та дерев у завданнях обробки природної мови, при цьому фрази та речення моделюються через векторне уявлення слів. RvNNs спочатку з'явилися для розподіленого уявлення структур, використовуючи предикати математичної логіки. Розробки рекурсивних мереж та перші моделі почалися в середині 1990-х. У найпростішій архітектурі вузли мережі сходяться до батьків через матрицю ваги прихованого шару, що використовується багаторазово через всю мережу, і нелінійну функцію активації типу гіперболічного тангенсу. Архітектура простої рекурсивної мережі (W - навчена матриця ваг) наведена на рис.2.3 [15].

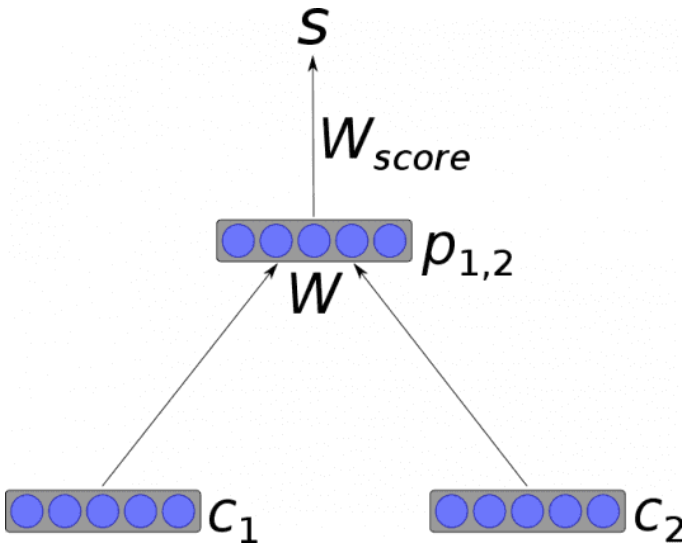

Рисунок 2.3 - Архітектура простої рекурсивної мережі

Ця архітектура з деяким удосконаленням використовується для послідовного дешифрування натуральних сцен зображення або для структурування речень природної мови.

Рекурсивна каскадна кореляція RecCC - це підхід до конструювання рекурсивних мереж, що оперують із трьома доменами, перші додатки такого роду з'явилися в хімії, а розширення утворює спрямований ациклічний граф. У 2004 році було запропоновано систему навчання рекурсивної мережі без вчителя. Тензорні рекурсивні мережі використовують одну функцію тензора для всіх вузлів дерева.

Рекурентна нейронна мережа, на відміну прямої нейронної мережі, є варіантом рекурсивної IHC, у якій зв'язки між нейронами — спрямовані цикли. Останнє означає, що вихідна інформація залежить не тільки від поточного входу, але також станів нейрона на попередньому кроці. Така пам'ять дозволяє користувачам вирішувати завдання NLP: розпізнавання рукописного тексту чи мови. У статті Natural Language Generation, Paraphrasing and Summarization of User Reviews with Recurrent Neural Networks автори показують модель рекурентної мережі, яка генерує нові пропозиції та короткий зміст текстового документа. Siwei Lai, Liheng Xu, Kang Liu, та Jun Zhao у своїй роботі Recurrent Convolutional Neural Networks for Text Classification створили рекурентну згорткову нейромережу для класифікації тексту без рукотворних ознак. Модель порівнюється з існуючими методами класифікації тексту - Bag of Words, Bigrams + LR, SVM, LDA, Tree Kernels, рекурсивними та згортковими мережами. Описана модель перевершує за якістю традиційні методи для всіх використовуваних датасетів [15].

Існує багато різновидів, рішень та конструктивних елементів рекурентних нейронних мереж. Проблема рекурентної мережі полягає в тому, що якщо враховувати кожен крок часу, то стає необхідним для кожного кроку

часу створювати свій шар нейронів, що викликає серйозні обчислювальні складності. Крім того, багатошарові реалізації виявляються обчислювально нестійкими, тому що в них зазвичай зникають або зашкалюють ваги. Якщо обмежити розрахунок фіксованим тимчасовим вікном, то отримані моделі не відображатимуть довгострокових трендів. Різні підходи намагаються вдосконалити модель історичної пам'яті та механізм запам'ятовування та забування.

Цілком рекурентна мережа, ця базова архітектура розроблена у 1980-х. Мережа будується з вузлів, кожен із яких з'єднаний з усіма іншими вузлами. У кожного нейрона поріг активації змінюється з часом і є речовим числом. Кожна сполука має змінну речовинну вагу.

Вузли поділяються на вхідні, вихідні та приховані для навчання з учителем з дискретним часом, кожен (дискретний) крок часу на вхідні вузли подаються дані, а інші вузли завершують свою активацію, і вихідні сигнали готуються для передачі нейроном наступного рівня. Якщо мережа відповідає за розпізнавання мови, в результаті на вихідні вузли надходять вже мітки (розпізнані слова) [15].

У навчанні з підкріпленням (reinforcement learning) немає вчителя, що забезпечує цільові сигнали для мережі, натомість іноді використовується функція пристосованості (придатності) або функція оцінки (reward function), за якою проводиться оцінка якості роботи мережі, при цьому значення на виході впливає на поведінка мережі на вході, зокрема, якщо мережа реалізує гру, на виході вимірюється кількість пунктів виграшу чи оцінки позиції. Кожен ланцюжок обчислює помилку як сумарну девіацію за вихідними сигналами мережі. Якщо є набір зразків навчання, помилка обчислюється з урахуванням помилок кожного зразка.

Мережа довгої короткострокової пам'яті (Long Short-Term Memory, LSTM) - різновид архітектури рекурентної нейромережі, створена для більш точного моделювання часових послідовностей та їх довгострокових залежностей, ніж традиційна рекурентна мережа.

Структура LSTM мережі наведена на рис.2.4.

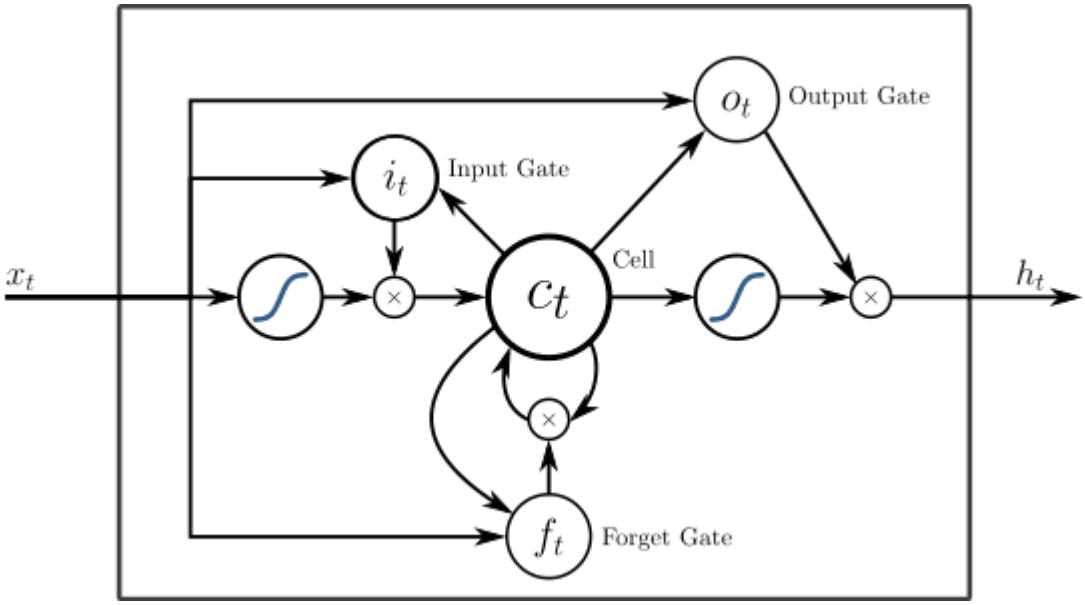

Рисунок 2.4 – Структура LSTM мережі

LSTM-мережа не використовує функцію активації в рекурентних компонентах, збережені значення не модифікуються, а градієнт не прагне зникнути під час тренування. Часто LSTM застосовується у блоках по кілька елементів. Ці блоки складаються з 3 або 4 затворів (наприклад, вхідного, вихідного та гейту забування), які контролюють побудову інформаційного потоку по логістичній функції.

У Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling автори показують архітектуру глибокої LSTM рекурентної мережі, яка досягає хороших результатів для великомасштабного акустичного моделювання [15].

У роботі Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Recurrent Neural Network представлена модель автоматичної морфологічної розмітки. Модель показує точність 97.4 % у розмітці. Apple, Amazon, Google, Microsoft та інші компанії впровадили у продукти LSTM-мережі як фундаментальний елемент.

Модель послідовності, якщо розглядати на високому рівні, модель seq2seq має кодер, декодер і проміжний крок як основні компоненти (рис.2.5).

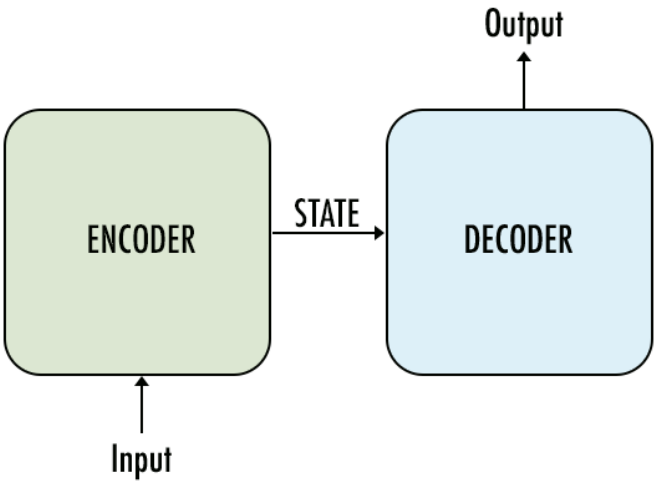

Рисунок 2.5 – Схема архітектури автоенкодера

Використовуючи вбудовування, спочатку треба скласти «словниковий» список, який містить усі слова, які потрібні, щоб модель могла використовувати або читати [16]. Вхідні дані моделі повинні бути тензорами, що містять ідентифікатори слів у послідовності. Однак існує чотири символи, які повинні бути в словнику. Логіка роботи автоенкодера наведена на рис.2.6.

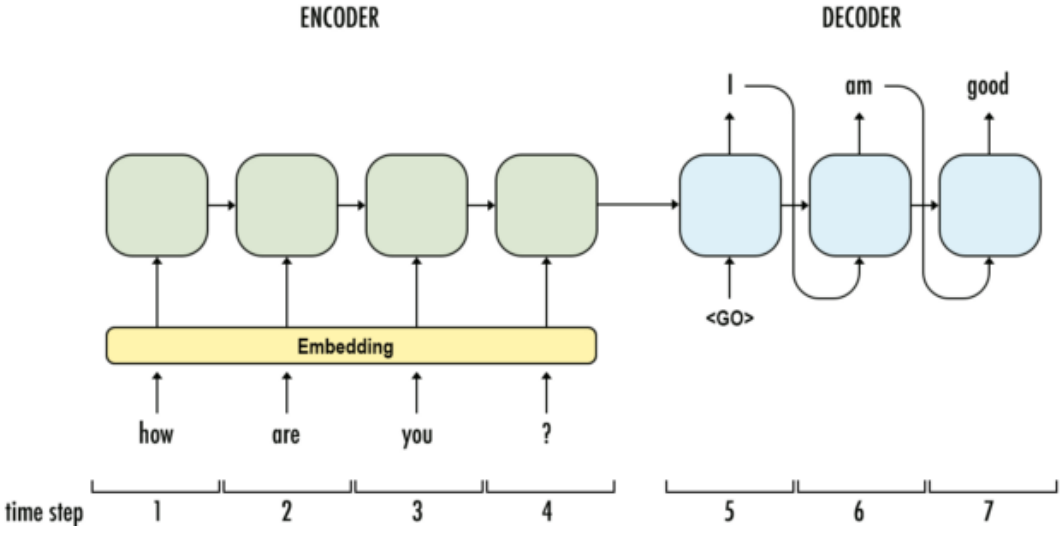

Рисунок 2.6 – Логіка роботи автоенкодера

Словники Seq2seq зазвичай резервують перші чотири місця для цих елементів:

– <PAD>: під час навчання потрібно буде передавати приклади в мережу пакетами. Усі вхідні дані в цих пакетах мають бути однакової ширини, щоб мережа могла виконати обчислення. Однак приклади не мають однакової довжини. Ось чому потрібно буде доповнити коротші вхідні дані, щоб привести їх до однакової ширини пакета

– <EOS>: це ще одна необхідність пакетування, але більше з боку декодера. Це дозволяє повідомляти декодеру, де закінчується речення, а також дозволяє декодеру вказувати те саме у своїх виводах.

– <Nk>: Якщо тренувати модель на реальних даних, виявиться, що можна значно підвищити ефективність ресурсів моделі, ігноруючи слова, які не часто виявляються у словникові запаси, щоб гарантувати розгляд.

– <GO>: Це вхідні дані для першого тимчасового кроку декодера, щоб повідомити декодеру, коли почати генерувати вихідні дані.

Примітка, для представлення цих функцій можна використовувати інші теги. Наприклад <s> і </s> замість <GO> і <EOS>. Тому потрібно переконатися, що все, що використовується, узгоджено через попередню обробку та навчання моделі/висновок.

Підготовка вхідних даних для навчального графіка трохи складніша з двох причин:

1. Ці моделі працюють набагато краще, якщо передати декодеру цільову послідовність, незалежно від того, які її часові кроки фактично виводяться під час тренування. Отже, на відміну від графіка, не потрібно передавати вихідні дані декодера самому собі на наступному часовому етапі.

2. Дозування, одна з оригінальних статей про послідовність дій, Sutskever et al. 2020, повідомляє про кращу продуктивність моделі, якщо вхідні

дані змінені. Таким чином, також можна змінити порядок слів у послідовності введення.

Під час попередньої обробки створюється свій словниковий запас з унікальних слів, створити копію розмов із заміною слів на їхні ідентифікатори, можна додати ідентифікатори слів <GO> і <EOS> до цільового набору даних зараз або зробити це під час навчання [16].

2.3 Концепція попередньої обробки та аналізу даних

У практиці використання нейронних мереж використовують різні підходи до нормалізації даних. Але всі вони спрямовані на утримання даних навчальної вибірки та вихідних даних прихованих шарів нейронної мережі в заданому діапазоні та з певними статистичними характеристиками вибірки, такими як дисперсія та медіана. У нейронах мережі застосовуються лінійні перетворення, які у процесі навчання зміщують вибірку у бік антиградієнта.

Розглянемо повнозв'язковий перцептрон з двома прихованими шарами. При прямому проході кожен шар генерує деяку сукупність даних, які є навчальною вибіркою для наступного шару. Результат роботи вихідного шару порівнюється з еталонними даними і зворотному проході поширюється градієнт помилки від вихідного шару через приховані шари до вихідних даних. Отримавши кожному нейроні свій градієнт помилки і оновлюючи вагові коефіцієнти, підлаштовуючи нейронну мережу під навчальні вибірки останнього прямого проходу. Виникає конфлікт: підлаштовуючи другий прихований шар (H2) під вибірку даних на виході першого прихованого шару (H1), в той час як змінивши параметри першого прихованого шару змінили масив даних [16]. Підлаштовуючи другий прихований шар під вже неіснуючу вибірку даних. Аналогічна ситуація з вихідним шаром, який підлаштовується під вже змінений вихід другого прихованого шару. А якщо ще врахувати

спотворення між першим і другим прихованими шарами, масштаби помилки збільшуються. І чим глибша нейронна мережа, тим сильніший прояв цього ефекту. Це було названо внутрішнім підступним зрушенням.

У класичних нейронних мережах зазначена проблема частково вирішувалася зменшенням коефіцієнта навчання. Невеликі зміни вагових коефіцієнтів не дуже змінюють розподіл вибірки на виході нейронного шару. Але такий підхід не вирішує масштабування проблеми зі зростанням кількості шарів нейронної мережі та знижує швидкість навчання. Ще одна проблема невеликого коефіцієнта навчання - застрявання в локальних мінімумах.

У лютому 2020 року Sergey Ioffe та Christian Szegedy запропонували метод пакетної нормалізації даних (Batch Normalization) для вирішення проблеми внутрішнього зсуву коваріації. Суть методу полягала у нормалізації кожного окремого нейрона на певному часовому інтервалі зі зміщенням медіани вибірки до нуля та приведенням дисперсії вибірки к1 [16].

Алгоритм проведення нормалізації наступний. Спочатку за вибіркою даних вважається середнє значення.

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \tag{2.5}$$

де m – розмір вибірки (batch).

Потім вважаємо дисперсію вихідної вибірки.

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2 \tag{2.6}$$

І нормалізуємо дані вибірки, привівши вибірку до нульового середнього та одиничної дисперсії.

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \quad (2.7)$$

В знаменнику до дисперсії вибірки додається константа є, невелике позитивне число з метою виключити поділ на нуль.

Така нормалізація може спотворити вплив вихідних даних. Тому автори методу додали ще один крок — масштабування та усунення. Було введено 2 змінні γ і β, які навчаються разом із нейронною мережею методом зворотного градієнтного спуску.

$$y_i = y\hat{x}_i + \beta \equiv BN_{y,\beta}(x_i) \quad (2.8)$$

Застосування цього методу дозволяє кожному етапі навчання отримувати вибірку даних з однаковим розподілом, що практично робить навчання нейронної мережі більш стабільним і дозволяє збільшити коефіцієнт навчання. Загалом це дозволить підвищити якість навчання за менших витрат часу на навчання нейронної мережі [16].

Але в той же час зростають витрати на зберігання додаткових коефіцієнтів. А також для розрахунку ковзної середньої та дисперсії потрібно зберігання в пам'яті історичних даних кожного нейрона на весь розмір пакету. Нагадаємо формулу експоненційної ковзної середньої.

$$\mu_i = \frac{(m-1)}{m}\mu_{i-1} + \frac{1}{m}x_i \quad (2.9)$$

де m - розмір вибірки (batch), i – ітерація.

Для зближення графіків ковзної дисперсії та експоненційної ковзної дисперсії потрібно трохи більше ітерацій (310-320), але в цілому картина схожа. Що стосується дисперсією застосування експоненціальної дає як

економію пам'яті, а й значно знижує кількість обчислень, т.к. для ковзної дисперсії перераховується відхилення від середньої для всього batch-а.

Експерименти, проведені авторами методу, показують, що застосування методу Batch Normalization виступає у ролі регуляризатора [19]. Це дозволяє відмовитися від інших методів регуляризації, зокрема від розглянутого раніше Dropout. Більше того, є пізніші роботи, в яких показано, що спільне використання Dropout та Batch Normalization негативно позначається на результатах навчання нейронної мережі. У сучасних архітектурах нейронних мереж запропонований алгоритм нормалізації можна зустріти у різних варіаціях. Автори пропонують використовувати Batch Normalization безпосередньо перед нелінійністю (формулою активації). Як одну із варіацій даного алгоритму можна розглядати метод Layer Normalization, представлений у липні 2021 року. Головна думка цього розділу в тому, що процес нормалізації вкрай важливий, щоб навчання нейронної мережі проходило ефективно і різні похибки даних не впливали на її роботу.

2.4 Висновок до другого розділу

У другому розділі проведено постановку завдання прогнозування. Зроблено порівняльний аналіз та вибір алгоритму регресії. Розглянуто процес попередньої обробки і аналізу даних. Виконавши аналіз сучасних інформаційних систем, які використовуються в якості головних електронних інструментів в метеорології, було сформовано концепцію розробки програмного засобу.

Таке рішення дозволить розробити сучасну та функціональну систему, а також покращити показники ефективності алгоритмів обробки та аналізу. Так як даних буде багато то програмний комплекс повинен бути оптимізований

для роботи з великими даними, в даному випадку кращим рішенням буде використовувати систему управління базами даних SQLite [19].

Отримані в рамках виконання даного розділу результати будуть застосовані під час практичної реалізації програмного забезпечення.

ВИСНОВКИ

В рамках виконання роботи були закріплені і розширені теоретичні знання, отримані при вивченні дисциплін, розвинені необхідні практичні вміння та навички відповідно до вимог рівня підготовки випускника та затвердженої тематики дослідження.

В результаті отриманих знань, і додатково вивчених матеріалів було проведене дослідження та розроблене програмне забезпечення що дозволяє оцінити ефективність нейромереж у задачах обробки та аналізу метеоданих.

Під час написання роботи, було виконано:

– аналіз того як машинне навчання допомогае вирішувати задачі метеорології;

– аналіз впливу інформаційних технологій на робочі процеси метеорологів;

– обґрунтувано впровадження інформаційних систем в сферу метеорологіх;

– визначення програмні засоби реалізації інформаційної системи;

– проектування інформаційної системи;

– розробку інформаційної системи для сфери метеорології;

– проведено дослідження результатів обробки та аналізу метеоданих на основі різних алгоритмів машинного навчання;

– проведено дослідження того які вхідні параметрі мали найбільший вплив на результат в різних алгоритмах машинного навчання.

Розробку програмного комплексу виконано з використанням мови програмування Python та спеціалізованого фреймворку для роботи з web-системами Django. В якості системи керування базами даних використано SQLite.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1 ДСТУ 8302:2015 «Інформація та документація. Бібліографічне посилання. Загальні положення та правила складання» – Чинний від 04.03.2016. – Київ: УкрНДНЦ, 2020. – 20 с.

2 Антоні С. Опануй самостійно програмування за 21 день / С. Антоні. – Москва: Вильямс, 2021. – 562 с.

3 Аронов І.З. Сучасні проблеми безпеки технічних систем і аналізу ризику / І.З. Аронов. – Москва: Інтуит. 2021 – 451 с.

4 Астелс Д. Керівництво з екстремального програмування / Д. Астелс. – Москва: Вільямс, 2022. – 468 с.

5 Багриновский К.А. Нові інформаційні технології / К.А. Багриновский, Є.Ю. Хрустальов. – Москва: ЕКО, 2019. – 212 с.

6 Гмурман В.Є. Теорія ймовірностей і математична статистика / В.Є. Гмурман. – Москва: Вища. Шк, 2022. – 479с.

7 Бахтізін В.В. Технології розробки програмного забезпечення / В.В. Бахтізін, Л.А. Глухова. – Мінськ: БДУІР, 2020. – 267 с.

8 Бенін Д.М. Системи підтримки прийняття рішень / В.Л. Сніжко, Д.М. Бенін – Москва: Тріада, 2022. – 165 с.

9 Бхаграва А. Грокаем алгоритми / А. Бхаграва – СПб .: ПИТЕР, 2020. – 288с

10 Вірт Н. Алгоритми і структури даних. / Н. Вірт – Москва: Свет, 2019. – 651 с.

11 Владимиров В.А. Управління ризиком / В.А. Владимиров, Ю.Л. Воробйов, Г.Г. Малинецкий. – Москва: ЮНИТИ, 2020. – 352 с.

12 Будума Н. Основи глибокого навчання. Створення алгоритмів для штучного інтелекту наступного покоління / Будума Н. – Москва: ЮНИТИ, 2022 – 520с.

13 Ендрю Т. Грокаем глибоке навчання / Т. Ендрю – СПб.: «БХВ-Петербург», 2021 – 712с

14 Прохоренок Н.А. Python 3 и PyQt 5. Розробка додатків / Н.А. Прохоренок, В.А. Дронов. – СПб.: БХВ-Петербург, 2020. – 832с.

15 Padmanabhan T.R. Programming with Python / T.R. Padmanabhan. – СПб.: Springer, 2020. – 349р.

16 Эрик М. Изучаем Python. Програмування ігор, візуалізація даних. Веб-додаток / М. Эрик. – СПб.: Питер, 2021. – 496с.

17 Себастіан Р. Python і машинне навчання / Р. Себастіан. – Москва: Диалектика, 2021. – 752 с.

18 Бейлин Л. Вивчаемо MySQL / Л. Бейлин. – Москва: Эксмо, 2022. – 1060 с.

19 Lane D. Web Database Application with PHP and MySQL / D. Lane. – New Jersey: O'Reilly, 2021. – 816 p.

20 Прохоренок Н.А. HTML, JavaScript, PHP и MySQL. Джентльменський набір Web-мастера / Н.А. Пархоменко. – СПб.: Питер, 2019. – 768 с.

21 Роббинс Д. HTML5, CSS3 и JavaScript. Повний посібник / Д. Роббинс. – Москва: Эксмо, 2019. – 528 с.

22 Шварц Б. MySQL. Оптимізація продуктивності / Б. Шварц. – Москва: Символ-Плю, 2019. – 483 с.

23 Гольцман В. MySQL 5.0 / В.Гольцман. – Москва: Питер, 2019. – 764 с.

24 Яргер Р.Д. MySQL и mSQL. Бази даних для невеликих підприємств та інтернету/ Р.Д. Яргер. – Москва: Символ-Плюс, 2021. – 929 с.

25 Уильман Л. MySQL. Посібник з вовчення мови / Л.Уильман. – Москва: ДМК Пресс, 2021. – 764 с.

26 Аткинсон Л. MySQL. Бібліотека професіонала / Л.Аткинсон. – Москва: Вильямс, 2021. – 1493 с.

27 Артеменко Ю. Н. MySQL. Довідник з мови / Ю.Н. Артеменко. – Москва: Вильямс, 2021. – 843 с.

28 Никсон Р. «Learning PHP, MySQL, JavaScript, CSS & HTML5 A Step-by-Step Guide to Creating Dynamic Websites» / Р. Никсон. – O'Reilly Media, 2020. – 730 с.

29 Rocha G.D. Learning SQLite for iOS / G.D. Rocha. – NY.: Packt Publishing, 2022. – 579 p.

30 Owens M. The Definitive Guide to SQLite / M. Owens, G. Allen. – NY.: Apress, 2022. – 739 p.