

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА

Факультет прикладної математики та інформатики

Кафедра прикладної математики

Дипломна робота

Використання алгоритмів розпізнавання емоцій для аналізу рівня задоволеності клієнта

Виконав: студент групи ПМП-42
спеціальності

113 - прикладна математика

Лисак Н.С.

(прізвище та ініціали)

Керівник _____
Переймибіда А.А.

(прізвище та ініціали)

Рецензент _____

(прізвище та ініціали)

Львів - 2023

Зміст

Зміст	1
1. Постановка задачі	2
2. Існуючі методи розв'язання задачі	5
2.1. Методи для розпізнавання людських емоцій	5
2.2. Алгоритми розрахунку задоволеності клієнтів	11
3. Програмна реалізація	14
4. Аналіз результатів	19
4.1. Приклад 1. Нейтральна емоція	19
4.2. Приклад 2. Позитивна емоція	20
4.3. Приклад 3. Позитивна емоція (власне фото)	21
4.4. Приклад 4. Негативна емоція	22
5. Висновок	24
6. Список літератури	25

1. Постановка задачі

Дуже часто продуктові іт-компанії не знають куди рухатись, мов вони спіткнулись об стіну і далі вже немає дороги. Компанії не знають, ні задоволеності людей від їхніх продуктів (програм), ні те, що потрібно змінити або покращити. Також, компанії не розуміють чи їхні продукти роблять людей щасливішими чи депресивними.

Для прикладу візьмемо мобільні додатки. Користувачі (users) дуже рідко залишають коментарі в App Store чи Play Market, навіть поставити рейтинг програмі люди не готові поки ти в них не попросиш під час їхнього використання програми. На додачу до цього, не реально зрозуміти стан людини по якихось зірочках (рейтингу програми, який вона ставить). Наша ідея полягає в тому, щоб людина не просто виставляла рейтинг програмі, хоча велика кількість користувачів і цього не роблять, а щоб вона фотографувала себе в реальний час і по фотографії компанії можуть зрозуміти стан людини, її рівень (коефіцієнт) задоволеності або депресії. Наприклад, програма може запросити в користувача (user) зробити знімок свого обличчя (selfie) раз в 3 місяці, тоді людям не буде набридати сповіщення про те, що потрібно сфотографуватись, бо пройде достатня кількість часу, щоб про це забути. Також, компанії зможуть скласти реальну статистику задоволеності клієнта програмою, або зможуть призупинити програму і переглянути ціль програми, якщо коефіцієнт депресії великий.

Ми пропонуємо бібліотеку, яку можуть використовувати в iOS розробники, в перспективі і інші розробники, для реалізації задуманої ідеї. Як воно працює? В юзера під час користування програмою спливає сповіщення, що необхідно зробити знімок обличчя, щоб компанія могла ще більше покращувати програму. Ми беремо цю фотографію і робимо запит в нашого навченого штучного інтелекту *«Яка емоція в людини? Який коефіцієнт задоволеності? Наскільки точно ти (штучний інтелект) оприділив емоцію людини?»*, штучний інтелект

надає нам цю інформацію і ми віддаємо цю інформацію людям, які встановили собі цю бібліотеку. Після цього вже компанії можуть оперувати цими даними.

Сьогодні компанії докладуть чимало зусиль, щоб дізнатися, що клієнти думають про їхній продукт. Від дорогих опитувань до фокус-груп і всього іншого, інформацію про клієнтів традиційно не можна отримати за одну ніч або за обмежений бюджет.

Але оскільки ця інформація зараз є настільки важливою для команд успіху клієнтів, з'явився стандартизований метод отримання справжньої інформації за доступною ціною. Відомий як Індекс підтримки споживача (Net Promoter Score (NPS)), він задає клієнтам одне просте запитання *“За шкалою від 0 до 10, наскільки ймовірно, що ви порекомендуєте мій бренд/продукт/послугу другові чи колезі?”*

Для того, щоб обчислити свій бал NPS, ви віднімаєте відсоток промоутерів і противників. «Пасиви» вважаються нейтральними і не впливають на ваш рейтинг NPS. Наприклад, якщо зі 100 відповідей NPS 38 були промоутерами, 29 були пасивними і 33 були негативними, тоді рівняння виглядало б $38\% - 33\% = 5\%$. Таким чином, ваш загальний бал NPS становить 5. Не дуже хороший результат! [6]

Обчислення NPS



рис. 1.1.

Після того, як ми описали логіку і розрахунки NPS (рис. 1.1.), розкажемо, як ми це можемо використати в себе. Так, як в нас є дані про людину, її емоція, коефіцієнт задоволеності, ми можемо оперувати тим, чи емоція позитивна, чи негативна, чи нейтральна і тоді можемо розрахувати NPS продукту.

Для того щоб розрахувати NPS ми будемо класифікувати емоції на 3 типи на позитивні, негативні, чи нейтральні. В програмі ми визначаємо 7 емоцій, такі як: злість, роздратування (відраза), страх, щастя, нейтральність, сум, здивованість. Тому нам необхідно розподілити емоцій на позитивні, негативні, чи нейтральні. Злість, роздратування, страх, сум віднесемо до негативних емоцій, щастя та здивованість - до позитивних, а нейтральність - до нейтральних. Тоді компанія зможе зберігати в себе інформацію про емоцію користувача і пізніше маючи велику кількість фідбеку від користувачів визначити NPS.

2. Існуючі методи розв'язання задачі

2.1. Методи для розпізнавання людських емоцій

Існують багато методів для розпізнавання людських емоцій. Ми опишемо основні і найбільш точніші методи розпізнавання людських емоцій. Для початку опишемо, що потрібно для кожного методу і як вони працюють.

1. Метод машинного навчання [3]:

- Збираємо велику кількість зображень обличчя людей з різними емоціями та сортуємо їх по емоціях.
- Навчаємо нейронну мережу використовуючи наші зображення.
- Використаємо нашу навчену мережу для розпізнавання емоцій.

2. Метод аналізу геометричних рис.обличчя [3]:

- Розпізнаємо різні частини обличчя (очі, ніс, рот, брови тощо).
- Визначаємо геометричні параметри кожної частини (наприклад, розташування, розмір, кут нахилу тощо).
- Використовуємо ці параметри для визначення емоції.

3. Метод аналізу голосу людини [1]:

- Збираємо велику кількість аудіозаписів людей з різними емоціями.
- Використаємо алгоритми обробки сигналів, щоб визначити параметри звуку, такі як тембр, частота тощо.
- Використовуємо ці параметри для визначення емоції.

4. Метод аналізу тексту [2]:

- Збираємо велику кількість текстів, що містять емоційну інформацію (наприклад, пости в соціальних мережах).
- Використаємо алгоритми обробки текстів, щоб визначити емоційну забарвленість тексту (наприклад, за допомогою аналізу тональності тексту).
- Використовуємо ці параметри для визначення емоції.

Звичайно, кожен з цих алгоритмів має свої переваги та недоліки, і їх краще використовувати у разі для отримання більш точних результатів.

На додачу до цього, існують готові бібліотеки, які можуть допомогти в реалізації розпізнавання емоцій. Наведемо для прикладу 2 бібліотеки, про які ми розкажемо більше далі, перша бібліотека OpenCV має вбудовані алгоритми для розпізнавання облич та емоцій на основі методів геометричного аналізу, а друга бібліотека TensorFlow має набір готових моделей для розпізнавання емоцій на основі методів глибокого навчання.

Звичайно кожен алгоритм має свої переваги та недоліки. Опишемо для кожного метода декілька з них. Наприклад, метод машинного навчання має такі недоліки, як потреба великої кількості даних для навчання моделі та буває складний у використанні та налаштуванні, а його переваги в тому, що він є точним і надійним, якщо модель правильно налаштована та має достатньо велику збірку даних для навчання. Також він є універсальним, оскільки може розпізнавати емоції за допомогою різних типів даних, таких як аудіо, зображення та відео. Тепер опишемо для методу геометричного аналізу облич, він має такі недоліки, як те, що на результат можуть вплинути різні фактори, такі як освітлення, поза, яку приймає людина та те, що метод вимагає точного вирізання облич з зображення, що може бути складним завданням, якщо на зображенні присутні інші об'єкти, а переваги цього методу в тому, що він точний для розпізнавання певних емоцій, особливо якщо мова йде про стандартні вирази обличчя та він може бути ефективним у випадку, якщо дані відносно невербальних знаків є недоступними. Також опишемо недоліки та переваги методу аналізу голосу людини, його недоліками є неможливість розпізнати емоції, які виражаються тільки за допомогою невербальних знаків та на його результат можуть вплинути різні фактори, такі як гучність мовлення, акцент, дикція, а перевагами є те, що він швидкий та легкий у використанні та може давати точні результати для високоякісних аудіозаписів. А метод аналізу тексту може мати проблеми з розумінням сарказму та іронії, що може призвести до неправильної інтерпретації емоцій та емоційні відтінки у тексті можуть бути залежними від контексту, що може бути складним для аналізу, а його переваги

це можливість обробляти великі обсяги текстових даних за короткий час, що робить його ефективним для масштабних завдань, можливість надати високу точність виявлення емоцій, оскільки текстові повідомлення часто містять чітку інформацію про емоційний стан автора та його можна використати для аналізу емоцій на різних мовах, що робить його універсальним методом.

TensorFlow - це відкрите програмне забезпечення для машинного навчання, розроблене Google. Бібліотека TensorFlow містить інструменти для розробки та навчання глибоких нейронних мереж. TensorFlow можна використовувати для різноманітних завдань машинного навчання, включаючи розпізнавання образів, розпізнавання мови, аналіз тексту та якраз для розпізнавання емоцій, того що потрібно для нас.

Опишемо основні переваги та недоліки бібліотеки TensorFlow. Перший перевага в тому, що бібліотека надає відкрите програмне забезпечення з відкритим кодом, що дозволяє користувачам вільно використовувати та змінювати його за своїми потребами. Наступна перевага це те, що TensorFlow працює дуже швидко та ефективно завдяки використанню векторних операцій та графів обчислень. Також, TensorFlow має багато інструментів та функцій для розробки та навчання глибоких нейронних мереж на різноманітних завданнях машинного навчання. Тепер перейдемо до недоліків бібліотеки TensorFlow. Перший і найсуттєвіший недолік нашої бібліотеки це те, що TensorFlow може бути досить складним у використанні для початківців і людей, які не знайомі з машинним навчанням. Також, TensorFlow може мати обмеження в роботі з графічними процесорами (GPU) залежно від моделі та задачі. І не забудемо те, що TensorFlow може мати обмеження в доступній пам'яті, особливо при обробці великих об'ємів даних, що може вплинути на ефективність роботи.

Розповідаючи про TensorFlow слід ще згадати за основні алгоритми розпізнавання. TensorFlow містить багато алгоритмів розпізнавання, які можуть бути використані залежно від завдання. Один з цих алгоритмів це кероване навчання (Supervised learning): цей алгоритм використовує мітки для навчання моделі, щоб вона могла відповісти на питання. Наприклад, якщо потрібно розпізнати рукописні цифри, модель використовуватиме набір даних з

рукописними цифрами та відповідними мітками, щоб навчитися розпізнавати цифри на зображеннях. Ще є алгоритм, який називається підсилене навчання (Reinforcement learning) цей алгоритм використовується для навчання моделі, щоб приймати рішення в середовищі, з метою максимізації винагороди. Також є алгоритм, який називається некероване навчання (Unsupervised learning) цей алгоритм не використовує мітки для навчання моделі. Він зазвичай використовується для кластеризації даних, тобто групування даних за схожими ознаками.

На додачу до цього, TensorFlow має багато готових моделей для різних задач машинного навчання, таких як розпізнавання образів, обробка природної мови та інші, які можна використати без необхідності написання власних алгоритмів [3].

OpenCV - це бібліотека відкритого коду для комп'ютерного зору та обробки зображень. Вона написана на C++ та має багато інтерфейсів для різних мов програмування, включаючи Python, Java та MATLAB.

OpenCV містить багато інструментів для роботи з зображеннями та відео, такі як зчитування та запис зображень, обрізання, збільшення або зменшення розміру зображення, фільтрація зображення, виявлення країв, розпізнавання об'єктів, вимірювання відстані, а також камери, що забезпечують розпізнавання облич, відслідковування об'єктів та інші. Також OpenCV можна використати для розробки різноманітних додатків, включаючи системи безпеки, медичні застосування, розпізнавання облич, відеоспостереження, роботи з даними зі сканерів та інші.

Однією з переваг OpenCV є те, що вона є вільною та розповсюджується під ліцензією BSD (Berkeley Software Distribution license), що дозволяє використовувати та розповсюджувати її безкоштовно, включаючи комерційні проекти. Також, OpenCV є велика та активна спільнота, яка розвиває та підтримує бібліотеку, а також надає документацію та приклади коду для використання бібліотеки. Недоліком OpenCV може бути складність використання для новачків у комп'ютерному зорі та обробці зображень, а також потреба в високій обчислювальній потужності для деяких застосувань [4].

Також необхідно згадати за фреймворк Apple's Core ML, який ми використали для розробки нашого додатку. Apple's Core ML - це фреймворк для розробки мобільних додатків на iOS, який дозволяє інтегрувати моделі машинного навчання безпосередньо в додатки. Core ML був запущений в 2017 році на конференції WWDC (Worldwide Developers Conference) і з того часу він став популярним інструментом серед розробників додатків для iOS. Також ця платформа дозволяє розробникам розгорнути потужні ML Model на пристрої, повністю використовуючи переваги уніфікованого представлення для всіх моделей.

Якщо говорити точніше, Core ML розроблено для оптимізації продуктивності для роботи на пристрої, дозволяючи розробникам вибирати з широкого спектру ML Model, які вони можуть розгорнути на обладнанні Apple, яке вже постачається зі спеціальними нейронними механізмами та прискорювачами ML.

Основні переваги Core ML це висока швидкість та продуктивність, що дозволяє запускати моделі машинного навчання навіть на пристроях з обмеженими ресурсами. Надійний захист інформації, оскільки дані не відправляються на сервери Apple або третіх сторін. Простота використання і інтеграції моделей машинного навчання в додатки. Також підтримка глибоких нейронних мереж та інших типів моделей машинного навчання. Однак, є деякі недоліки, які суттєво впливають на розробку. Перший недолік в тому, що Core ML доступний лише для платформи iOS, тому використовувати його можна тільки для розробки додатків для цієї платформи. Обмежена кількість моделей машинного навчання, які підтримуються фреймворком. Також, необхідність перетворення моделей машинного навчання в формат, який підтримує Core ML.

Також необхідно розказати за ML Model (Machine Learning Model). ML Model - це математичний алгоритм, який вивчається на основі історичних даних та дозволяє автоматично здійснювати прогнози та приймати рішення без явного програмування. Моделі машинного навчання можуть бути навчені на різних типах даних, таких як числові дані, зображення, текст, звук та інші, і

використовуватись для різних завдань, таких як класифікація, регресія, кластеризація, генерація тексту, машинний переклад та інші.

Для створення ML-моделі потрібно зібрати відповідний набір даних, обробити його та підготувати до навчання моделі. Після цього модель можна навчити на даних, використовуючи різні методи машинного навчання, такі як нейронні мережі, дерева рішень, логістична регресія та інші. Після навчання моделі вона може бути використана для прогнозування результатів на нових даних.

ML-моделі можуть мати переваги, такі як точність, ефективність та здатність до автоматизації процесів, що дозволяє зменшити витрати на робочу силу та збільшити продуктивність. Проте, ML-моделі також можуть мати недоліки, такі як нестабільність, відсутність розуміння причинно-наслідкових зв'язків, проблеми з поясненням результатів та інші.

В цілому, ML-моделі є корисним інструментом для рішення складних завдань, що дозволяє здійснювати прогнозування та приймати рішення на основі даних з високою точністю та ефективністю [9].

2.2. Алгоритми розрахунку задоволеності

клієнтів

Окрім NPS існують ще кілька алгоритмів. Їх можна використовувати разом з NPS для отримання більш повної картини задоволеності клієнтів або окремо. Опишемо ці алгоритми.

Перший алгоритм це CSAT (Customer Satisfaction Score). CSAT - це метод оцінки задоволеності клієнтів, який вимірює, наскільки клієнти задоволені конкретним продуктом, послугою або досвідом. Клієнтам пропонується оцінити продукт або послугу за шкалою від 1 до 5 або від 1 до 10. Чим вище рейтинг, тим більше клієнти задоволені продуктом або послугою [5].

Наступний алгоритм це CES (Customer Effort Score). Цей метод вимірює, наскільки легко клієнти можуть взаємодіяти з компанією для отримання послуги або підтримки. Клієнтам пропонується оцінити, наскільки легко їм було вирішити свою проблему або отримати допомогу, за шкалою від 1 до 5 або від 1 до 10. Чим менше зусиль вони витрачають, тим вище рейтинг [7].

Також є алгоритм, який називається CLV (Customer Lifetime Value). CLV - це метод вимірювання потенційної прибутковості клієнта для компанії на протязі всього періоду взаємодії. Цей показник враховує, наскільки часто клієнт здійснює покупки, скільки він витрачає на кожну покупку та скільки часу він залишається вірним клієнтом компанії [8].

Звичайно ці алгоритми можна використовувати окремо, але для більш повної картини їх краще використовувати разом. Ці алгоритми можуть доповнити NPS, дозволяючи компаніям отримати більш повну картину про задоволеність та лояльність клієнтів, а також визначити, які аспекти їхньої роботи потребують покращення.

Також було б добре описати переваги та недоліки кожного з методу. Почнемо з методу NPS. Його переваги в тому, що він простий та його легко зрозуміти. Він має високі кореляції з реальною лояльністю та повторними покупками клієнтів. NPS дозволяє порівняти результати з іншими компаніями, що використовують той самий метод. А його недоліки в тому, що він часто

використовується як єдиний метод оцінки клієнтів, не дозволяючи отримати більш детальну картину задоволеності та лояльності. Також він не розкриває причини низької оцінки та не надає додаткової інформації для подальших вдосконалень [6].

Переваги та недоліки CSAT. Переваги методу CSAT в тому, що дозволяє отримати докладну інформацію про те, що саме впливає на задоволеність клієнтів та як її можна покращити. Він надає більш детальну картину, ніж NPS, що дозволяє компанії більш точно налаштувати свої стратегії. Його недоліки це менш корисний для вимірювання лояльності клієнтів, оскільки він не забезпечує інформації про можливість рекомендації компанії друзям та знайомим. Також його результати можуть бути спотворені, якщо під час опитування ставляться некоректні або недостатньо точні запитання [5].

Переваги та недоліки методу CES. Він має декілька переваг такі, як дозволяє компанії зрозуміти, наскільки складним та часовим затратним є процес взаємодії з компанією. Він надає додаткову інформацію про те, як полегшити взаємодію з клієнтами та зменшити їхні зусилля. Також він може бути більш корисним за NPS або CSAT особливо для тих компаній, які надають послуги. А його недоліки полягають в тому, що він є недостатньо ефективний для вимірювання лояльності та здатності клієнтів рекомендувати компанію друзям та знайомим. Та він вимірює лише один аспект взаємодії з компанією, тому не може замінити більш комплексний підхід до оцінки клієнтів [7].

Переваги та недоліки методу CLV. Переваги методу CLV полягають в тому, що він дозволяє компанії зрозуміти, які клієнти є найбільш цінними та як їх можна зберегти на довгій перспективі. Також, він надає більш глибоку та комплексну інформацію про клієнтів, що дозволяє розробляти більш ефективні стратегії управління взаємодією з ними. А його недоліки в тому, що він вимагає значної кількості даних та складних аналітичних методів, щоб розрахувати. Він не дозволяє компанії отримати докладну інформацію про те, як саме покращити задоволеність та лояльність клієнтів [8].

Отже, кожен з цих методів має певні переваги та недоліки. NPS є простим та легким у використанні методом, який дозволяє порівняти результати з

іншими компаніями, а CSAT та CES надають докладну інформацію про те, що саме впливає на задоволеність та легкість взаємодії з компанією. CLV дозволяє компанії зосередитись на найбільш цінних клієнтах. Навівши переваги і недоліки кожного з методів тепер більш зрозуміло чому їх варто використовувати разом, а не в окремо.

3. Програмна реалізація

Найважливіший аспект при створенні мобільного додатку це вирішити, який архітектурний патерн ми будемо використовувати. Є 4 основні архітектурні патерни для мобільної розробки. Перший - це MVC (Model-View-Controller). Архітектура патерну MVC (рис. 2.1.) є однією з найпоширеніших архітектур для розробки програмного забезпечення. Ця архітектура розбиває додаток на три основні компоненти: модель (Model), представлення (View) та контролер (Controller).

Модель (Model) відповідає за представлення даних та бізнес-логіки програми. Модель зберігає дані та забезпечує доступ до них. Це можуть бути дані з бази даних, файлів або з інших джерел.

Представлення (View) відповідає за відображення даних користувачу. Воно може бути візуальним інтерфейсом користувача, який відображає дані у вигляді таблиць, графіків, текстів та інших елементів. Представлення може також обробляти вхідні дані користувача.

Контролер (Controller) відповідає за обробку даних та реагування на події, які створюються користувачем або іншими джерелами. Контролер забезпечує взаємодію між моделлю та представленням. Він обробляє запити користувача, змінює стан моделі та оновлює представлення.

У архітектурі MVC модель, представлення та контролер мають свої відокремлені обов'язки та не залежать один від одного. Це робить код більш модульним та дозволяє змінювати одну складову без впливу на інші. Це також сприяє збільшенню перевикористання коду та зменшенню кількості дублювання коду.

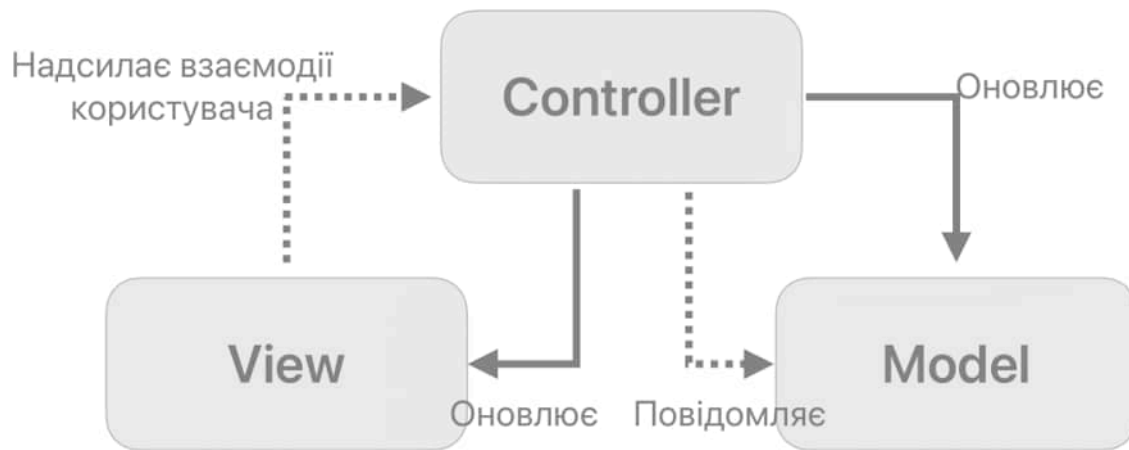


рис. 2.1.

Другий патерн - це MVP (Model-View-Presenter). MVP - це патерн, в якому з'являється додатковий компонент, Presenter, який відповідає за місткість між моделлю та представленням. Presenter взаємодіє з View, виконуючи роль посередника між View та Model. Цей патерн дозволяє розділити логіку бізнес-логіки та логіку відображення на різні компоненти та підвищує тестуваність додатку.

Третій патерн - це MVVM (Model-View-ViewModel). MVVM - це патерн, який використовує ViewModel як додатковий компонент. ViewModel є проміжним етапом між Model та View, і відповідає за конвертацію даних з Model в формат, зрозумілий для View. Це дозволяє відділити представлення та логіку додатку, що дозволяє забезпечити більшу гнучкість та покращує тестування.

Четвертий патерн - це VIPER. VIPER - це архітектурний патерн, який складається з п'яти компонентів: View, Interactor, Presenter, Entity та Router. Кожен з цих компонентів виконує певну роль у проекті, дозволяючи розділити логіку додатку на маленькі, незалежні від інших частин. VIPER дозволяє розділити логіку додатку на різні слої та забезпечує їх взаємодію з мінімальною залежністю між ними.

Apple відразу при створенні програми пропонує користувачеві використовувати архітектурний патерн MVC. Через те, що він більш зрозумілий для початківців та легший в реалізації. Так, як в нас програма не є великою, то ми вирішили використати патерн MVC [10].

Далі ми розкажемо про ARKit. ARKit - це фреймворк, розроблений компанією Apple для розробки додатків з розширеною реальністю (AR) на платформі iOS. ARKit дозволяє розробникам створювати додатки, які можуть відтворювати віртуальні об'єкти в реальному світі, використовуючи камеру та датчики пристрою.

Для розробки додатків з використанням ARKit вам знадобиться Xcode та мова програмування Swift. Основна ідея ARKit полягає в тому, що він допомагає визначити місцезнаходження та орієнтацію пристрою в просторі, що дозволяє розробникам створювати віртуальні об'єкти, які можуть взаємодіяти з реальним світом.

А головне і найвагомніше для нашого проекту це те, що ARKit також надає можливість розпізнавати інші об'єкти, такі як обличчя та руки, та здійснювати взаємодію з ними [11].

В ARKit ми використовуємо модель VNCoreMLModel. VNCoreMLModel - це обгортка для моделей машинного навчання, побудованих з використанням фреймворку Core ML в Swift. Ця бібліотека дозволяє інтегрувати моделі машинного навчання в додатки для обробки зображень та відео. Ми нашу модель, VNCoreMLModel, ініціалізуємо нашою моделлю машинного навчання для розпізнавання емоцій. Ми використаємо загально-доступну модель машинного навчання Core ML для розпізнавання емоцій [15].

Пройдемося поетапно по програмі. Перший етап, ми створили екран на якому є UILabel, UIImageView і UIButton, всі ці і інші класи є доступними в бібліотеці UIKit. UIKit є однією з ключових бібліотек для розробки інтерфейсу користувача (UI) для додатків iOS на мові програмування Swift або Objective-C. Бібліотека UIKit надає розробникам доступ до різноманітних інструментів і функцій, що дозволяють створювати високоякісний та привабливий інтерфейс для своїх додатків. UILabel ми будемо виводити, яка в людини емоція, в UIImageView ми будемо зберігати фотографію, а UIButton ми використовуємо для того, щоб людина змогла вибрати або зробити фотографію.

Другий етап, після того, як користувач натиснув на наш UIButton нам потрібно дати можливість йому вибрати, чи хоче він вибрати фотографію з

галереї, чи хоче він зробити фотографію. Також, необхідний аспект є в тому, що системи Apple є дуже конфіденційні, тому нам необхідно запитати дозвіл в користувача на доступ до галереї фотографій та до камери. Після того, як користувач надав доступ до цих даних, ми використаємо ще один клас з бібліотека UIKit, а саме UIAlertController, це такий екран який з'явиться поверху основного екрану і ми додали 3 події на цьому екрані [14]. Перша подія - це якщо користувач хоче сфотографуватись, друга подія - це якщо користувач хоче вибрати фото з фото галереї, а третя подія - це якщо користувач хоче повернутись до попереднього екрану. Щоб відкрити камеру або фото галерею використаємо клас UIImagePickerController, це екран, який вже має свій дизайн і реалізований [13]. Після того, як ми вибрали фотографію або зробили фото, нам повернеться деяка інформація про фото і саме фото. А повернеться, бо ми підписались на події цього екрану і відповідно до певної події, ми будемо виконувати певний дії.

Третій етап, після того, як ми отримали наше фото, ми починаємо опрацьовувати його. Ми беремо це фото і робимо піксельний буфер фотографії. Пізніше, ми ще використовуємо клас CIDetector для перевірки, чи є обличчя на фотографії. CIDetector є частиною фреймворка CoreImage в Swift, і він використовується для виявлення об'єктів на зображенні [12][17]. І ми перевіряємо, якщо є хоча б одне обличчя на фотографії, тоді будемо розпізнавати емоції. Для того, щоб розпізнати емоцію користувача ми використовуємо нашу модель машинного навчання та VNImageRequestHandler. VNImageRequestHandler - це клас, що надає інтерфейс для обробки зображень в рамках фреймворку Vision у Swift. Для обробки зображень з використанням Vision API, спочатку потрібно створити екземпляр класу VNImageRequestHandler. Ініціалізуємо наш VNImageRequestHandler і тепер, коли у нас є екземпляр VNImageRequestHandler, ми можемо передати нашу модель машинного навчання у метод perform() VNImageRequestHandler та отримали результати у вигляді масиву VNClassificationObservation [16]. В нас масив складається з 7 елементів і в кожному елементі в нас є така інформація, як назва емоції та відповідна впевненість (confidence) моделі цієї емоції. Також,

ми перевіряємо чи наша впевненість більша за 0.7 і якщо так, тоді ми користувачу показуємо назву емоції, а якщо менше, тоді ми показуємо такий текст “Can't recognize” (“Не можу розпізнати”)

4. Аналіз результатів

4.1. Приклад 1. Нейтральна емоція



рис. 3.1

Emotions	Confidence
Angry	0,016754
Disgust	0,079251
Fear	0,011009
Happy	0,189331
Neutral	0,702637
Sad	0,002333
Surprise	0,007866

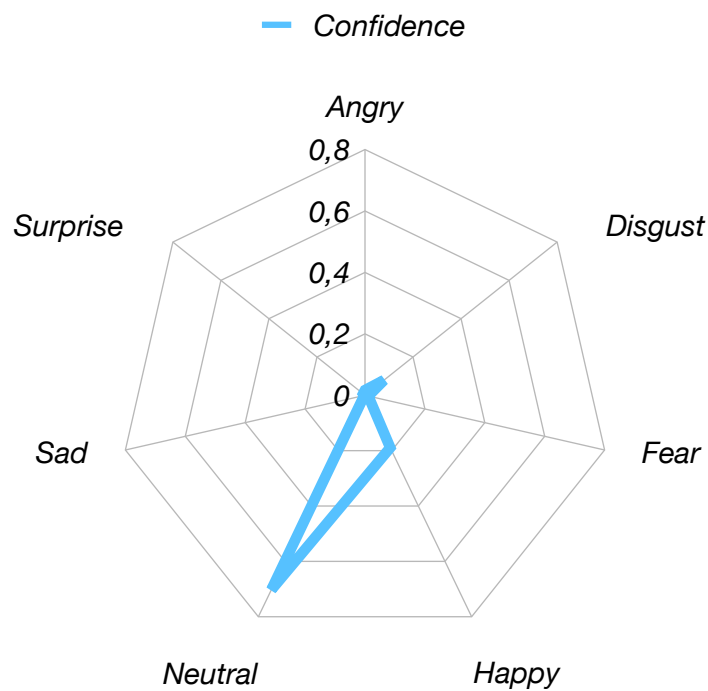


рис. 3.2.

Для нашого прикладу ми вибрали рис. 3.1. на якому зображений чоловік і по фотографії зрозуміло, що його емоція є нейтральною. Фотографія кольорова та її розширення 540 x 810. Ми добавили це фото в нашу програму і запустили її і програма нам показала, що наш штучний інтелект більш впевнений, що його

емоція є нейтральною. Також ми виписали впевненості в певній емоції нашого штучного інтелекту в табличку та зобразили на графіку (рис. 3.2.).

4.2. Приклад 2. Позитивна емоція



рис. 4.1.

Emotions	Confidence
Angry	0,000804
Disgust	0,000217
Fear	0,000804
Happy	0,883301
Neutral	0,114075
Sad	0,000590
Surprise	0,000089

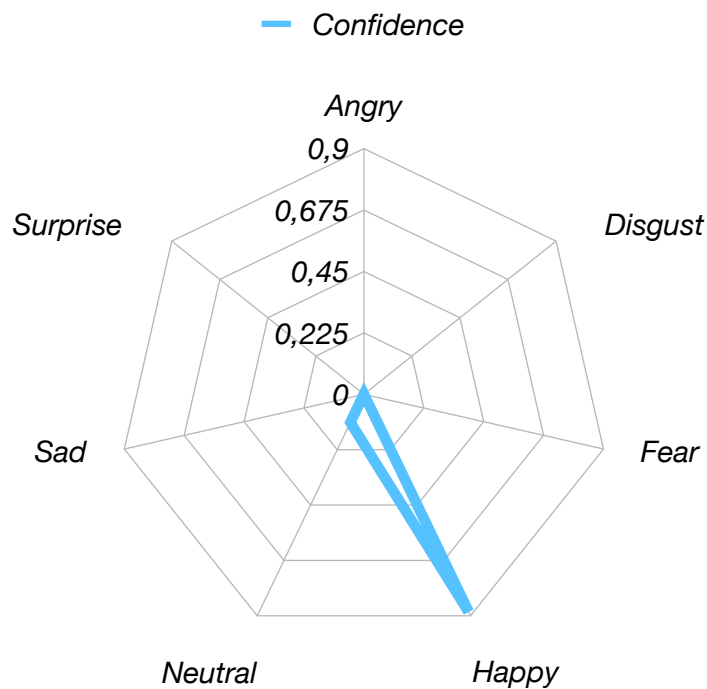


рис. 4.2.

Для нашого прикладу ми вибрали рис. 4.1. на якому зображена дівчина і по фотографії зрозуміло, що вона є щасливою. Саме фото чорно-біле, а розширення фотографії 910 x 1274. Ми добавили це фото в нашу програму і запустили її. Програма нам показала, що наш штучний інтелект більш

впевнений, що дівчина є щасливою. Також ми виписали впевненості в певній емоції нашого штучного інтелекту в таблицку та зобразили на графіку (рис. 4.2.).

4.3. Приклад 3. Позитивна емоція (власне фото)



Emotions	Confidence
Angry	0,013649
Disgust	0,009964
Fear	0,002947
Happy	0,742188
Neutral	0,227783
Sad	0,001850
Surprise	0,001995

рис. 5.1.

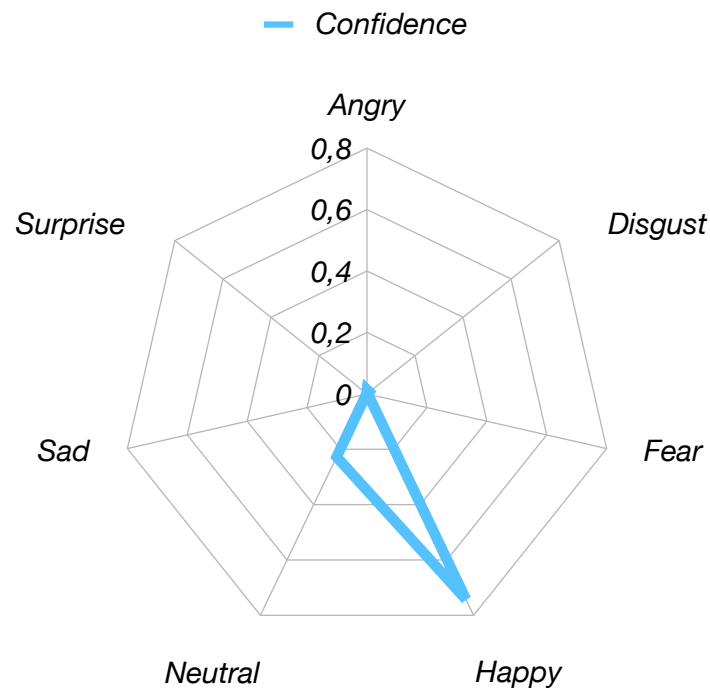


рис. 5.2.

Для нашого прикладу ми зробили власне фото (рис. 5.1.) на якому зображений хлопець і по фотографії зрозуміло, що він є щасливим. Фотографія

кольорова та її розширення 1194 x 1603. Ми добавили це фото в нашу програму і запустили її і програма нам показала, що наш штучний інтелект більш впевнений, що користувач є щасливим. Також ми вписали впевненості в певній емоції нашого штучного інтелекту в табличку та зобразили на графіку (рис. 5.2.).

4.4. Приклад 4. Негативна емоція



Emotions	Confidence
Angry	0,852051
Disgust	0,003874
Fear	0,000548
Happy	0,136353
Neutral	0,000027
Sad	0,007561
Surprise	0,000101

рис. 6.1.

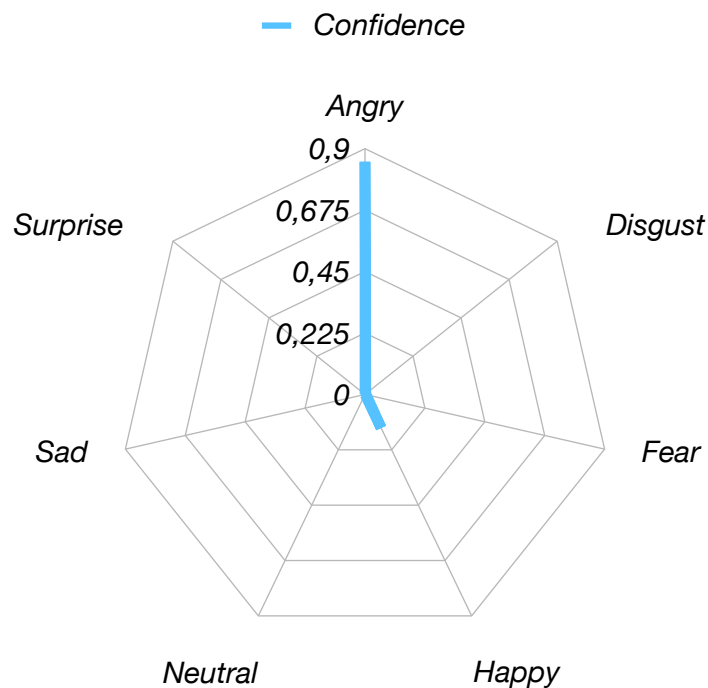


рис. 6.2.

Для нашого прикладу ми вибрали рис. 6.1. на якому зображений чоловік і по фотографії зрозуміло, що він є злим. Фотографія кольорова та її розширення 800 x 533. Ми добавили це фото в нашу програму і запустили її і програма нам показала, що наш штучний інтелект більш впевнений, що користувач є злий в даний момент. Також ми виписали впевненості в певній емоції нашого штучного інтелекту в таблицку та зобразили на графіку (рис. 6.2.).

5. ВИСНОВОК

Розроблена програма, яка по фотографії користувача буде визначати його емоцію. Робота опублікована на <https://gitlab.com> <https://gitlab.com/lnazars67/emotion-recognition>. Компанії зможуть використовувати нашу програму для визначення задоволеності клієнта по його емоційному стані. Також компанії зможуть зберігати в себе дані про всіх користувачів і за допомогою нашої програми визначати рух своєї програми. Наприклад, якщо в великого відсотка користувачів негативні емоції, то компанія може взяти це до уваги і почати змінювати програму, її інтерфейс чи саму реалізацію програми. А якщо в користувачів вбільшості позитивні емоції, то компанія може це використати для популяризації продукту, який робить користувачів щасливішими.

Також, в нашій роботі описано, як працює наша програма і додаткові методи реалізації розпізнавання емоцій, які теж в майбутньому можна реалізувати для кращої картини та більш точних даних стосовно продукту. На додачу до цього, реалізовано тільки для iOS програм, але як перспектива нашу ідею можна реалізувати і для Android програм. Додатково варто виділити те, що в майбутньому наш алгоритм і ідею можна використати для окремої бібліотеки, яку розробники зможуть собі підключати і в певний момент програми її використовувати. Наприклад, з'явиться спливаюче вікно, де програма попросить користувача зробити фото (selfie), користувач зможе відхилити цю пропозицію або зробити фото, після цих двох дій користувач далі може користуватись програмою, а сама програма після того, як користувач зробив фото надішле ці дані на сервер і по цих даних вже можна буде зробити графік задоволеності. Також, якщо емоція користувача була негативною, то можна попросити його перейти по посиланню і в певній формі описати недоліки програми і на що варто звернути увагу.

6. Список літератури

1. Librosa // <https://librosa.org/doc/latest/index.html> (дата звернення: 15.11.2022)
2. Applied Text Mining in Python // <https://www.coursera.org/learn/python-text-mining> (дата звернення: 15.11.2022)
3. TensorFlow // <https://www.tensorflow.org/> (дата звернення: 15.11.2022)
4. OpenCV // <https://opencv.org/> (дата звернення: 15.11.2022)
5. Naveen Nair. What is CSAT Score and How to Boost it Using Chatbots // <https://customerthink.com/what-is-csat-score-and-how-to-boost-it-using-chatbots/> (дата звернення: 23.12.2022)
6. Sonika Mehta. Why Measure Net Promoter Score (NPS)? // <https://customerthink.com/why-measure-net-promoter-score-nps/> (дата звернення: 23.12.2022)
7. Martin Powton. What is Customer Effort Score? // <https://customerthink.com/what-is-customer-effort-score/> (дата звернення: 23.12.2022)
8. Mary Kay Evans. Customer Lifetime Value: The Metric that Helps Build Customer Loyalty // <https://customerthink.com/customer-lifetime-value-the-metric-that-helps-build-customer-loyalty/> (дата звернення: 23.12.2022)
9. Ray Wenderlich // <https://www.kodeco.com/books/machine-learning-by-tutorials/v2.0/chapters/1-machine-learning-ios-you> (дата звернення: 21.01.2023)
10. Hacking with Swift // <https://www.hackingwithswift.com/articles/41/mvc-vs-mvvm-vs-viper-which-to-use-for-ios> (дата звернення: 05.02.2023)
11. Apple documentation. ARKit // <https://developer.apple.com/documentation/arkit> (дата звернення: 05.02.2023)
12. Apple documentation. CIDetector // <https://developer.apple.com/documentation/coreimage/cidetector> (дата звернення: 05.02.2023)
13. Apple documentation. UIImagePickerControllerController // <https://developer.apple.com/documentation/uikit/uiimagepickercontroller> (дата звернення: 05.02.2023)
14. Apple documentation. UIAlertController // <https://developer.apple.com/documentation/uikit/uialertcontroller> (дата звернення: 05.02.2023)

15. Apple documentation. VNCoreMLModel // <https://developer.apple.com/documentation/vision/vncoremlmodel> (дата звернення: 05.02.2023)
16. Apple documentation. VNImageRequestHandler // <https://developer.apple.com/documentation/vision/vnimagerequesthandler> (дата звернення: 05.02.2023)
17. Apple documentation. Core Image // <https://developer.apple.com/documentation/coreimage> (дата звернення: 05.02.2023)