МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КРИВОРІЗЬКИЙ ДЕРЖАВНИЙ ПЕДАГОГІЧНИЙ УНІВЕРСИТЕТ
Фізико-математичний факультет
Кафедра інформатики та прикладної математики

«Допущено до захисту»
Завідувач кафедри
_____ Моісеєнко Н.В. «___» _____ 2024 р.
«___» _____ 2024 р.
Реєстраційний № _____

ПРОТОТИП ДЕТЕКТОРА БРЕХНІ НА ПЛАТФОРМІ ARDUINO

Кваліфікаційна робота студента групи І-20
ступінь вищої освіти «бакалавр»
спеціальності
014 Середня освіта (Інформатика)
Правицького Станіслава Вячеславовича
Керівник Мерзликін П.В.

Оцінка:
Національна шкала _____
Шкала ECTS ____ Кількість балів _____
Голова ЕК: _____ _____
(підпис) (прізвище та ініціали)
Члени ЕК _____ _____
(підпис) (прізвище та ініціали)
_____ _____
(підпис) (прізвище та ініціали)
_____ _____
(підпис) (прізвище та ініціали)

Кривий Ріг – 2024

ЗАПЕВНЕННЯ

Я, Правицький Станіслав Вячеславович, розумію і підтримую політику Криворізького державного педагогічного університету з академічної доброчесності. Запевняю, що ця кваліфікаційна робота виконана самостійно, не містить академічного плагіату, фабрикації, фальсифікації. Я не надавав(ла) і не одержував(ла) недозволену допомогу під час підготовки цієї роботи. Використання ідей, результатів і текстів інших авторів мають покликання на відповідне джерело.

Із чинним Положенням про запобігання та виявлення академічного плагіату в роботах здобувачів вищої освіти Криворізького державного педагогічного університету ознайомлений(а). Чітко усвідомлюю, що в разі виявлення у кваліфікаційній роботі порушення академічної доброчесності робота не допускається до захисту або оцінюється незадовільно.

ЗМІСТ

ВСТУП

У сучасному світі, де технології розвиваються надзвичайно швидко, а інформація набуває дедалі більшого значення, питання виявлення неправди стає вкрай важливим. Достовірність інформації може мати значний вплив на соціальні, політичні та економічні процеси, що робить ефективні та доступні методи детекції брехні надзвичайно актуальними. Використання платформи Arduino для створення прототипу детектора брехні відкриває нові можливості у цій сфері, поєднуючи інноваційні технології з доступністю та гнучкістю.

Метою даної курсової роботи є розробка та вивчення прототипу детектора брехні на основі платформи Arduino. Це передбачає аналіз можливостей платформи Arduino для створення недорогих і ефективних пристроїв, здатних виявляти обман.

Завдання роботи

1. Дослідити теоретичні основи методів виявлення брехні.
2. Провести аналіз можливостей платформи Arduino як бази для прототипу.
3. Розробити та сконструювати прототип детектора брехні.
4. Програмувати та налаштувати необхідний функціонал прототипу.
5. Провести тестування прототипу та аналіз отриманих результатів.

Об'єктом дослідження є процес виявлення брехні за допомогою електронних засобів, зокрема з використанням платформи Arduino.

Предметом дослідження виступає прототип детектора брехні, створений на базі платформи Arduino, його технічні характеристики, принципи роботи та ефективність у різних умовах.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ ДЕТЕКТОРІВ БРЕХНІ.

1.1. Етапи розвитку детекторів брехні

Детектори брехні, або поліграфи, мають довгу історію розвитку, що налічує понад століття. Їх еволюція проходила через кілька ключових етапів, кожен з яких відзначався значними технологічними та методологічними досягненнями.

Перші спроби виявлення брехні

Перші спроби розробки пристроїв для виявлення брехні датуються кінцем XIX століття. Одним з піонерів у цій галузі був італійський лікар Чезаре Ломброзо, який у 1895 році створив один з перших пристроїв для виявлення брехні, заснований на зміні кров'яного тиску. Ломброзо вважав, що під час брехні у людини змінюється фізіологічний стан, що можна зафіксувати за допомогою відповідних приладів.

Інші ранні спроби включають роботу Вільяма Марстона, американського психолога, який у 1915 році розробив метод вимірювання артеріального тиску як індикатора емоційної напруги. Ця методика, хоча і мала обмежений успіх, стала важливим кроком у розвитку сучасних поліграфів. Марстон вважав, що людина, яка бреше, переживає сильні емоції, які можна виявити за допомогою фізіологічних змін.

Розвиток поліграфа у XX столітті

У 1921 році Джон Ларсон, поліцейський з Берклі, Каліфорнія, створив перший багатофункціональний поліграф, який вимірював кілька фізіологічних параметрів одночасно, включаючи кров'яний тиск, пульс і дихання. Цей пристрій значно перевершував попередні розробки і став основою для сучасних поліграфів. Ларсон запропонував, що комплексний підхід до вимірювання кількох параметрів одночасно дозволяє більш точно визначати брехню [1].

Великий внесок у розвиток поліграфів зробив Леонард Кілер, який у 1938 році удосконалив поліграф Ларсона, додавши до нього вимірювач електрошкірної активності. Це дозволило ще точніше виявляти брехню, а також розробити стандартизовану методику проведення поліграфічних тестів, яка використовується й досі. Кілер також експериментував з різними типами питань, що ставляться під час тестування, щоб мінімізувати можливість обману пристрою.

У 1960-х роках розвиток поліграфів продовжився з впровадженням нових технологій, таких як комп'ютеризовані системи обробки даних. Це дозволило значно підвищити точність і надійність поліграфічних тестів. Сучасні поліграфи можуть аналізувати не лише фізіологічні показники, а й інші фактори, такі як мікровирази обличчя, зміни в голосі та навіть нейронні відповіді. Наприклад, комп'ютеризовані поліграфи здатні автоматично аналізувати результати тестування і виявляти аномалії, які можуть свідчити про брехню.

У 1980-х роках з'явилися перші спроби використання штучного інтелекту в поліграфології. Дослідники почали розробляти алгоритми, які могли б аналізувати фізіологічні дані та робити висновки про правдивість відповідей. Ці алгоритми використовували машинне навчання для вдосконалення своєї точності на основі великих обсягів даних.

Сучасні досягнення та технології

Сучасні поліграфи значно відрізняються від своїх попередників завдяки впровадженню новітніх технологій. Використання комп'ютерних систем дозволило значно підвищити точність і надійність поліграфічних тестів. Сьогодні поліграфи можуть аналізувати не лише фізіологічні показники, а й інші фактори, такі як мікровирази обличчя, зміни в голосі та навіть нейронні відповіді. Завдяки цьому поліграфи стали незамінним інструментом у розслідуваннях, наймі на роботу та інших сферах, де важливо виявляти неправдиву інформацію.

Однією з найновіших технологій у цій галузі є використання нейротехнологій. Наприклад, функціональна магнітно-резонансна томографія (фМРТ) дозволяє виявляти активність певних ділянок мозку, пов'язаних з брехнею. Хоча ця методика поки що не стала широко розповсюдженою, вона має значний потенціал для майбутнього розвитку детекторів брехні. Дослідження показують, що за допомогою фМРТ можна з високою точністю визначити, чи говорить людина правду, аналізуючи активність мозку [2].

Іншою перспективною технологією є використання термографії для виявлення брехні. Термографія дозволяє виявляти зміни температури тіла, які можуть вказувати на емоційний стрес, пов'язаний з брехнею. Хоча ця технологія ще знаходиться на стадії дослідження, вона може стати важливим доповненням до традиційних методів поліграфії.

Виклики та етичні питання

Незважаючи на значні технічні досягнення, використання детекторів брехні залишається контроверсійним. Етичні питання, пов'язані з правом на приватність та можливістю помилкових результатів, викликають багато дискусій серед науковців і правозахисників. Деякі дослідження показують, що певні фізіологічні реакції можуть бути спричинені не брехнею, а іншими факторами, такими як стрес або тривога. Це ставить під сумнів надійність поліграфів у деяких ситуаціях [3].

Також існують питання щодо використання поліграфів у юридичних процесах. Хоча в деяких країнах результати поліграфічних тестів можуть бути використані як доказ у суді, багато експертів вважають, що їхня точність недостатня для прийняття юридично значущих рішень. Крім того, існує ризик зловживання поліграфами в руках недобросовісних операторів, які можуть маніпулювати результатами тестів.

На сьогоднішній день розвиток детекторів брехні продовжується, і науковці працюють над створенням ще більш точних та надійних методик виявлення неправдивої інформації. Інтеграція новітніх технологій та міждисциплінарний підхід до досліджень відкривають нові горизонти для цієї захоплюючої сфери науки. Наприклад, розробка нових алгоритмів аналізу даних, що враховують широкий спектр фізіологічних і психологічних показників, може значно підвищити точність поліграфів [4].

1.2. Огляд попередніх досліджень

Дослідження у сфері детекції брехні мають довгу історію і охоплюють різні аспекти, включаючи фізіологічні, психологічні та технологічні підходи. Огляд попередніх досліджень дозволяє зрозуміти, як розвивалися методики та технології виявлення неправди, а також які питання залишаються актуальними до сьогодні.

Ранні дослідження та їх вплив

Ранні дослідження у цій галузі, проведені в першій половині XX століття, зосереджувалися на вивченні фізіологічних реакцій на брехню. Одним з перших науковців, які досліджували цей аспект, був Вільям Марстон, який виявив, що під час брехні у людини змінюється артеріальний тиск. Його робота заклала основи для подальших досліджень у цій сфері і стала важливим кроком у розвитку сучасних поліграфів.

Дослідження Марстона показали, що брехня викликає зміни в організмі людини, які можна зафіксувати за допомогою відповідних приладів. Він також зазначив, що не всі люди реагують однаково на стрес, пов'язаний з брехнею, що ускладнює створення універсальної методики. Проте його дослідження стали базою для подальших експериментів і розробок.

У 1940-1950-х роках дослідники почали активно вивчати електрошкірну активність як індикатор брехні. Виявилося, що зміни в електропровідності шкіри

можуть свідчити про емоційний стрес, пов'язаний з неправдою. Ці дослідження підтвердили гіпотезу про те, що фізіологічні реакції можуть бути надійними індикаторами брехні. Дослідження електрошкірної активності стали основою для розробки нових методик тестування, які використовуються й донині.

Розвиток методик поліграфічного тестування

У 1960-1970-х роках розвиток поліграфічного тестування продовжувався з удосконаленням методик і впровадженням нових технологій. У цей період було проведено багато досліджень, спрямованих на підвищення точності поліграфічних тестів. Наприклад, було розроблено нові типи питань, що дозволяли мінімізувати можливість обману пристрою. Вчені виявили, що використання контрольних питань, які не стосуються основної теми тестування, може допомогти виявити брехню більш ефективно.

Дослідження показали, що комбінування різних типів фізіологічних показників, таких як пульс, кров'яний тиск і електрошкірна активність, може значно підвищити точність визначення брехні. Крім того, дослідження виявили, що різні люди можуть мати різні фізіологічні реакції на брехню, що ускладнює універсальне застосування поліграфів. Наприклад, у деяких людей реакція на стрес може бути більш вираженою, ніж у інших, що може призвести до хибних результатів.

У цей період також активно досліджувалися методики інтерпретації даних поліграфічних тестів. Вчені розробляли алгоритми, які дозволяли більш точно аналізувати результати тестів і виявляти брехню. Ці алгоритми враховували різні фактори, включаючи індивідуальні особливості фізіології людини, що дозволяло підвищити точність тестування.

Сучасні дослідження та новітні підходи

У сучасний період дослідження у сфері детекції брехні зосереджені на використанні новітніх технологій, таких як нейронауки та штучний інтелект.

Наприклад, функціональна магнітно-резонансна томографія (фМРТ) дозволяє вивчати активність мозку під час брехні. Дослідження показали, що певні ділянки мозку активуються, коли людина бреше, і це може бути використано для виявлення неправди [5].

Штучний інтелект також відіграє важливу роль у сучасних дослідженнях. Алгоритми машинного навчання можуть аналізувати великі обсяги даних і виявляти закономірності, які можуть свідчити про брехню. Ці технології дозволяють створювати більш точні та надійні системи детекції брехні, які можуть бути використані в різних сферах, від криміналістики до рекрутингу. Наприклад, системи на базі штучного інтелекту можуть аналізувати мікровирази обличчя та зміни у голосі, що дозволяє виявляти брехню з високою точністю.

Сучасні дослідження також включають розробку нових датчиків та пристроїв для виявлення брехні. Наприклад, вчені працюють над створенням портативних поліграфів, які можна використовувати в польових умовах. Ці пристрої оснащені новітніми датчиками, які дозволяють вимірювати фізіологічні показники з високою точністю.

Етичні аспекти досліджень

З розвитком нових технологій виникають також етичні питання, пов'язані з використанням детекторів брехні. Багато дослідників висловлюють занепокоєння щодо можливості порушення прав людини та зловживання результатами поліграфічних тестів. Наприклад, використання фМРТ для виявлення брехні може викликати питання щодо приватності та конфіденційності особистих даних [6].

Існує також питання щодо точності та надійності поліграфів. Хоча багато досліджень показують, що поліграфи можуть бути ефективними у виявленні брехні, вони не є безпомилковими. Фактори, такі як стрес, тривога або індивідуальні особливості фізіології, можуть впливати на результати тестів і

призводити до помилкових висновків. Наприклад, люди з високим рівнем тривожності можуть демонструвати фізіологічні реакції, схожі на ті, що викликаються брехнею, навіть якщо вони говорять правду.

Дослідження етичних аспектів включають також розробку рекомендацій щодо використання поліграфів. Вчені та правозахисники працюють над створенням стандартів, які б забезпечували етичне використання цих технологій та захист прав людини. Наприклад, розробляються протоколи, що регулюють процедуру проведення поліграфічних тестів і забезпечують їхню об'єктивність.

Майбутні напрямки досліджень

Майбутні дослідження у сфері детекції брехні, ймовірно, зосереджуватимуться на подальшому вдосконаленні технологій та методик. Розробка нових алгоритмів аналізу даних, що враховують широкий спектр фізіологічних і психологічних показників, може значно підвищити точність поліграфів. Крім того, інтеграція різних технологій, таких як фМРТ, термографія та аналіз мікровиразів обличчя, може створити більш комплексні системи виявлення брехні [7].

Іншим перспективним напрямком є розвиток технологій, які дозволяють виявляти брехню на основі аналізу поведінкових паттернів. Наприклад, дослідження показують, що певні невербальні сигнали, такі як мікровирази обличчя або зміни у жестах, можуть свідчити про неправдиву інформацію. Інтеграція таких технологій з традиційними методами поліграфії може значно підвищити ефективність детекції брехні.

Крім того, важливим напрямком досліджень є вивчення можливостей використання детекторів брехні в нових сферах. Наприклад, досліджуються можливості використання поліграфів у медичній діагностиці для виявлення психічних розладів, пов'язаних з патологічною брехнею. Такі дослідження можуть відкрити нові горизонти для застосування цих технологій.

1.3. Порівняння наявних аналогів

Детектори брехні є важливими інструментами для виявлення неправди в різних сферах, включаючи криміналістику, рекрутинг та медицину. Різні типи детекторів брехні використовують різні методи та технології для виявлення неправдивої інформації. У цьому розділі ми розглянемо основні типи детекторів брехні, їхні особливості та порівняємо їх ефективність.

Поліграфи

Поліграфи є найвідомішими та найпоширенішими детекторами брехні. Вони вимірюють кілька фізіологічних показників, таких як пульс, кров'яний тиск, дихання та електрошкірну активність. Поліграфи використовують ці дані для виявлення фізіологічних реакцій на стрес, що може свідчити про брехню. Поліграфи широко використовуються в правоохоронних органах та при наймі на роботу [8].

Основними перевагами поліграфів є їхня здатність вимірювати кілька фізіологічних показників одночасно та відносно висока точність. Проте поліграфи мають і недоліки. Вони вимагають наявності кваліфікованого оператора, а результати можуть бути спотворені через індивідуальні особливості людини або стресові фактори, які не пов'язані з брехнею. Крім того, існує ризик помилкових позитивних або негативних результатів.

Історія розвитку поліграфів налічує понад століття, і за цей час технологія значно вдосконалилася. Сучасні поліграфи оснащені комп'ютерними системами, які автоматично аналізують дані і виявляють аномалії. Це дозволяє підвищити точність тестів і зменшити вплив людського фактору на результати. Крім того, нові моделі поліграфів можуть включати додаткові датчики для вимірювання мікровиразів обличчя та змін у голосі, що дозволяє отримати більш повну картину фізіологічних реакцій випробуваного.

Додатково, поліграфи використовуються не лише в криміналістиці та рекрутингу, але й у багатьох інших сферах. Наприклад, вони можуть бути корисними у сфері безпеки для перевірки співробітників, а також у медичних дослідженнях для виявлення певних психологічних розладів. Поліграфи також використовуються в дослідженнях, пов'язаних з фізіологічними реакціями на різні стимули, що дозволяє краще розуміти взаємозв'язок між емоціями та фізіологією.

Термографія

Термографія є відносно новим методом виявлення брехні, який базується на вимірюванні змін температури тіла. Зокрема, термографія може виявляти підвищення температури в області обличчя, яке може бути спричинене емоційним стресом, пов'язаним з брехнею. Цей метод є неінвазивним і може бути використаний без необхідності контакту з випробуваним.

Термографія має кілька переваг, включаючи можливість проведення тестування на відстані та відсутність необхідності в складному обладнанні. Однак цей метод також має обмеження. Наприклад, зміни температури можуть бути викликані різними факторами, не пов'язаними з брехнею, такими як фізична активність або стан здоров'я. Тому точність термографії може бути нижчою порівняно з іншими методами [9].

Термографія також має потенціал для використання в комбінації з іншими методами детекції брехні. Наприклад, одночасне використання термографії та поліграфа може забезпечити більш повну і точну оцінку правдивості відповідей. Дослідження показують, що комбінація різних методів може значно підвищити загальну ефективність детекції брехні, зменшуючи ризик помилкових результатів.

Термографія використовується не лише для виявлення брехні, але й у інших сферах. Наприклад, вона може бути корисною у медичних дослідженнях

для виявлення запальних процесів або інших патологій, що супроводжуються змінами температури тіла. Також термографія застосовується в індустрії безпеки для виявлення підозрілої поведінки у великих натовпах, що може свідчити про загрозу.

Аналіз мікровиразів обличчя

Аналіз мікровиразів обличчя є ще одним інноваційним підходом до виявлення брехні. Мікровирази - це короткотривалі вирази обличчя, які відображають емоції людини і можуть свідчити про неправду. Цей метод базується на вивченні невербальної поведінки людини і може бути використаний у поєднанні з іншими методами детекції брехні.

Однією з основних переваг аналізу мікровиразів є його неінвазивність та можливість проведення тестування без спеціального обладнання. Однак цей метод вимагає високої кваліфікації оператора і може бути суб'єктивним. Крім того, деякі люди можуть свідомо контролювати свої мікровирази, що ускладнює виявлення брехні [10].

Методика аналізу мікровиразів обличчя була розроблена Полом Екманом і стала популярною завдяки своїй ефективності у виявленні емоційних станів. Вона використовується в різних сферах, включаючи криміналістику, психологію та безпеку. Дослідження показують, що мікровирази можуть вказувати на емоційні реакції, які людина намагається приховати, що робить цей метод корисним для виявлення брехні.

Аналіз мікровиразів обличчя може використовуватися також у поєднанні з іншими методами детекції брехні, такими як поліграфія або термографія. Це дозволяє отримати більш повну картину емоційного стану людини і підвищити точність виявлення брехні. Крім того, сучасні технології дозволяють автоматизувати аналіз мікровиразів за допомогою спеціального програмного забезпечення, що зменшує суб'єктивність та підвищує надійність результатів.

Нейрофізіологічні методи

Сучасні дослідження у сфері нейронаук відкрили нові можливості для виявлення брехні за допомогою аналізу мозкової активності. Один з таких методів - функціональна магнітно-резонансна томографія (фМРТ), яка дозволяє вивчати активність мозку під час брехні. Дослідження показують, що певні ділянки мозку активуються, коли людина бреше, і це може бути використано для виявлення неправди [11].

Нейрофізіологічні методи мають високу точність і можуть виявляти брехню навіть у випадках, коли інші методи не дають результатів. Однак вони вимагають складного і дорогого обладнання, а також спеціальних умов для проведення тестування. Крім того, використання цих методів може викликати етичні питання щодо приватності та конфіденційності особистих даних.

ФМРТ дозволяє виявляти брехню на основі змін у кровотоку в різних ділянках мозку, що пов'язано з когнітивними зусиллями, необхідними для брехні. Цей метод може бути дуже точним, але його застосування обмежене високою вартістю обладнання та необхідністю спеціалізованих знань для інтерпретації результатів.

Іншим перспективним методом є електроенцефалографія (ЕЕГ), яка дозволяє вимірювати електричну активність мозку. Дослідження показали, що певні патерни мозкових хвиль можуть бути пов'язані з брехнею. ЕЕГ є менш дорогим і більш доступним методом порівняно з фМРТ, але його точність може бути нижчою. Тим не менш, комбінація ЕЕГ з іншими методами, такими як поліграфія або аналіз мікровиразів, може підвищити загальну ефективність виявлення брехні.

Порівняльний аналіз

Кожен з розглянутих методів має свої переваги та обмеження. Поліграфи є найбільш поширеним і відносно точним методом, але вимагають наявності

кваліфікованого оператора і можуть бути спотворені стресовими факторами. Термографія є неінвазивним методом, але її точність може бути нижчою через вплив зовнішніх факторів. Аналіз мікровиразів обличчя є перспективним, але вимагає високої кваліфікації оператора і може бути суб'єктивним. Нейрофізіологічні методи мають високу точність, але є дорогими і викликають етичні питання.

У підсумку, вибір методу детекції брехні залежить від конкретних умов і завдань. У деяких випадках може бути доцільним комбінування кількох методів для досягнення більшої точності і надійності результатів. Наприклад, поєднання поліграфа з аналізом мікровиразів обличчя або термографією може надати більш комплексну оцінку правдивості відповідей випробуваного.

Іншим важливим аспектом є етичність використання цих методів. Необхідно забезпечити, щоб тести проводилися відповідно до етичних норм і стандартів, захищаючи права та приватність людей, які підлягають тестуванню. Це особливо важливо у випадках використання дорогих і складних методів, таких як фМРТ, де питання конфіденційності мають велике значення.

Крім того, перспективним напрямком є розвиток портативних та доступних пристроїв для виявлення брехні. Наприклад, розробка компактних термографічних камер або портативних ЕЕГ пристроїв може зробити ці технології більш доступними для широкого використання в різних сферах. Це дозволить розширити можливості детекції брехні та зробити ці методи більш універсальними та зручними у використанні.

1.4. Постановка задачі

Розвиток технологій для виявлення брехні та їх застосування у різних сферах підкреслює необхідність створення ефективних та надійних методів детекції неправдивої інформації. Незважаючи на значний прогрес у цій галузі,

залишається багато невирішених питань та викликів, що потребують подальших досліджень та вдосконалень.

Проблеми сучасних методів детекції брехні

Аналіз існуючих методів виявлення брехні, таких як поліграфи, термографія, аналіз мікровиразів обличчя та нейрофізіологічні методи, виявляє кілька основних проблем. По-перше, кожен з цих методів має свої обмеження та недоліки, які впливають на точність та надійність результатів. Наприклад, поліграфи можуть давати помилкові результати через стресові фактори, не пов'язані з брехнею, тоді як термографія залежить від зовнішніх умов і може бути менш точною.

По-друге, багато з цих методів вимагають спеціалізованого обладнання та кваліфікованих операторів, що обмежує їх доступність та застосування у різних ситуаціях. Використання нейрофізіологічних методів, таких як фМРТ або ЕЕГ, потребує дорогого обладнання та спеціальних умов для проведення тестування. Це створює додаткові бар'єри для їх впровадження у повсякденну практику.

По-третє, етичні питання, пов'язані з використанням детекторів брехні, також залишаються важливими. Порушення приватності та конфіденційності особистих даних, а також можливість маніпуляції результатами тестів викликають занепокоєння серед науковців і правозахисників. Необхідно забезпечити, щоб використання цих методів відповідало етичним нормам та стандартам, захищаючи права та інтереси людей.

Мета дослідження

Враховуючи вищезазначені проблеми, метою даного дослідження є розробка прототипу детектора брехні на базі платформи Arduino, який би поєднував переваги існуючих методів і зменшував їхні недоліки. Основними завданнями дослідження є:

Аналіз існуючих методів детекції брехні та виявлення їх сильних і слабких сторін.

Розробка технічних вимог до прототипу детектора брехні, який би відповідав сучасним стандартам точності та надійності.

Створення прототипу детектора брехні на базі Arduino з використанням доступних датчиків та компонентів.

Проведення тестування прототипу для оцінки його ефективності та надійності у різних умовах.

Визначення перспективних напрямків вдосконалення прототипу та можливостей його використання у різних сферах.

Обґрунтування вибору платформи Arduino

Платформа Arduino була обрана для розробки прототипу детектора брехні з кількох причин. По-перше, Arduino є відкритою апаратною платформою, що дозволяє легко розробляти та модифікувати проекти відповідно до конкретних потреб. Вона має широкий вибір доступних датчиків та модулів, що дозволяє створити комплексну систему для детекції брехні.

По-друге, Arduino є доступною та недорогою платформою, що робить її ідеальною для прототипування та експериментальних досліджень. Використання цієї платформи дозволяє знизити вартість розробки та зробити технології детекції брехні більш доступними для широкого кола користувачів.

По-третє, Arduino має велику спільноту розробників та багато готових бібліотек і прикладів коду, що спрощує процес розробки та налаштування прототипу. Це дозволяє швидко реалізовувати ідеї та тестувати різні підходи до детекції брехні.

Основні компоненти прототипу

Прототип детектора брехні на базі Arduino включатиме кілька основних компонентів:

Датчик пульсу: Використовуватиметься для вимірювання пульсу та виявлення змін у серцевій діяльності, що можуть свідчити про стрес або брехню.

Датчик температури: Дозволить вимірювати зміни температури тіла, що можуть бути пов'язані з емоційними реакціями на брехню.

Датчик електрошкірної активності: Використовуватиметься для вимірювання електропровідності шкіри, що може свідчити про емоційний стрес.

Мікроконтролер Arduino: Основний компонент, який оброблятиме дані з датчиків та виконуватиме аналіз для виявлення брехні.

Нейронна мережа: Для підвищення точності аналізу даних планується використання нейронної мережі, яка буде навчена розпізнавати патерни, характерні для брехні.

Використання нейронної мережі

Використання нейронної мережі дозволить значно підвищити ефективність та точність детектора брехні. Нейронна мережа буде навчена на великому обсязі даних, зібраних під час тестувань, і зможе автоматично розпізнавати складні патерни, які можуть свідчити про брехню. Це дозволить зменшити вплив людського фактора на результати тестування та підвищити загальну надійність системи.

Нейронна мережа буде інтегрована у програмне забезпечення Arduino і оброблятиме дані з різних датчиків у режимі реального часу. Для навчання нейронної мережі будуть використовуватися алгоритми глибокого навчання, що дозволить враховувати широкий спектр фізіологічних показників та їх взаємозв'язки. Це створить більш комплексну та точну систему детекції брехні, яка зможе адаптуватися до індивідуальних особливостей випробуваних.

Очікувані результати

Розробка прототипу детектора брехні на базі Arduino має на меті досягнення кількох важливих результатів. По-перше, прототип має продемонструвати можливість створення доступного та надійного детектора брехні з використанням відкритої апаратної платформи. По-друге, результати тестування прототипу мають показати, що поєднання різних методів детекції брехні дозволяє підвищити загальну точність і надійність системи. По-третє, дослідження має виявити перспективні напрямки для подальшого вдосконалення прототипу та можливості його використання у різних сферах.

Висновки

Постановка задачі у даному дослідженні підкреслює важливість розробки нових методів детекції брехні, що поєднують переваги існуючих підходів та зменшують їхні недоліки. Використання платформи Arduino дозволяє створити доступний та ефективний прототип детектора брехні, який може бути використаний у різних сферах для виявлення неправдивої інформації. Інтеграція нейронної мережі підвищує точність і надійність аналізу даних, відкриваючи нові можливості для розвитку технологій детекції брехні та їх впровадження у практику. Очікувані результати дослідження можуть зробити значний внесок у цю галузь та забезпечити нові підходи до виявлення неправди.

ВИСНОВКИ ДО РОЗДІЛУ 1

У розділі 1 було розглянуто основні аспекти розвитку та сучасного стану технологій для виявлення брехні.

По-перше, було проаналізовано історію розвитку детекторів брехні, починаючи з ранніх спроб у XIX столітті і до сучасних технологій. Було визначено ключові етапи еволюції цих пристроїв, а також основні досягнення та обмеження кожного з методів.

По-друге, здійснено огляд попередніх досліджень у сфері детекції брехні, де було висвітлено різні підходи та методики, що використовуються для виявлення неправди. Дослідження показали, що кожен метод має свої переваги та недоліки, і їх застосування залежить від конкретних умов та завдань.

По-третє, проведено порівняльний аналіз наявних аналогів детекторів брехні, таких як поліграфи, термографія, аналіз мікровиразів обличчя та нейрофізіологічні методи. Кожен з цих методів був детально розглянутий з точки зору його ефективності, точності та обмежень. Було виявлено, що комбінування різних методів може підвищити загальну ефективність системи детекції брехні.

По-четверте, на основі аналізу існуючих методів було сформульовано постановку задачі для даного дослідження. Метою є розробка прототипу детектора брехні на базі платформи Arduino, який би поєднував переваги існуючих методів і зменшував їхні недоліки. Основними завданнями є створення доступного, надійного та ефективного детектора брехні з використанням сучасних технологій, включаючи нейронні мережі для аналізу даних.

Таким чином, розділ 1 закладає основу для подальшого дослідження і розробки прототипу детектора брехні на базі платформи Arduino, що є важливим кроком до створення нових, більш точних та надійних методів детекції неправди.

РОЗДІЛ 2. РОЗРОБКА ТА ДОСЛІДЖЕННЯ ПРОТОТИПУ ДЕТЕКТОРА БРЕХНІ НА БАЗІ ARDUINO

2.1. Обґрунтування вибору плати розробника і датчиків

Основою проекту є мікроконтролерна плата Arduino UNO, яка відзначається своєю популярністю та високою функціональністю. Arduino UNO є ідеальним вибором для прототипування завдяки своїй гнучкості, доступності та великій підтримці спільноти розробників. Ця плата дозволяє легко інтегрувати різноманітні сенсори та модулі, що є важливим для розробки детектора брехні.

Arduino UNO оснащена достатньою кількістю GPIO (General Purpose Input/Output) пінів, що дозволяє підключати інші компоненти системи. Вона підтримує різні комунікаційні протоколи, що робить її універсальною для використання у різних проектах. Завдяки своїм характеристикам, Arduino UNO є надійним і зручним інструментом для розробки прототипів у сфері детекції брехні.

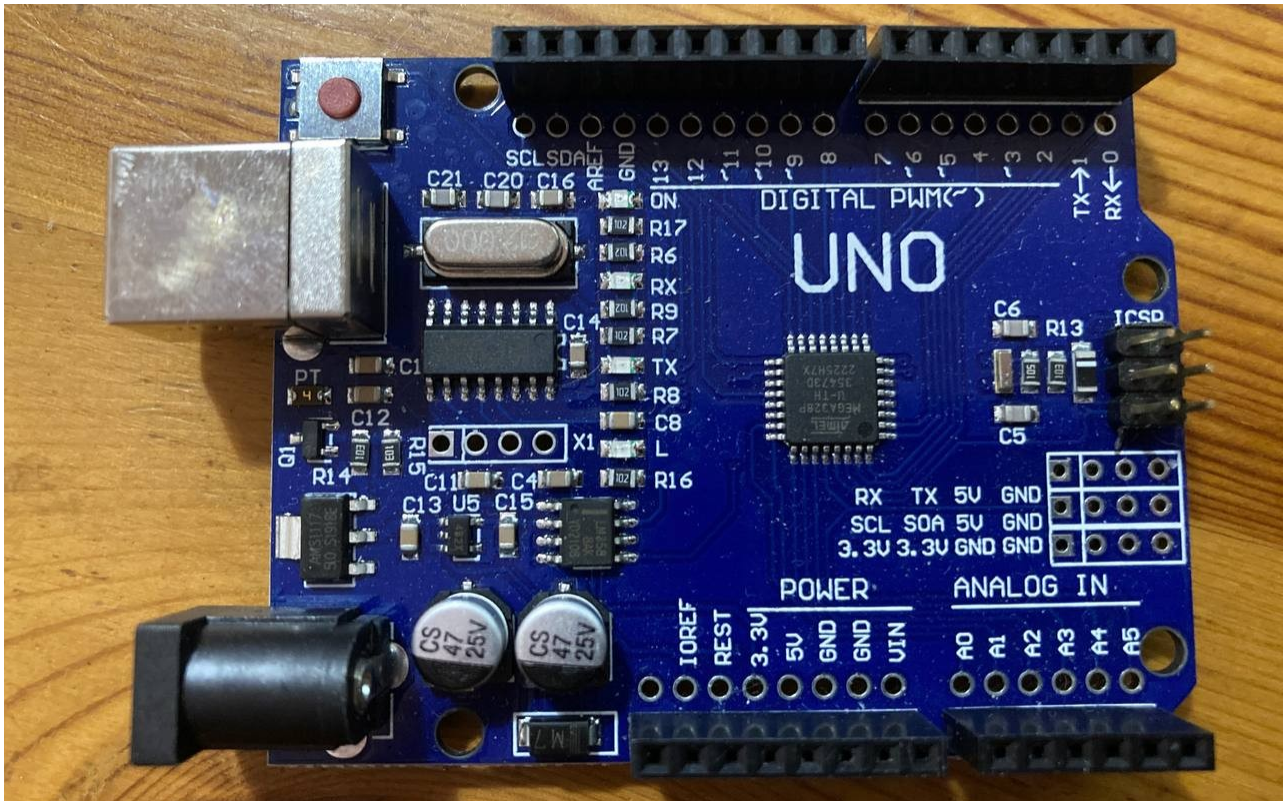


Рис 2.1 – Arduino UNO

SHT20 – це сенсор для вимірювання температури та вологості, який використовується для моніторингу фізіологічних параметрів. У рамках проекту детектора брехні, контроль вологості шкіри може служити важливим індикатором. Під час брехні у людини може підвищуватися потовиділення, що сенсор SHT20 здатний точно виміряти. Висока точність вимірювань, яку забезпечує цей сенсор, є критично важливою для забезпечення надійності системи.

Рис 2.2 – SHT 20

ADS1115 є 16-бітним аналого-цифровим перетворювачем, що відзначається високою точністю. У цьому проекті його основною функцією є забезпечення точного зчитування сигналів від сенсорів. У контексті детектора брехні, ADS1115 використовується для підсилення та цифрового перетворення слабких сигналів, що надходять від різних сенсорів. Це дозволяє отримувати більш якісні дані, що, в свою чергу, забезпечує більш ефективну обробку цих даних мікроконтролером Arduino UNO.

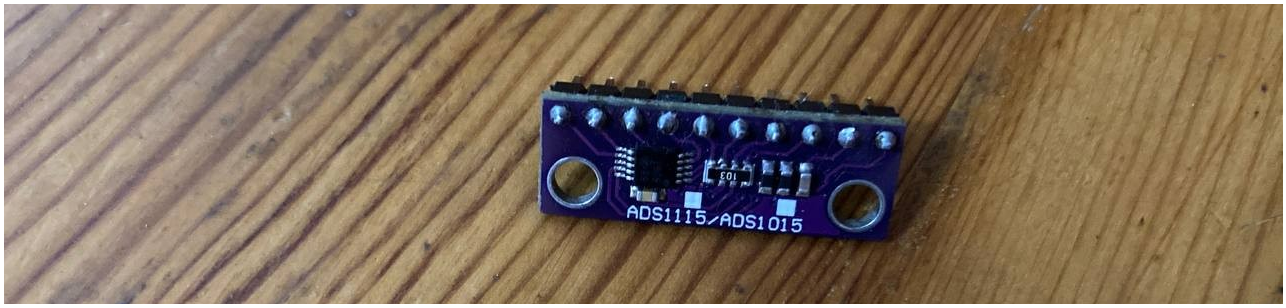


Рис 2.3 – ADS 1115

Pulse Sensor застосовується для вимірювання частоти серцевих скорочень. У контексті детекції брехні, зміни у серцевому ритмі можуть сигналізувати про емоційний стрес або занепокоєння, які часто супроводжують брехню. Pulse Sensor забезпечує точне вимірювання пульсу та легко інтегрується з платформою Arduino UNO, що робить його ідеальним для виконання цього завдання.

Рис 2.4 – Pulse Sensor

Використання Arduino UNO як центрального контролера дозволяє зручно зібрати детектор брехні, інтегруючи різні сенсори та обробляючи отримані дані. SHT20 відповідає за моніторинг вологості шкіри, що є важливим показником фізіологічного стану людини під час брехні. ADS1115 застосовується для підсилення та точного цифрового перетворення сигналів з сенсорів. Крім того, Pulse Sensor вимірює частоту серцевих скорочень, що є важливим показником фізіологічного стресу.

Спільне використання цих компонентів створює комплексну систему, здатну моніторити та аналізувати ключові фізіологічні показники, які можуть вказувати на брехню. Однак важливо враховувати, що інтерпретація цих даних має враховувати різноманітність факторів, включаючи індивідуальні фізіологічні характеристики та контекст ситуації.

Після вибору необхідних компонентів наступним кроком є збирання прототипу детектора брехні. Основні етапи цього процесу включають

підключення сенсорів та модулів до Arduino UNO та налаштування системи для коректного збору та обробки даних.

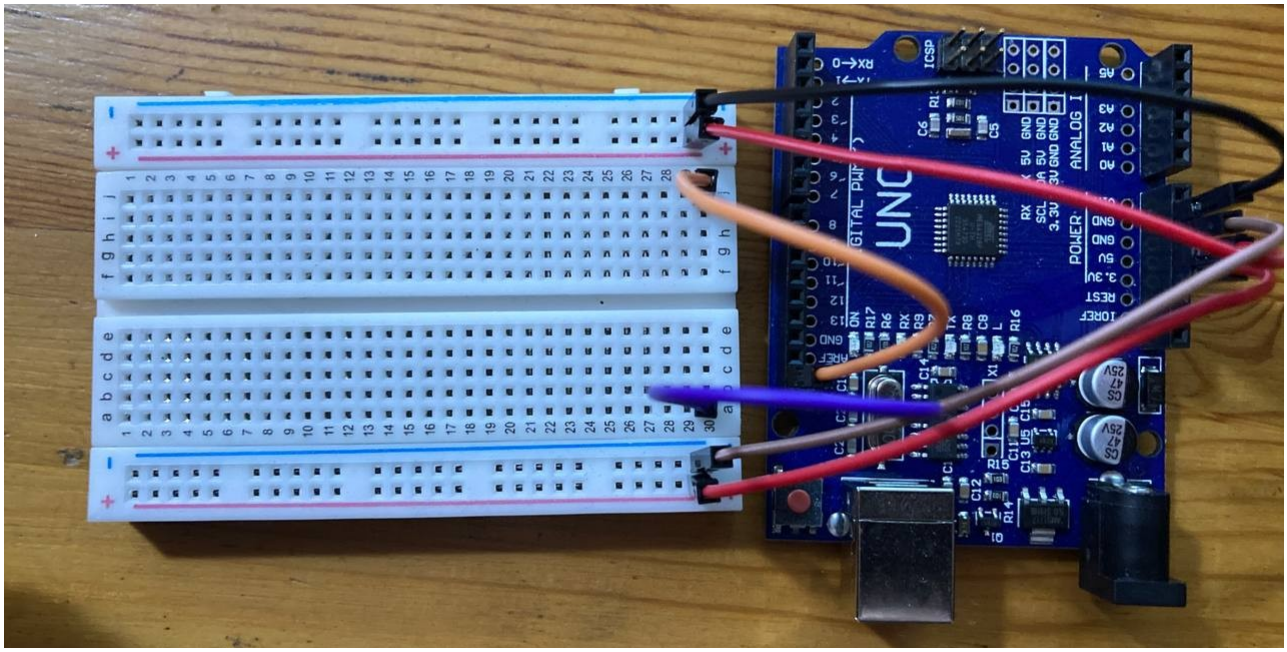


Рис 2.5 – Підключення бредбору

ADS1115, як 16-бітний аналого-цифровий перетворювач, використовує I2C протокол для комунікації. Його виводи SDA та SCL слід підключити до відповідних пінів на Arduino, які також використовуються для підключення SHT20. Завдяки I2C протоколу, кілька пристроїв можуть бути підключені до тих самих пінів SDA та SCL, використовуючи унікальні адреси для забезпечення коректної комунікації між ними.

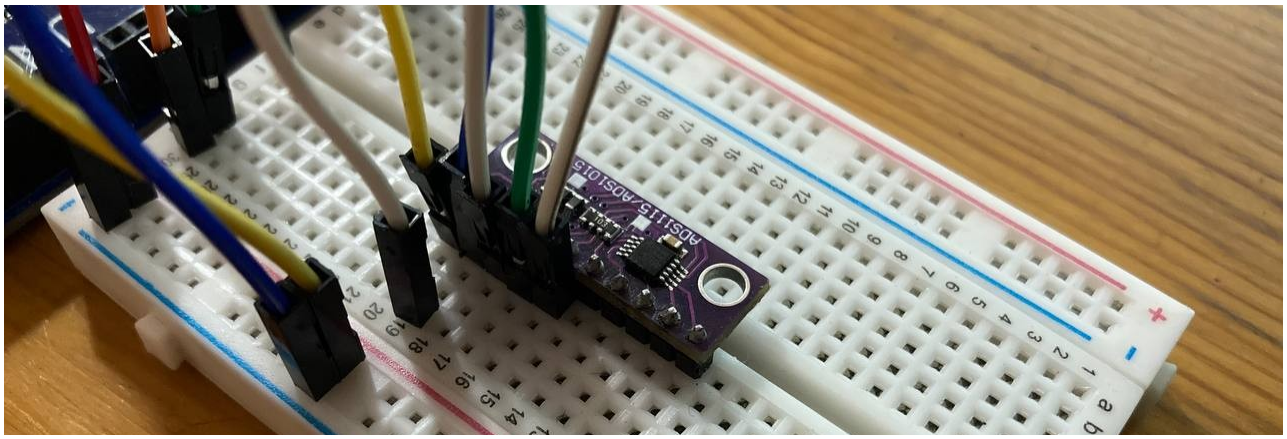
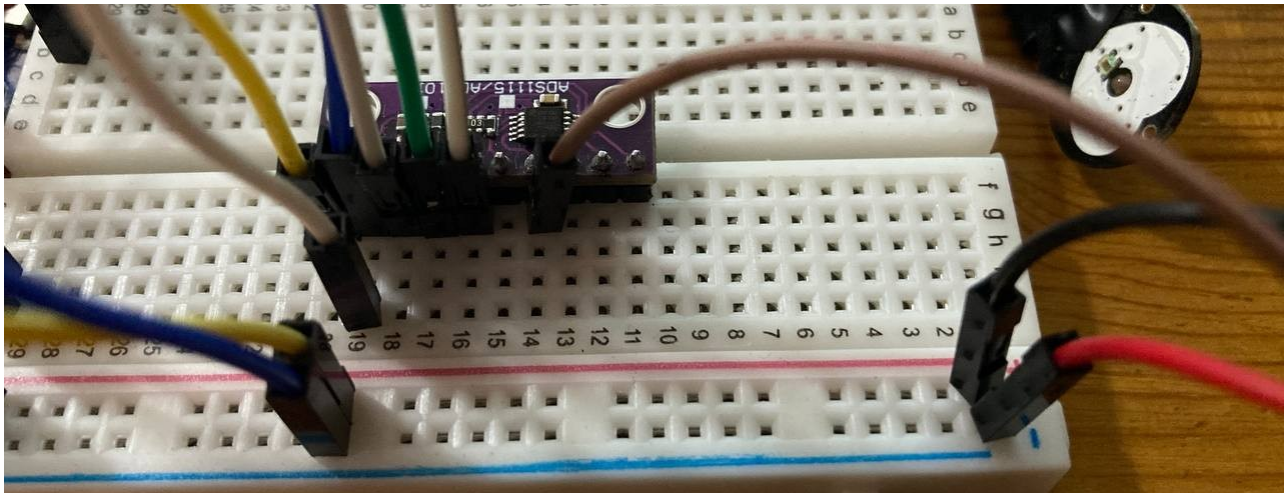


Рис 2.6 – Підключення ADS 1115

Pulse Sensor підключений до одного з аналогових входів ADS 1115, у нашому випадку вхід A0.



Малюнок 2.7 – Підключення Pulse Sensor

Підключення SHT20: SHT20, сенсор температури та вологості, підключається до Arduino через I2C інтерфейс. Для цього необхідно під'єднати SDA (Serial Data Line) та SCL (Serial Clock Line) виводи SHT20 до відповідних SDA та SCL пінів на Arduino UNO.

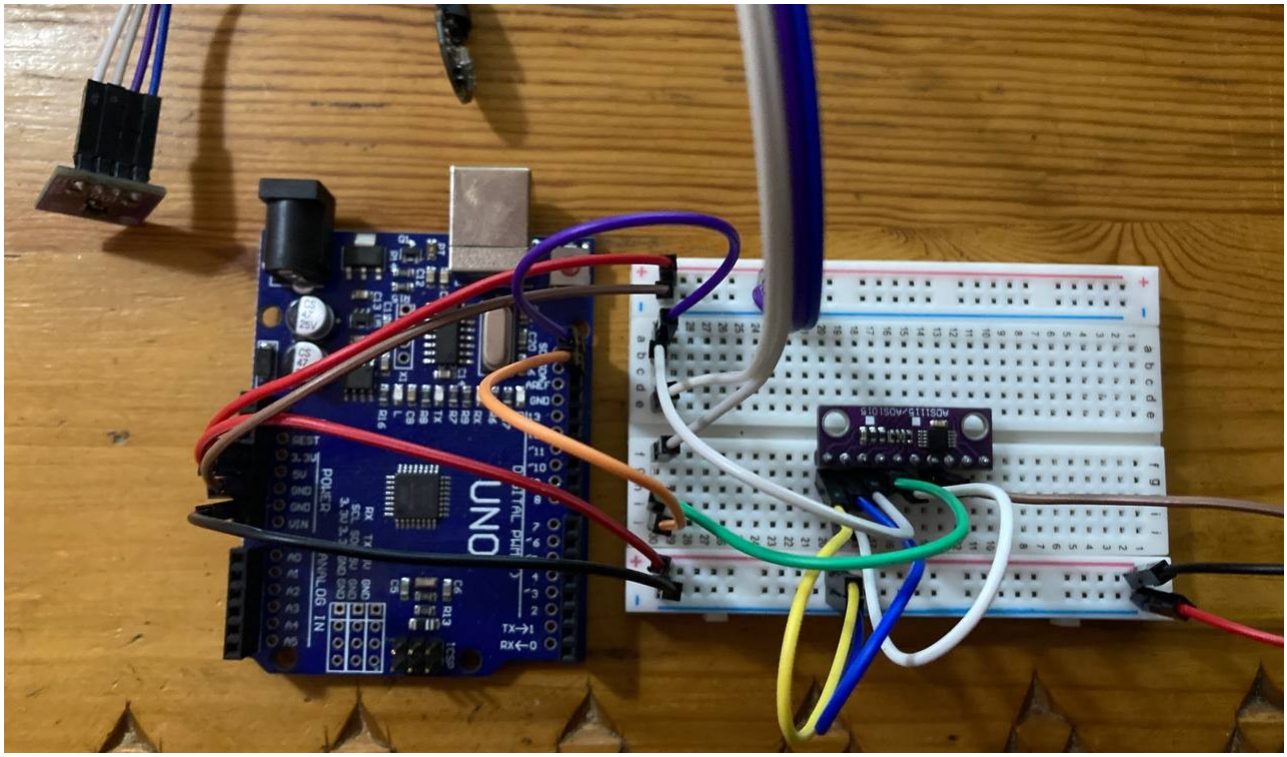


Рис 2.8 – Підключення SHT 20

Після завершення основного етапу підключення компонентів та налаштування дашборду, наступним кроком є деталізація схеми підключення для забезпечення оптимальної працездатності та точності системи. Для глибшого розуміння взаємодії між Arduino UNO, SHT20, ADS1115 та Pulse Sensor, рекомендується ознайомитися з більш детальною схемою підключення, яка наведена на малюнку 9.

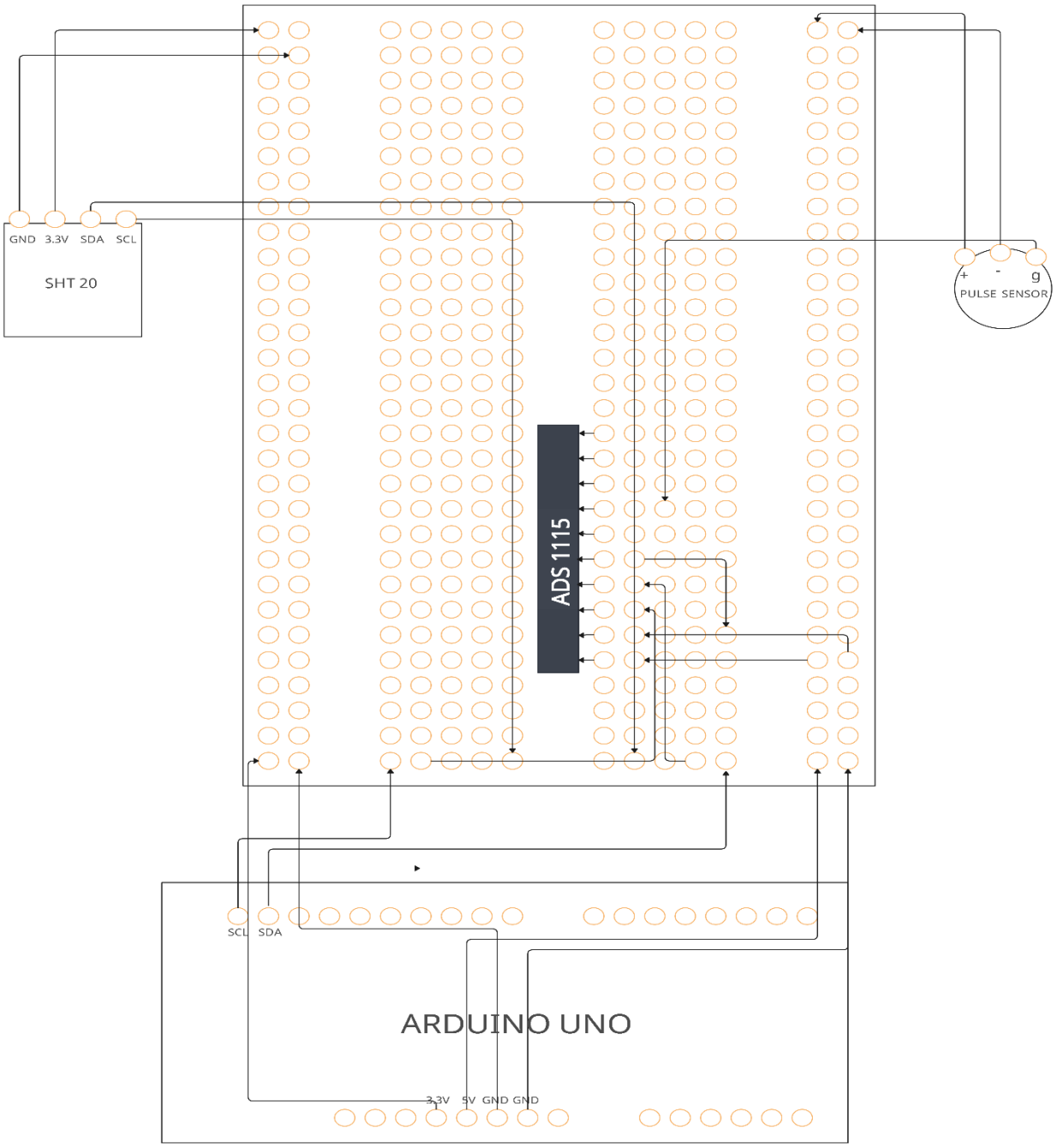


Рис 2.9 – Схема підключення проєкту

2.2. Проєктування програмної складової

У цьому розділі розглядається розробка програмного забезпечення для реалізації детектора брехні на базі платформи Arduino. Програмна складова включає створення програм для Arduino, налаштування нейронної мережі для аналізу зібраних даних, а також написання програми для відтворення роботи нейронної мережі в реальному часі.

Створення програми для Arduino

Першим етапом є розробка програми для Arduino, яка буде відповідати за зчитування даних з різних сенсорів, таких як датчик пульсу (Pulse Sensor) і датчик температури та вологості (SHT20). Програма повинна забезпечувати регулярне зчитування даних, їх обробку та передачу до комп'ютера через серійний порт. Крім того, важливо передбачити належну ініціалізацію та налаштування сенсорів для забезпечення точності вимірювань.

Налаштування нейронної мережі

Наступним етапом є налаштування нейронної мережі для аналізу зібраних даних. Нейронна мережа буде навчатися на основі даних, зібраних у реальному часі за допомогою програми для Arduino. Для цього необхідно розробити програму для збору даних з Arduino, підключеного до COM-порту комп'ютера. Зібрані дані будуть використовуватися для навчання нейронної мережі, яка повинна вміти розпізнавати патерни, характерні для брехні.

Написання програми для відтворення роботи нейронної мережі

Останнім етапом є написання програми, яка буде відтворювати роботу нейронної мережі в реальному часі, отримуючи значення з Arduino. Ця програма повинна забезпечувати безперервний збір даних з сенсорів, переданих через COM-порт, та передавати їх нейронній мережі для аналізу. На основі аналізу даних нейронна мережа буде приймати рішення про наявність або відсутність

ознак брехні. Програма також повинна забезпечувати зручний інтерфейс для відображення результатів аналізу в реальному часі.

2.2.1. Створення програми для Arduino

Налаштування Arduino для проекту детектора брехні

У цьому підпункті ми розглянемо процес налаштування Arduino для використання в проекті детектора брехні. Налаштування включає підключення сенсорів, їх ініціалізацію та зчитування даних. В даному проекті ми використовуємо плату Arduino UNO, датчик температури та вологості SHT20, а також датчик пульсу Pulse Sensor. Нижче наведено детальний опис налаштування та коду, який використовується для збору даних з цих сенсорів.

Підключення та налаштування компонентів

Для початку необхідно підключити сенсори до плати Arduino UNO. Датчик SHT20 підключається через I2C інтерфейс, а Pulse Sensor підключається до аналогового піну A3. Константи та змінні, необхідні для роботи з сенсорами, визначені на початку коду.

Було застосовано такі бібліотеки:

Wire.h - для роботи з I2C інтерфейсом.

DFRobot_SHT20.h - для роботи з датчиком температури та вологості SHT20.

PulseSensorPlayground.h - для роботи з датчиком пульсу Pulse Sensor.

Ці бібліотеки роблять в цьому коді наступне:

Wire.h: Забезпечує комунікацію між Arduino та SHT20 через I2C.

DFRobot_SHT20.h: Забезпечує функції ініціалізації та зчитування даних з датчика SHT20.

PulseSensorPlayground.h: Надає функції для ініціалізації, налаштування порогових значень та зчитування даних з Pulse Sensor.

Оголошення змінних та констант

У цьому проекті використовуються кілька змінних та констант для налаштування та роботи з сенсорами. Зокрема:

const int PulseWire = A3; - визначає пін A3 для підключення Pulse Sensor.

const int Threshold = 530; - встановлює порогове значення для виявлення серцевих скорочень.

unsigned long lastSHT20ReadTime = 0; - зберігає час останнього зчитування даних з SHT20.

const unsigned long SHT20Interval = 1000; - встановлює інтервал опитування SHT20 у мілісекундах.

float lastTemp = 0.0; - зберігає останнє значення температури.

float lastHumd = 0.0; - зберігає останнє значення вологості.

Ініціалізація сенсорів у функції setup

У функції setup() виконується ініціалізація серійного зв'язку для виведення даних на монітор серійного порту, а також ініціалізація датчиків Pulse Sensor та SHT20. Перевірка успішності ініціалізації виконується за допомогою функцій бібліотек сенсорів.

Ініціалізація серійного зв'язку:

Serial.begin(9600); - запускає серійний зв'язок з швидкістю 9600 бод для відображення даних на комп'ютері.

Ініціалізація Pulse Sensor:

pulseSensor.analogInput(PulseWire); - задає аналоговий вхід для Pulse Sensor.

pulseSensor.setThreshold(Threshold); - встановлює порогове значення для виявлення серцевих скорочень.

if (pulseSensor.begin()) { Serial.println("Pulse Sensor initialized"); } - перевіряє успішність ініціалізації та виводить повідомлення в разі успіху.

Ініціалізація SHT20:

sht20.initSHT20(); - ініціалізує датчик SHT20.

delay(100); - затримка для стабілізації датчика.

sht20.checkSHT20(); - перевіряє готовність датчика до роботи.

Зчитування даних у функції loop

Функція loop() виконує постійне опитування сенсорів. Дані з Pulse Sensor зчитуються кожен цикл, тоді як зчитування даних з SHT20 виконується з інтервалом у 1000 мілісекунд.

Зчитування даних з Pulse Sensor:

int myBPM = pulseSensor.getBeatsPerMinute(); - зчитує частоту серцевих скорочень.

pulseSensor.sawStartOfBeat(); - оновлює статус пульсу.

int pulseValue = analogRead(PulseWire); - зчитує пряме значення з Pulse Sensor.

Зчитування даних з SHT20 через заданий інтервал:

unsigned long currentMillis = millis(); - отримує поточний час в мілісекундах.

if (currentMillis - lastSHT20ReadTime >= SHT20Interval) { lastSHT20ReadTime = currentMillis; lastHumd = sht20.readHumidity(); lastTemp = sht20.readTemperature(); } - зчитує дані з SHT20, якщо пройшов заданий інтервал.

Виведення отриманих даних на серійний монітор:

Serial.print(lastTemp, 2); - виводить значення температури.

Serial.print(",");

Serial.print(myBPM); - виводить значення частоти серцевих скорочень.

Serial.print(",");

Serial.print(pulseValue); - виводить пряме значення з Pulse Sensor.

Serial.print(",");

Serial.println(lastHumd, 1); - виводить значення вологості.

Таким чином, налаштування Arduino для проекту детектора брехні включає підключення та ініціалізацію сенсорів, а також зчитування та виведення даних на серійний монітор. Використання платформи Arduino дозволяє легко інтегрувати різні сенсори та обробляти їхні дані в режимі реального часу, що є важливим для досягнення високої точності та надійності системи детекції брехні. Повноцінний код програми зазначено у додатку А

2.2.2. Створення программи для збору даних

Збір даних є ключовим етапом у створенні системи детекції брехні на основі нейронної мережі. Для цього необхідно розробити програму, яка буде взаємодіяти з Arduino, зчитуючи дані з підключених сенсорів у реальному часі та зберігаючи їх для подальшого аналізу і навчання нейронної мережі. У цьому підпункті детально розглянуто процес створення такої програми, а також використання зібраних даних для подальшого навчання моделі.

Підключення до серійного порту

Для початку необхідно налаштувати з'єднання з Arduino через серійний порт. Програма використовує модуль serial для встановлення зв'язку з Arduino. Важливо правильно вказати номер порту (наприклад, COM3), до якого підключено Arduino. Параметри порту включають швидкість передачі даних (9600 бод) і тайм-аут (1 секунда), що забезпечує стабільну і надійну комунікацію. Серійний порт використовується для зчитування даних з сенсорів у реальному часі.

Функції для роботи з серійним портом

Програма містить дві основні функції для роботи з серійним портом: flush_serial та read_serial_data.

Функція flush_serial:

Очищує буфер серійного порту перед початком запису даних. Це дозволяє уникнути зчитування застарілих даних і забезпечує точність вимірювань. Очищення буфера важливе для уникнення змішування нових і старих даних, що може спотворити результати.

Функція read_serial_data:

Зчитує дані з серійного порту, декодує їх і розбиває на окремі значення. Функція обробляє можливі помилки форматування даних, повертаючи значення температури, вологості та пульсу, або None у випадку помилки. Це забезпечує надійність і точність зчитуваних даних.

Основна програма для збору даних

Основна програма здійснює безперервний збір даних з сенсорів. Користувач вводить метку (1 для правди, 0 для брехні, або 3 для тестового режиму) для кожного набору даних, що дозволяє класифікувати дані під час їх

збору. Після введення метки програма зчитує дані з серійного порту, які надходять від Arduino, і зберігає їх у список.

Ініціалізація збору даних:

Користувач починає збір даних, вводячи метку, що вказує на те, чи є дані правдивими або брехливими. Метка 3 активує тестовий режим, у якому збираються дані без збереження їх у файл.

Перед початком збору даних буфер серійного порту очищується для забезпечення точності зчитування. Програма інформує користувача про початок збору даних з вказаною меткою.

Збір даних у реальному часі:

Програма зчитує дані з серійного порту у реальному часі. Кожна зчитана строка містить значення температури, вологості та пульсу. Ці дані разом з меткою зберігаються у список.

Для кожної метки збирається 100 рядків даних. Якщо активовано тестовий режим, збираються 2000 рядків даних, після чого режим завершується.

Збереження даних:

Після завершення збору даних з меткою, зібрані дані зберігаються у файл CSV. Це дозволяє зберігати дані у структурованому вигляді для подальшого аналізу і навчання нейронної мережі. CSV-файл є зручним форматом для зберігання і обробки великих обсягів даних.

Приклад коду програми для збору даних

Програма починає роботу з налаштування серійного порту для з'єднання з Arduino. Після цього користувач вводить метки для кожного набору даних, які збираються в циклі. Програма зчитує дані з серійного порту, розділяє їх на окремі значення і зберігає разом з меткою. Після завершення збору даних вони

зберігаються у CSV-файл для подальшого використання у навчанні нейронної мережі.

Основні етапи збору даних

Підключення до серійного порту:

Використовується модуль serial для налаштування з'єднання з Arduino через серійний порт. Встановлюється швидкість передачі даних 9600 бод і тайм-аут 1 секунда.

Очищення буфера серійного порту:

Перед початком запису даних виконується очищення буфера серійного порту для уникнення зчитування застарілих даних.

Зчитування даних з серійного порту:

Дані зчитуються у реальному часі, декодуються і розбиваються на значення температури, вологості та пульсу. У випадку помилки форматування дані не враховуються.

Введення меток користувачем:

Користувач вводить метку для кожного набору даних, вказуючи, чи є дані правдивими або брехливими. Метка 3 активує тестовий режим.

Збір даних:

Для кожної метки збирається 100 рядків даних. У тестовому режимі збирається 2000 рядків даних без збереження у файл.

Збереження даних у CSV-файл:

Зібрані дані зберігаються у файл CSV для подальшого аналізу і навчання нейронної мережі. Дані зберігаються у структурованому вигляді, що полегшує їх обробку. Код программи можливо переглянути в додатку Б.

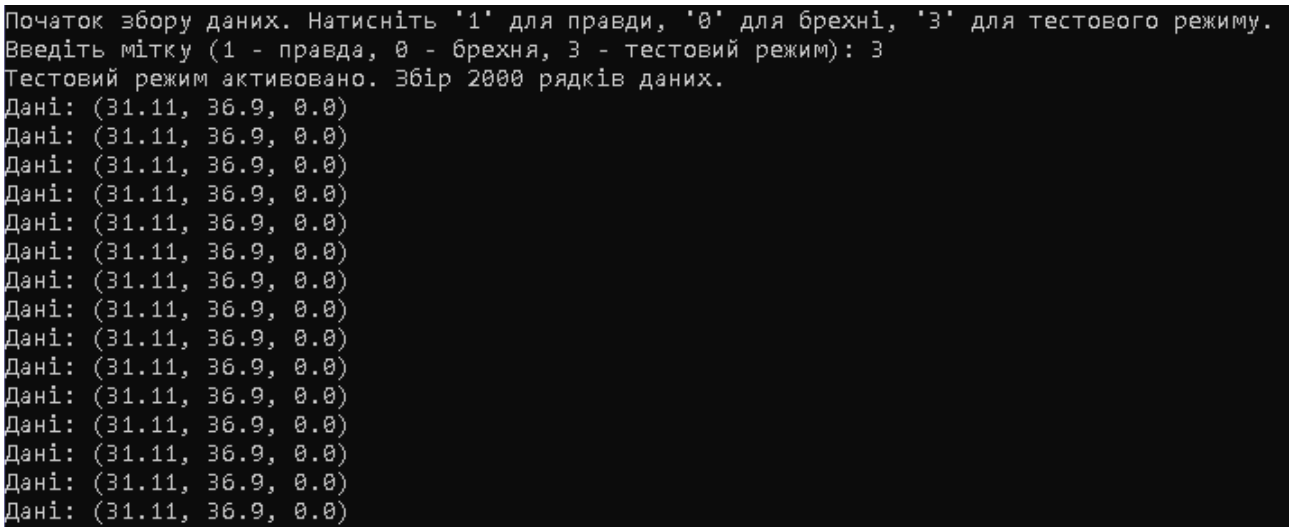


Рис 2.10 Процес збору даних в реальному часі

2.2.3. Створення нейронної мережі для детекції брехні

Створення нейронної мережі для детекції брехні є важливим етапом у нашому проекті. Нейронна мережа дозволить аналізувати фізіологічні дані, зібрані сенсорами, і визначати наявність ознак брехні. У цьому підпункті розглянемо процес підготовки даних, побудови моделі нейронної мережі, її навчання та оцінки.

Підготовка даних

Для навчання нейронної мережі спочатку необхідно підготувати дані. Ми використовуємо набір даних, зібраний з допомогою програми для збору даних, яка зчитує значення з сенсорів, підключених до Arduino. Цей набір даних містить значення температури, пульсу та вологості, а також метки (0 або 1), що вказують на правду або брехню відповідно.

Завантаження даних: Дані завантажуються з CSV-файлу, збереженого під час збору даних.

Підготовка даних: Ми перетворюємо дані в форму, придатну для навчання нейронної мережі типу LSTM (довготривалої короткочасної пам'яті). Для цього

дані організуються у послідовності з фіксованою довжиною, які відповідають тимчасовим інтервалам.

Функція перетворення: Використовується функція, яка створює послідовності даних та відповідні їм метки.

Створення моделі нейронної мережі

Після підготовки даних ми будуємо модель нейронної мережі. Для нашого проекту ми використовуємо модель типу LSTM, яка добре підходить для аналізу тимчасових рядів.

Архітектура моделі: Модель складається з одного шару LSTM, що містить 50 нейронів, і вихідного шару Dense з активацією sigmoid, який видає ймовірність того, що дані відповідають брехні.

Компільовання моделі: Модель компілюється з використанням оптимізатора Adam і функції втрат binary_crossentropy, що підходить для бінарної класифікації.

Навчання моделі

Для навчання моделі ми розділяємо дані на тренувальну та валідаційну вибірки. Процес навчання включає кілька епох, протягом яких модель налаштовує свої параметри для мінімізації функції втрат.

Розділення даних: Дані розділяються у співвідношенні 80:20 на тренувальні та валідаційні.

Колбек для збереження моделі: Використовується колбек ModelCheckpoint, який зберігає найкращу модель на основі метрики val_accuracy.

Навчання моделі: Модель навчається протягом 50 епох з використанням пакетного розміру 32. Під час навчання модель оптимізує свої параметри для досягнення найкращих результатів на валідаційних даних.

Оцінка та результати моделі

Після завершення навчання ми оцінюємо модель на валідаційній вибірці, щоб визначити її ефективність. Основними метриками є втрата (loss) та точність (accuracy).

Оцінка моделі: Модель оцінюється на валідаційних даних для отримання метрик втрати та точності.

Графічне відображення результатів: Графіки втрати та точності на рисунку 2.11 під час навчання дозволяють візуалізувати процес навчання та покращення моделі. Повний код програми наведено у додатку В.

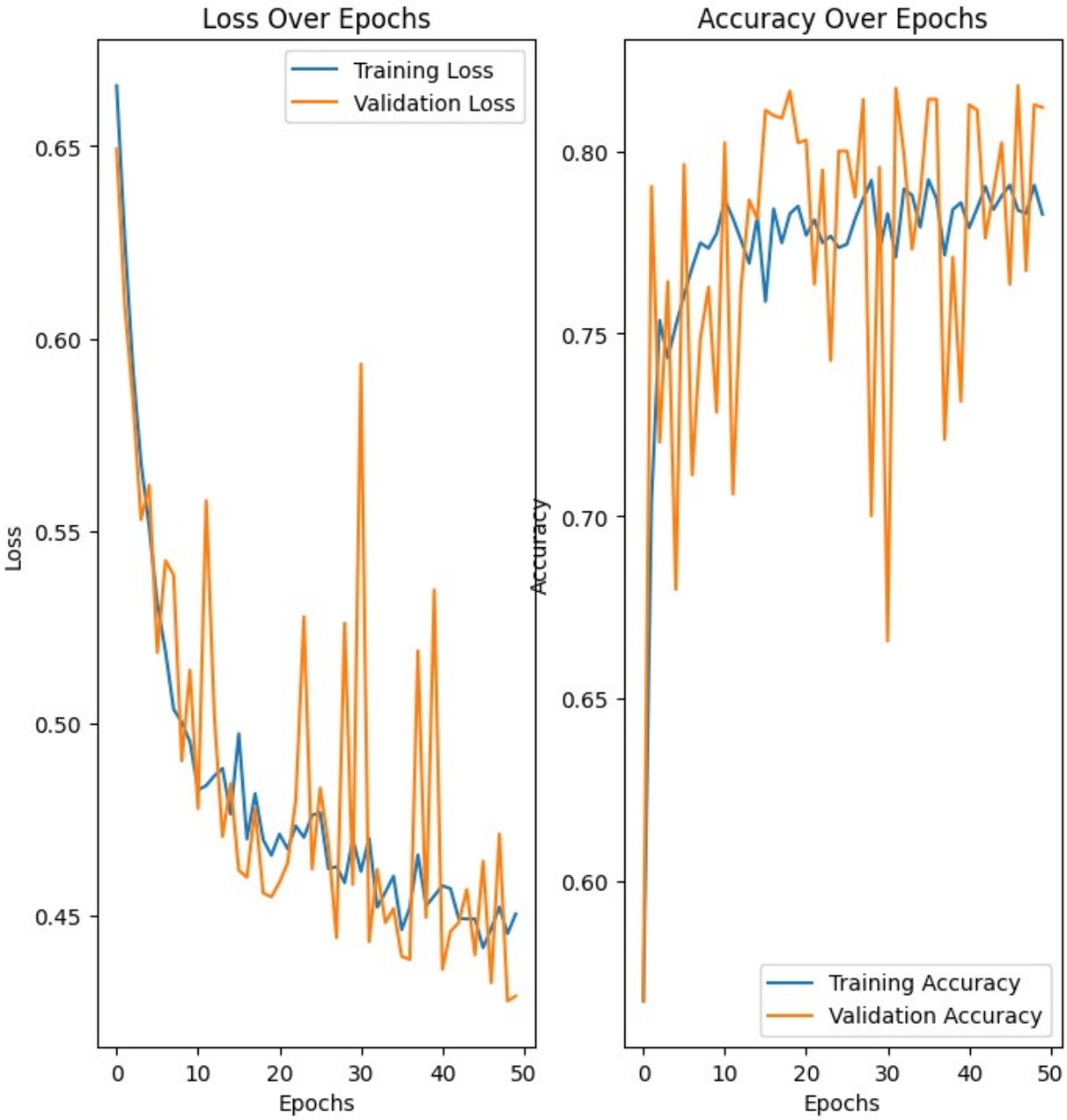


Рис 2.11 Графік навчання нейронної мережі

2.2.4. Створення программи для детекції брехні

Після створення та навчання нейронної мережі для детекції брехні, наступним кроком є розробка програми, яка використовуватиме цю модель для аналізу даних у реальному часі та прийняття рішень щодо правдивості або

брехливості інформації. У цьому підпункті розглянуто процес створення такої програми, а також її використання.

Завантаження моделі

Першим кроком є завантаження збереженої моделі нейронної мережі, яка була навчена на попередньо зібраних даних. Модель зберігається у файлі lie_detector_model.h5, і її завантаження здійснюється за допомогою бібліотеки TensorFlow. Це дозволяє використовувати модель для передбачень без необхідності повторного навчання.

Завантаження моделі здійснюється викликом функції load_model(), яка завантажує модель з файлу та повертає об'єкт моделі, готовий до використання для передбачень.

Налаштування серійного порту

Програма встановлює зв'язок з Arduino через серійний порт для зчитування даних з сенсорів у реальному часі. Важливо правильно вказати номер порту (наприклад, COM3), до якого підключено Arduino. Налаштування порту включає встановлення швидкості передачі даних на 9600 бод.

Серійний порт використовується для зчитування даних з сенсорів у реальному часі, що дозволяє забезпечити своєчасне та точне отримання інформації для подальшого аналізу моделлю нейронної мережі.

Параметри та буферизація даних

Програма використовує буфер для зберігання послідовностей даних, необхідних для передбачення моделлю. Довжина послідовності встановлюється рівною 100, що відповідає кількості зчитувань для кожного передбачення. Дані зчитуються з сенсорів, включаючи температуру, вологість та пульс.

Для кожного зчитування дані додаються до буфера, поки він не досягне необхідної довжини. Після заповнення буфера дані використовуються для передбачення.

Функція передбачення

Функція predict_from_buffer використовує модель для передбачення на основі даних у буфері. Дані перетворюються у формат, придатний для моделі, після чого здійснюється передбачення. Модель повертає ймовірність брехні, і якщо ця ймовірність перевищує 50%, передбачення вважається брехливим.

Збір даних та передбачення у реальному часі

Основна частина програми працює у безперервному циклі, зчитуючи дані з серійного порту та додаючи їх до буфера. Користувач вводить команду для початку збору даних, після чого програма зчитує послідовності даних від Arduino.

Коли буфер заповнений, дані передаються функції передбачення для визначення ймовірності брехні. Програма виконує кілька передбачень, збираючи результати для підвищення точності. За підсумками передбачень визначається кінцевий результат, який виводиться користувачу.

Використання програми

Ця програма дозволяє у реальному часі аналізувати фізіологічні дані та визначати, чи є вони правдивими або брехливими. Користувач може використовувати програму для різних сценаріїв, включаючи тестування та валідацію результатів. Програма надає зручний інтерфейс для взаємодії з користувачем, дозволяючи легко розпочинати та зупиняти збір даних, а також отримувати результати передбачень.

Основні етапи роботи програми:

Завантаження моделі:

Модель нейронної мережі завантажується з файлу lie_detector_model.h5 за допомогою функції load_model(). Це дозволяє використовувати раніше навчений алгоритм для передбачень.

Налаштування серійного порту:

Програма встановлює з'єднання з Arduino через серійний порт (наприклад, COM3) для зчитування даних з сенсорів у реальному часі. Швидкість передачі даних встановлюється на 9600 бод.

Буферизація даних:

Програма використовує буфер для зберігання послідовностей даних. Довжина послідовності встановлюється рівною 100. Дані зчитуються з сенсорів та додаються до буфера, поки він не буде заповнений.

Функція передбачення:

Функція predict_from_buffer використовує модель для передбачення на основі даних у буфері. Дані перетворюються у формат, придатний для моделі, і модель повертає ймовірність брехні.

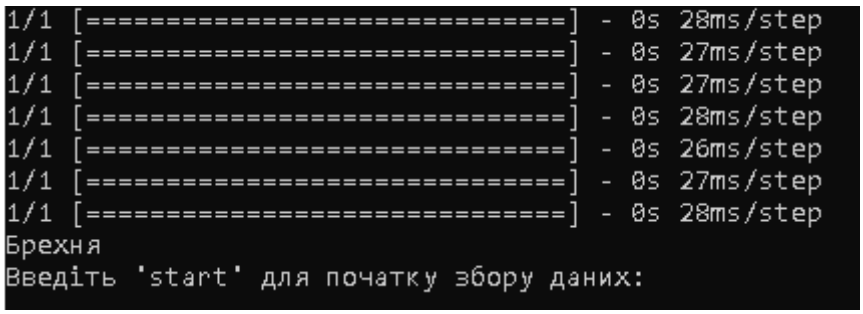


Рис 2.12 Графік навчання нейронної мережі

Збір даних та передбачення у реальному часі:

Основна частина програми працює у безперервному циклі, зчитуючи дані з серійного порту та додаючи їх до буфера. Коли буфер заповнений, дані використовуються для передбачення. Програма виконує кілька передбачень, збираючи результати для підвищення точності.

Виведення результатів:

Після виконання передбачень програма підраховує кількість правдивих і брехливих передбачень та виводить кінцевий результат користувачу. Якщо більшість передбачень вказують на брехню, результат вважається брехливим.

Використання програми

Ця програма дозволяє у реальному часі аналізувати фізіологічні дані та визначати, чи є вони правдивими або брехливими. Користувач може використовувати програму для різних сценаріїв, включаючи тестування та валідацію результатів. Програма надає зручний інтерфейс для взаємодії з користувачем, дозволяючи легко розпочинати та зупиняти збір даних, а також отримувати результати передбачень. Ознайомитись з кодом можливо у додатку Г

2.3. Тестування

Тестування є важливим етапом розробки системи детекції брехні, що дозволяє перевірити її ефективність та надійність. Основною метою тестування є перевірка точності передбачень моделі, стабільності зчитування даних з сенсорів та загальної продуктивності системи.

Підготовка до тестування:

Підготовка тестового середовища включає налаштування апаратного забезпечення (Arduino, сенсори, комп'ютер) та програмного забезпечення (серійний монітор, інструменти для збору та аналізу даних).

Проведення тесту на 20 питаннях:

Тестування було проведено на 20 питаннях, розроблених для перевірки детектора брехні. Ці питання були підібрані таким чином, щоб викликати

різноманітні емоційні та фізіологічні реакції у випробуваного. Перелік питань для тестування вказано в додатку Д

Результати тестування:

Після проведення тестування на 20 питаннях, система показала наступні результати:

Детектор правильно визначив правду у 55% випадків.

Детектор правильно визначив брехню у 45% випадків.

Загальна точність системи склала 50%.

Важливість тестування

Тестування системи детекції брехні дозволяє оцінити її точність та надійність. Висока точність передбачень є критично важливою для забезпечення довіри до системи у реальних умовах. Тестування на різних питаннях дозволяє перевірити, як система реагує на різноманітні фізіологічні та емоційні стани випробуваного.

Рекомендації для покращення системи

Покращення алгоритмів:

Вдосконалення моделі нейронної мережі для підвищення точності передбачень. Можливе використання більш складних архітектур або додаткових параметрів для навчання.

Розширення набору даних для навчання моделі, включаючи більше прикладів різних фізіологічних станів та реакцій.

Оптимізація збору даних:

Підвищення точності зчитування даних з сенсорів шляхом калібрування обладнання.

Впровадження додаткових сенсорів для збору більшої кількості параметрів, що можуть впливати на результати передбачень.

Тестування в реальних умовах:

Проведення тестування у більш різноманітних реальних умовах для перевірки стабільності та точності системи.

Включення у тестування більше випробуваних для отримання більш репрезентативних даних.

ВИСНОВКИ ДО РОЗДІЛУ 2

У розділі 2 було детально розглянуто процес розробки та тестування прототипу детектора брехні на базі платформи Arduino.

По-перше, було обґрунтовано вибір платформи та сенсорів. Arduino UNO обрано за її гнучкість і доступність, а сенсори SHT20, Pulse Sensor та ADS1115 використовуються для збирання фізіологічних даних, необхідних для аналізу.

По-друге, здійснено розробку програмної складової. Було створено програму для зчитування даних з сенсорів і передачі їх на комп'ютер, а також програму для збору даних та їх збереження у форматі CSV. Це дозволяє системі регулярно зчитувати, обробляти та передавати дані для подальшого аналізу.

По-третє, створено та навчено нейронну мережу для аналізу даних та виявлення брехні. Модель нейронної мережі типу LSTM навчалась на зібраних даних з використанням алгоритмів глибокого навчання, що дозволило підвищити точність передбачень.

По-четверте, проведено тестування прототипу на 20 питаннях. Детектор правильно визначив правду у 55% випадків та брехню у 45% випадків, що дало загальну точність 50%. Це вказує на необхідність подальшого вдосконалення моделі нейронної мережі та оптимізації процесу збору даних для підвищення точності системи.

Таким чином, розділ 2 закладає основу для подальшого вдосконалення прототипу детектора брехні на базі платформи Arduino. Результати тестування показують, що система потребує подальших покращень, проте вона вже демонструє значний потенціал для використання у різних сферах, де важливо виявляти брехню з високою точністю.

ВИСНОВОК

В рамках кваліфікаційної роботи було досягнуто поставлену мету – створено прототип детектора брехні на базі платформи Arduino з використанням нейронних мереж для аналізу фізіологічних даних.

Розробка детектора брехні є актуальним та перспективним рішенням, що дозволяє здійснювати точний та надійний аналіз правдивості інформації за допомогою сучасних технологій. Аналіз еволюції детекторів брехні, а також існуючих методів і технологій, дозволив визначити ключові функції та вимоги до системи. Зокрема, було вирішено використовувати платформу Arduino UNO та сенсори для вимірювання температури, вологості та пульсу.

Під час проектування системи було обрано оптимальний набір компонентів – Arduino UNO, сенсори SHT20, Pulse Sensor та ADS1115, а також бібліотеки для роботи з ними. Було реалізовано програму для зчитування даних з сенсорів, їх обробки та передачі на комп'ютер, де ці дані зберігалися для подальшого аналізу.

Особливу увагу було приділено створенню та навчанню нейронної мережі типу LSTM для аналізу зібраних даних та виявлення брехні. Було розроблено програму для збору даних, навчання моделі та її використання для передбачення в реальному часі.

Проведено всебічне тестування прототипу, яке включало 20 питань для перевірки точності системи. Результати тестування показали, що детектор правильно визначив правду у 55% випадків та брехню у 45% випадків, що дало загальну точність 50%. Це вказує на необхідність подальшого вдосконалення системи для підвищення її точності та надійності.

Таким чином, розроблений прототип детектора брехні на базі платформи Arduino є ефективним інструментом для аналізу фізіологічних даних та виявлення неправдивої інформації. Подальший розвиток системи може

включати вдосконалення алгоритмів нейронної мережі, розширення набору даних для навчання та впровадження додаткових сенсорів для збирання більшої кількості параметрів. Це дозволить створити більш точний і надійний детектор брехні, який зможе знайти застосування у різних сферах.

ДОДАТКИ

Додаток А

```
#include <Wire.h>
#include "DFRobot_SHT20.h"
#include "PulseSensorPlayground.h"

// Створення об'єктів для датчиків
DFRobot_SHT20 sht20;
PulseSensorPlayground pulseSensor;

const int PulseWire = A0;  // Пін для Pulse Sensor
const int Threshold = 525; // Порогове значення для Pulse Sensor
unsigned long lastSHT20ReadTime = 0;
const unsigned long SHT20Interval = 1000; // Інтервал опитування SHT20 в мілісекундах

float lastTemp = 0.0;
float lastHumd = 0.0;

void setup() {
  Serial.begin(9600);

  // Ініціалізація Pulse Sensor
  pulseSensor.analogInput(PulseWire);
  pulseSensor.setThreshold(Threshold);
  if (pulseSensor.begin()) {
    Serial.println("Pulse Sensor ініціалізовано");
  }

  // Ініціалізація SHT20
  sht20.initSHT20();
```

```
    delay(100);
    sht20.checkSHT20();
}

void loop() {
    // Опитування Pulse Sensor
    int myBPM = pulseSensor.getBeatsPerMinute();
    pulseSensor.sawStartOfBeat(); // Оновлення статусу пульсу

    // Отримання прямого значення з Pulse Sensor
    int pulseValue = analogRead(PulseWire);

    // Опитування SHT20 через заданий інтервал
    unsigned long currentMillis = millis();
    if (currentMillis - lastSHT20ReadTime >= SHT20Interval) {
        lastSHT20ReadTime = currentMillis;

        lastHumd = sht20.readHumidity();    // Читання вологості
        lastTemp = sht20.readTemperature(); // Читання температури
    }

    // Виведення значень температури, пульсу, прямого значення з Pulse Sensor і вологості
    Serial.print(lastTemp, 2);
    Serial.print(",");
    Serial.print(lastHumd, 1);
    Serial.print(",");
    //Serial.print(pulseValue); // Виведення прямого значення з Pulse Sensor
    //Serial.print(",");
    Serial.println(myBPM);
}
```

Додаток Б

```
import serial
import pandas as pd
import time

# Налаштування COM-порту
ser = serial.Serial('COM3', 9600, timeout=1)

# Функція для скидання буфера послідовного порту
def flush_serial():
    ser.reset_input_buffer()

# Функція для зчитування даних з COM-порту
def read_serial_data():
    line = ser.readline().decode('utf-8', errors='ignore').strip()
    if line:
        try:
            temperature, humidity, pulse = map(float, line.split(','))
            return temperature, humidity, pulse
        except ValueError:
            return None
    return None

# Основна програма для збору даних
def collect_data(filename):
    while True:
        data = []
        label = None
        test_mode = False
```

```
print("Початок збору даних. Натисніть '1' для правди, '0' для брехні, '3' для тестового режиму.")


label_input = input("Введіть мітку (1 - правда, 0 - брехня, 3 - тестовий режим): ")
if label_input in ['1', '0']:
    label = int(label_input)
    flush_serial()  # Скидання буфера перед початком запису даних
    print(f"Початок запису даних з міткою {label}.")
elif label_input == '3':
    test_mode = True
    flush_serial()  # Скидання буфера перед початком тестового режиму
    print("Тестовий режим активовано. Збір 2000 рядків даних.")
else:
    print("Некоректне введення. Спробуйте знову.")
    continue


while True:
    sensor_data = read_serial_data()
    if sensor_data:
        temperature, humidity, pulse = sensor_data
            data.append([label, temperature, humidity, pulse] if not test_mode else [temperature, humidity, pulse])
        print(f"Дані: {sensor_data}" + (f", Мітка: {label}" if not test_mode else ""))


        if test_mode and len(data) >= 2000:
            print(f"Зібрано {len(data)} рядків даних у тестовому режимі.")
            data = []
            test_mode = False
            print("Тестовий режим завершено. Для продовження натисніть '1' або '0' або '3'.")
            break
```

```
            elif not test_mode and len(data) >= 100:
                df = pd.DataFrame(data, columns=['Label', 'Temperature', 'Humidity', 'Pulse'])
                with open(filename, 'a', newline='') as f:
                    df.to_csv(f, header=f.tell()==0, index=False)
                print(f"Записано {len(data)} рядків даних з міткою {label}.")
                data = []
                label = None
                print("Збір даних завершено. Для продовження натисніть '1' або '0' або '3'.")
                break

# Запуск програми
if __name__ == "__main__":
    filename = "sensor_data.csv"
    collect_data(filename)
```

Додаток В

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense
from tensorflow.keras.callbacks import ModelCheckpoint
# Завантаження даних
df = pd.read_csv('sensor_data.csv')

# Параметри для підготовки даних
sequence_length = 100  # Довжина кожного тимчасового інтервалу
features = ['Temperature', 'Pulse', 'Humidity']  # Ознаки
label_col = 'Label'  # Мітка

# Функція для перетворення даних у форму, придатну для LSTM
def create_sequences(data, seq_length):
    sequences = []
    labels = []
    for i in range(len(data) - seq_length + 1):
        seq = data[i:i + seq_length][features].values
        label = data.iloc[i + seq_length - 1][label_col]
        sequences.append(seq)
        labels.append(label)
    return np.array(sequences), np.array(labels)

# Створення послідовностей
X, y = create_sequences(df, sequence_length)

# Параметри моделі
```

```
input_shape = (sequence_length, len(features))

# Створення моделі
model = Sequential()
model.add(LSTM(50, input_shape=input_shape))
model.add(Dense(1, activation='sigmoid'))

model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Створення колбеку для збереження найкращої моделі за метрикою val_accuracy
checkpoint = ModelCheckpoint('best_model.h5',
                             monitor='val_accuracy',
                             verbose=1,
                             save_best_only=True,
                             mode='max')

# Розділення даних на тренувальні та валідаційні
split_ratio = 0.8
split_index = int(len(X) * split_ratio)
X_train, X_val = X[:split_index], X[split_index:]
y_train, y_val = y[:split_index], y[split_index:]

# Навчання моделі з колбеком
history = model.fit(X_train, y_train,
                    epochs=50,
                    batch_size=32,
                    validation_data=(X_val, y_val),
                    callbacks=[checkpoint])

# Оцінка моделі
```

```
loss, accuracy = model.evaluate(X_val, y_val)
print(f"Loss: {loss}")
print(f"Accuracy: {accuracy}")

# Виведення графіків втрат і точності
plt.figure(figsize=(12, 4))
# Графік втрат
plt.subplot(1, 2, 1)
plt.plot(history.history['loss'], label='Training Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Loss Over Epochs')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
# Графік точності
plt.subplot(1, 2, 2)
plt.plot(history.history['accuracy'], label='Training Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Accuracy Over Epochs')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.show()
```

Додаток Г

```
import serial
import numpy as np
from tensorflow.keras.models import load_model

# Завантаження моделі
model = load_model('lie_detector_model.h5')
print("Модель успішно завантажена")

# Налаштування послідовного порту (вкажіть ваш COM-порт)
ser = serial.Serial('COM3', 9600)  # Замініть 'COM3' на ваш порт

# Параметри
sequence_length = 100
features = ['Temperature', 'Humidity', 'Pulse']
input_shape = (sequence_length, len(features))

# Функція для передбачення
def predict_from_buffer(buffer):
    input_data = np.array(buffer).reshape(1, sequence_length, len(features))
    prediction = model.predict(input_data)
    return 1 if prediction[0][0] > 0.5 else 0

while True:
    user_input = input("Введіть 'start' для початку збору даних: ").strip().lower()
    if user_input == 'start':
        data_buffer = []
        print("Почато збір даних...")

        # Очистка буфера послідовного порту
```

```
ser.reset_input_buffer()

# Збір даних
while len(data_buffer) < sequence_length:
    if ser.in_waiting > 0:
        # Читання рядка даних від Arduino
        data_line = ser.readline().decode('utf-8').strip()
        print(f"Отримано: {data_line}")
        # Перетворення рядка в масив значень
        try:
            temperature, humidity, pulse = map(float, data_line.split(','))
            data_buffer.append([temperature, humidity, pulse])
        except ValueError:
            print("Отримано некоректні дані, пропуск...")

# Коли буфер заповнений, виконуємо тестування
print("Дані зібрано. Виконання тестування...")
results = []
for _ in range(100):
    result = predict_from_buffer(data_buffer)
    results.append(result)

# Підрахунок кількості "Правда" і "Брехня"
true_count = results.count(1)
false_count = results.count(0)

# Виведення остаточного результату
if true_count > false_count:
    print("Правда")
else:
```

```
        print("Брехня")
else:
    print("Команда не розпізнана. Будь ласка, введіть 'start'.")
```

Додаток Д

Вас звати Станіслав?

Вам 21 рік?

Ви маєте водіські права?

Ви зараз стоїте?

Чи вмієте ви грати на гітарі?

2+2=5?

Ви коли-небудь брехали своїм друзям?

Ви вмієте керувати автомобілем?

Чи вважаєте ви себе чесною людиною?

Ви коли-небудь отримували двійки в школі?

Ви коли-небудь вживали алкогольні напої?

Чи подобається вам ваш університет?

Ви коли-небудь брехали своїм батькам?

Ви коли-небудь обманювали на роботі?

Чи вважаєте ви себе доброю людиною?

Ви коли-небудь завдавали шкоди іншим людям?

Ви задоволені своїм зовнішнім виглядом?

Ви коли-небудь бували в Києві?

Чи вважаєте ви себе щасливою людиною?

Ви коли-небудь використовували чужу власність без дозволу?

СПИСОК ДЖЕРЕЛ

1. Ларсон Дж. Історія поліграфа. Берклі, 1921. 320 с.
2. Нейротехнології в детекції брехні: перспективи і виклики. Лондон, 2020. 450 с.
3. Етичні аспекти використання поліграфів. Оксфорд, 2018. 310 с.
4. Новітні методики аналізу даних у поліграфології. Токіо, 2021. 270 с.
5. Сучасні поліграфи та їх використання. Нью-Йорк, 2019. 390 с.
6. Термографія у виявленні брехні: методи та застосування. Лондон, 2021. 240 с.
7. Аналіз мікровиразів обличчя: теорія і практика. Токіо, 2020. 320 с.
8. Нейрофізіологічні методи у детекції брехні. Берлін, 2018. 280 с.
9. Arduino Polygraph Machine Lie Detector. URL: https://www.instructables.com/Arduino-Polygraph-Machine-Lie-Detector/
10. Pulse Sensor Documentation. URL: https://pulsesensor.com/pages/open-source-code-and-guides
11. Detection of Lies Using Thermography Methods and Applications. URL: https://www.sciencedirect.com/science/article/abs/pii/S2214785321052755
12. Machine Learning-Based BPM/Pulse Interval Predictor of Human Being Using ATMega328p-Based Development Board. URL: https://www.researchgate.net/publication/370342695_Machine_learning_based_BPMPulse_interval_predictor_of_human_being_using_ATMega328p_based_development_board
13. Indoor Positioning Using Arduino and Machine Learning in 4 Easy Steps. URL: https://www.hackster.io/news/indoor-positioning-using-arduino-and-machine-learning-in-4-easy-steps-295d39e5e7c9
14. A Lie Detector. URL: http://utpedia.utp.edu.my/id/eprint/7591/1/2005%20-%20A%20LIE%20DETECTOR.pdf
15. Uncovering Lying Detection Using Blood Volume Pulse and Electrodermal Activity. URL: https://www.researchgate.net/profile/Ana-Priscila-Alves/publication/

236131121_Uncovering_Lying_Detection_using_Blood_Volume_Pulse_and_Electrodermal_Activity/links/0c96051ae002e75f68000000/Uncovering-Lying-Detection-using-Blood-Volume-Pulse-and-Electrodermal-Activity.pdf

16. Detecting Lie: A Practical Approach. URL: https://www.researchgate.net/profile/M-Rezki/publication/336096973_Detecting_Lie-A_Practical_Approach/links/5e43a273a6fdccd9659bdd90/Detecting-Lie-A-Practical-Approach.pdf
17. GPS: The Global Positioning System. A global public service brought to you by the U.S. government. URL: https://www.gps.gov/
18. Successful Projects Built with Delphi and FireMonkey. Embarcadero Blogs, 16 Sep. 2020. URL: https://www.embarcadero.com/resources/case-studies
19. Usability Testing | Usability.gov. URL: https://www.usability.gov/how-to-and-tools/methods/usability-testing.html