Designing and evaluating an affordable Arduino-based lie detector prototype

Stanislav V. Pravytskyi, Pavlo V. Merzlykin and Alexander N. Stepanyuk

Kryvyi Rih State Pedagogical University, 54 Universytetskyi Ave., Kryvyi Rih, 50086, Ukraine

Abstract

Lie detection is an important issue in various contexts ranging from criminal investigations to hiring processes. The paper covers the prototyping and evaluation of an affordable Arduino-based lie detector that integrates physiological sensors and machine learning to detect deception. Testing on 20 questions showed the detector achieved 55% accuracy in identifying truth and 45% accuracy in identifying lies, with an overall accuracy of 50%. While further refinements are needed, this prototype demonstrates the challenges of developing an accessible lie-detection system.

Keywords

lie detection, Arduino, physiological sensors, machine learning, LSTM, neural networks

1. Introduction

Lie detection has been a topic of fascination and study for over a century. From early attempts [1] to modern functional magnetic resonance imaging (fMRI) techniques [2], researchers have sought reliable methods to discern truthful statements from deceptive ones.

As a result of amateur electronics platforms affordability, some DIY lie detector prototypes were reported recently [3, 4]. They usually use Arduino, which is one of the most popular amateur platforms, and mostly cover data collection without sufficient processing. In this paper we are trying to estimate the accuracy and feasibility of such DIY lie detectors combined with machine learning techniques.

2. Literature review

Polygraph tests, which measure multiple physiological indicators, became the dominant lie detection tool in the 20th century [1, 5]. However, these tests have been criticized for their lack of scientific validity, vulnerability to countermeasures, and inadmissibility as legal evidence [6, 7, 8].

In recent years, researchers have turned to neuroscience tools like fMRI [2]. These studies suggest that certain brain regions, such as the prefrontal cortex, are more active during lying than truth-telling.

Another emerging trend is the use of machine learning. Recent work [9] has applied deep learning algorithms and reported 57–63% accuracy, which leaves a gap for further research.

Alongside scientific developments, ethical and legal debates surround lie detectors. Critics argue that polygraphs and other lie detection technologies are unreliable, violate privacy rights, and may be misused or overinterpreted in high-stakes contexts like criminal investigations and employment decisions [10]. In the US, the Employee Polygraph Protection Act of 1988 prohibited most private employers from using lie detectors. However, the technology is still widely used in government and law enforcement settings [11].

CS&SE@SW 2024: 7th Workshop for Young Scientists in Computer Science & Software Engineering, December 27, 2024, Kryvyi Rih, Ukraine

coffitronak@gmail.com (S. V. Pravytskyi); ipmcourses@gmail.com (P. V. Merzlykin); alexanderstepanyuk@gmail.com (A. N. Stepanyuk)

https://kdpu.edu.ua/personal/pvmerzlykin.html (P. V. Merzlykin); https://kdpu.edu.ua/personal/omstepaniuk.html (A. N. Stepanyuk)

0000-0002-4017-7172 (P. V. Merzlykin); 0000-0001-9088-2294 (A. N. Stepanyuk)

While researchers continue to develop and test new methods, fundamental questions persist about the accuracy, validity, and ethics of lie detectors. The current study aims to advance this field by designing an Arduino-based lie detector that integrates physiological measurements with machine learning analysis. By examining the potential and limitations of affordable lie detectors, we hope to contribute to the ongoing dialogue.

3. Methods

3.1. Hardware

The lie detector prototype was built using an Arduino UNO development board, chosen for its affordability and extensive community support. The following peripherals were connected to the board:

- SHT20 temperature and humidity sensor
- Pulse sensor to measure heart rate
- ADS1115 16-bit ADC to increase measurement precision

The schematics is shown in figure 1.

3.2. Software

Three main software components were developed:

1. Arduino sketch to read sensor data and print it to the serial port
2. Data collection program to save sensor readings along with truth/lie labels
3. Machine learning model to classify data sequences as indicating truth or lies

The Arduino code initializes the sensors and reads temperature, humidity and pulse values at regular intervals. The data is printed to the serial port to be processed on the computer. Key sections of the code are shown below.

In `setup()` function, which is executed on startup, we initialize all the interfaces. We use dedicated libraries to handle both SHT20 and the pulse sensor. Serial connection is established to transfer the collected data outside.

```
//Libraries to handle the interfaces
#include <Wire.h>
#include "DFRobot_SHT20.h"
#include "PulseSensorPlayground.h"

DFRobot_SHT20 sht20; //SHT20 instance
PulseSensorPlayground pulseSensor; //pulse sensor instance

const int PulseWire = A0; // Pin for Pulse Sensor
const int Threshold = 525; // Threshold value for Pulse Sensor

void setup() {
  Serial.begin(9600); //establishing serial connection

  // Initialize Pulse Sensor
  pulseSensor.analogInput(PulseWire);
  pulseSensor.setThreshold(Threshold);

  // Initialize SHT20
```

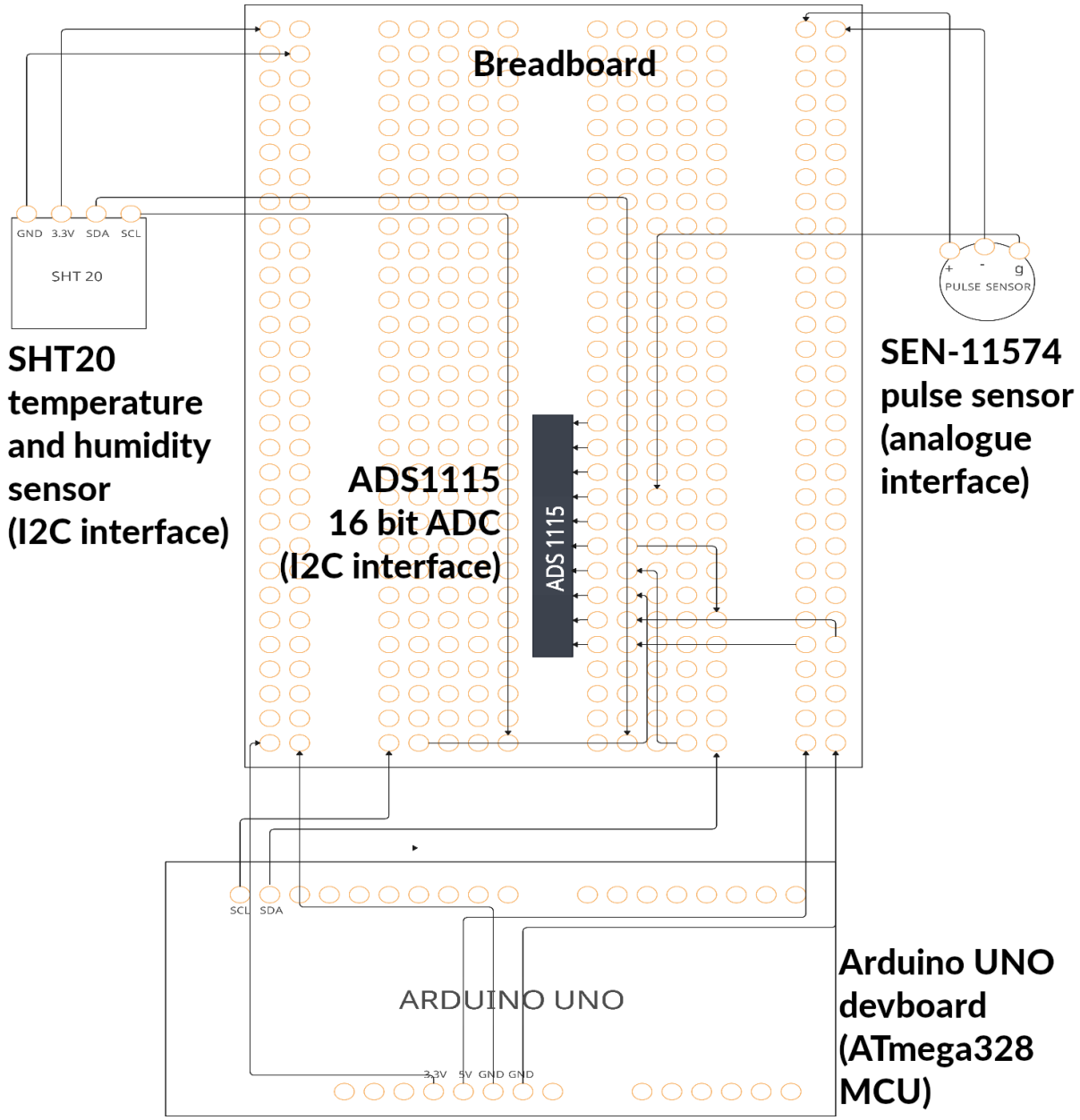

Figure 1: The schematics of Arduino lie detector.

```
  sht20.initSHT20();
  delay(100);
  sht20.checkSHT20();
}
```

Unlike `setup()` function, `loop()` subroutine is executed repeatedly while the device is powered on. It collects sensors and sends it via serial interface for further processing.

```
void loop() {
  //read pulse sensor data
  int myBPM = pulseSensor.getBeatsPerMinute();
  pulseSensor.sawStartOfBeat(); // Update pulse status
  int pulseValue = analogRead(PulseWire);
```

```
  // Read SHT20 data at set interval
  if (currentMillis - lastSHT20ReadTime >= SHT20Interval) {
    lastSHT20ReadTime = currentMillis;
    lastHumd = sht20.readHumidity();
    lastTemp = sht20.readTemperature();
  }

  //send collected data to serial port
  Serial.print(lastTemp, 2);
  Serial.print(",");
  Serial.print(lastHumd, 1);
  Serial.print(",");
  Serial.println(myBPM);
}
```

The data collection script is written in Python and consists of two subroutines. The `read_serial_data()` functions reads messages from serial port and returns them.

```
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
```

The `collect_data()` function asks the user to label each set of collected data as true (1), false (0), or test mode (3). In normal mode, 100 lines of labelled sensor data are saved to a CSV file for each prompt. Test mode collects 2000 lines of unlabeled data for evaluating the model. Key functions are shown below:

```
def collect_data(filename):
    while True:
        data = []
        label = None
        test_mode = False

        print("Start collecting data. Press '1' for truth, '0' for lie, " +
        "'3' for test.")
        label_input = input("Enter label (1-truth, 0-lie, 3-test): ")
        if label_input in ['1', '0']:
            label = int(label_input)
            flush_serial()
            print(f"Start recording data with label {label}.")
        elif label_input == '3':
            test_mode = True
            flush_serial()
            print("Test mode active. Collecting 2000 lines of data.")
        else:
            print("Invalid input. Try again.")
```

```
            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])

            if test_mode and len(data) >= 2000:
                print(f"Collected {len(data)} lines in test mode.")
                data = []
                test_mode = False
                print("Test mode complete. To continue, press '1', '0',"+
                " or '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"Recorded {len(data)} lines with label {label}.")
                data = []
                label = None
                break
```

A long short-term memory (LSTM) neural network was implemented using the Keras library. The model architecture consists of an LSTM layer with 50 neurons followed by a dense output layer with sigmoid activation. It was trained on overlapping sequences of 100 sensor readings to predict the probability that each sequence corresponds to a lie. An 80/20 train/validation split and early stopping based on validation accuracy were used. The key model setup code is shown below:

```
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)

X, y = create_sequences(df, sequence_length)

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'])

checkpoint = ModelCheckpoint('best_model.h5',
                             monitor='val_accuracy',
```

```
                                save_best_only=True, mode='max')

history = model.fit(X_train, y_train,
                    epochs=50, batch_size=32,
                    validation_data=(X_val, y_val),
                    callbacks=[checkpoint])
```

Finally, a real-time prediction program was developed to load the trained model, collect live sensor data from the Arduino, and output truth/lie classifications. Predictions are made on rolling windows of the last 100 sensor values. Key sections are shown below:

```
model = load_model('lie_detector_model.h5')

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("Enter 'start' to begin data collection: ")
                 .strip().lower()
    if user_input == 'start':
        data_buffer = []

        # Clear serial buffer
        ser.reset_input_buffer()

        # Collect data
        while len(data_buffer) < sequence_length:
            if ser.in_waiting > 0:
                data_line = ser.readline().decode('utf-8').strip()
                temperature, humidity, pulse = map(float, data_line.split(','))
                data_buffer.append([temperature, humidity, pulse])

        # When buffer is full, run test
        print("Data collected. Running test...")
        results = []
        for _ in range(100):
            result = predict_from_buffer(data_buffer)
            results.append(result)

        true_count = results.count(1)
        false_count = results.count(0)

        # Output final result
        if true_count > false_count:
            print("Truth")
        else:
            print("Lie")
```

These software components work together to enable the Arduino lie detector prototype to collect physiological data, analyze it using a trained machine learning model, and output lie/truth classifications in real time. The system's modular design allows for easy modification and extension of its capabilities.

4. Results

The lie detector prototype was tested on 20 questions designed to elicit a mix of true and false responses:

1. Is your name Stanislav?
2. Are you 21 years old?
3. Do you have a driver's license?
4. Are you standing right now?
5. Can you play guitar?
6. Is 2+2=5?
7. Have you ever lied to your friends?
8. Can you drive a car?
9. Do you consider yourself an honest person?
10. Did you ever get failing grades in school?
11. Have you ever consumed alcoholic drinks?
12. Do you like your university?
13. Have you ever lied to your parents?
14. Have you ever cheated at work?
15. Do you consider yourself a kind person?
16. Have you ever harmed other people?
17. Are you satisfied with your appearance?
18. Have you ever been to Kyiv?
19. Do you consider yourself a happy person?
20. Have you ever used someone else's property without permission?

The system accurately classified 55% of true statements and 45% of lies, for an overall accuracy of 50% (table 1).

Table 1
Lie detector accuracy on test questions.

Measure	Value
True statements correctly classified	55%
False statements correctly classified	45%
Overall classification accuracy	50%

The results leave significant room for the accuracy improvement. Potential enhancements include:

- Collecting a larger and more varied training dataset
- Tuning the neural network architecture and hyperparameters
- Incorporating additional physiological sensors
- Personalizing models to each individual's baseline physiology

These results highlight both the promise and challenges of developing an affordable lie detection system. With an overall accuracy of 50%, the current prototype performs similarly to the average human lie detector [12].

5. Discussion

This work demonstrates that an inexpensive lie detector can be constructed by interfacing physiological sensors with an Arduino microcontroller and applying machine learning to the collected data. However the LSTM neural network's accuracy (50% overall) is not yet sufficient for practical application.

It falls short of the claims made by polygraph proponents, who often report accuracy rates of 90% [8], although these claims are highly controversial. However another machine learning approach is reported to have 57% accuracy [9], which is comparable to our results.

The Arduino prototype's slight bias towards classifying statements as true (55% accuracy on truths vs. 45% on lies) is consistent with the "truth bias" observed in human lie detection [13].

Several limitations of the current study should be acknowledged. First, the test questions, while designed to elicit a range of truthful and deceptive responses, may not fully capture the complexity and motivation of real-world deception. The stakes in a laboratory setting are inherently lower than in high-consequence contexts like criminal investigations or national security screenings.

Second, the physiological measures used by the prototype (skin temperature, humidity, pulse) are a subset of those typically collected by polygraphs, which also measure respiration and blood pressure. Incorporating additional sensors could potentially improve the system's accuracy.

Third, the current prototype uses a single machine-learning model trained on data from multiple individuals. Developing personalized models tailored to each individual's baseline physiological responses could potentially improve accuracy, as prior work has shown that accounting for individual differences can enhance deception detection [9]. However, collecting sufficient training data from each user to develop robust personalized models would be a significant practical challenge.

6. Ethical challenges

All the mentioned gaps in the results return us to the ethical challenges surrounding lie detection. Since physiological responses measured by lie detectors can be influenced by various factors unrelated to deception, such as anxiety, fear, or medical conditions, it is doubtful that these devices should be used in the cases which can have serious consequences for individuals' lives and reputations.

Moreover, persons subjected to polygraph testing may not fully understand how the test works, which may affect the results. Ensuring that individuals are adequately informed is an important ethical challenge for lie detectors research and practical usage.

Privacy and personal data collection is another issue that should be considered in such tests. There is always a risk that information obtained during testing could be used against individuals in ways they did not anticipate.

The potential lasting impact of undergoing a lie detector test on mental health and well-being is another concern.

Addressing these challenges requires a careful consideration of the implications for individuals and society, as well as a commitment to ethical principles such as validity, informed consent, privacy, and fairness.

7. Conclusion

The development of an Arduino-based lie detector prototype demonstrates the challenges for low-cost, accessible DIY lie detection tools. The prototype's performance, while not yet sufficient for practical application, highlights the promise of this approach.

Substantial improvements in accuracy, reliability, and generalizability will be necessary for such a system to be viable for real-world use. This will likely require more extensive and more diverse training datasets, more sophisticated machine learning models, and the integration of additional physiological and behavioural measures. Personalized models that take into account individual differences may also be a promising direction.

At the same time, it is crucial to recognize that the challenges of lie detection are not purely technological. Even a substantially improved lie detector would still face fundamental questions about the nature of deception, the ethics of its application, and its appropriate role in legal, commercial, and personal contexts.

Lie detector studies should take into account the mentioned ethical considerations and be designed to minimize harm, provide benefits, and respect the autonomy of participants.

Declaration on Generative AI: During the preparation of this work, the authors used Claude 3 Opus in order to: Drafting content, Text translation, Generate literature review, Grammar and spelling check, Content enhancement. After using this service, the authors reviewed and edited the content as needed and takes full responsibility for the publication's content.

References

[1] G. C. Bunn, 'Supposing that truth is a woman, what then?': The lie detector, the love machine, and the logic of fantasy, History of the Human Sciences 32 (2019) 135–163. doi:10.1177/0952695119867022.

[2] E. Rusconi, T. Mitchener-Nissen, Prospects of functional magnetic resonance imaging as lie detector, Frontiers in Human Neuroscience (2013). doi:10.3389/fnhum.2013.00594.

[3] BuildItDR, Arduino Lie Detector — projecthub.arduino.cc, https://projecthub.arduino.cc/BuildItDR/arduino-lie-detector-41f703, 2022.

[4] S. Olfat, Arduino Polygraph Machine (Lie Detector) - ElectroPeak — electropeak.com, https://electropeak.com/learn/arduino-lie-detector-polygraph-machine/, 2016.

[5] C. A. Ruckmick, The truth about the lie detector, Journal of Applied Psychology 22 (1938) 50–58. doi:10.1037/h0059742.

[6] W. G. Iacono, D. T. Lykken, The validity of the Lie detector: Two surveys of scientific opinion, Journal of Applied Psychology 82 (1997) 426–433. doi:10.1037/0021-9010.82.3.426.

[7] D. T. Lykken, Psychology and the lie detector industry, The American psychologist 29 (1974) 725–739. doi:10.1037/h0037441.

[8] W. G. Iacono, Psychology and the lie detector industry: A fifty-year perspective, Biological Psychology 190 (2024) 108808. doi:10.1016/j.biopsycho.2024.108808.

[9] N. Rodriguez-Diaz, D. Aspandi, F. M. Sukno, X. Binefa, Machine learning-based lie detector applied to a novel annotated game dataset, Future Internet 14 (2022). doi:10.3390/fi14010002.

[10] J. J. Furedy, R. J. Heslegrave, Validity of the Lie Detector: A Psychophysiological Perspective, Criminal Justice and Behavior 15 (1988) 219–246. doi:10.1177/0093854888015002008.

[11] V. Mellema, Lie Detector Tests, in: The Encyclopedia of Civil Liberties in America: Volumes One-Three, volume 2, 2015, pp. 567–568. doi:10.4324/9781315699868-398.

[12] T. H. Feeley, M. J. Young, Humans as lie detectors: Some more second thoughts, Communication Quarterly 46 (1998) 109–126. doi:10.1080/01463379809370090.

[13] T. R. Levine, C. N. H. Street, Lie-truth judgments: Adaptive lie detector account and truth-default theory compared and contrasted, Communication Theory 34 (2024) 143–153. doi:10.1093/ct/qtae008.