Smart Mailbox with Mail Classification

Rigdha Acharya, *Member, IEEE and* Yang Peng, *Member, IEEE*

Abstract—Users get instant notification and spam control for emails but the postbox that provides them with important documents such as passports, debit cards, orders from online purchases etc. is unable to notify users when important mail arrives or to help users filter out junk mail. Additionally, users must coordinate time off from work or drive to local post offices during business hours to sign for important packages that require delivery confirmation. Postal Services across the world are looking for IoT based solutions to streamline the mail delivery scenario and adapt to delivery time and security requirements because of the boom in parcel delivery due to ecommerce. This project implements a connected mailbox that will notify users when important documents and parcels arrive, help users avoid junk mail and electronically sign for important packages without requiring the user to be physically present at home.

Index Terms—postbox, internet of things, internet of postal things, mail classification, delivery confirmation.

1 Introduction

THE postbox is over 100 years old [1] and unable to provide notification or spam filtering to users. Postal services such as USPS in USA and Posti in Finland have attempted strategies to provide value to mail senders and receivers such as mail tracking, requesting signature on certain parcels, managing delivery times etc. However, there is no solution for filtering out spam mails, allowing remote delivery signature or ensuring users are home to receive delivery. Providing these services is not only convenient to the user but also profitable to couriers such as USPS, DHL and Posti because it allows them to reduce mail delivery attempts when the user is not available, get faster confirmation of delivery and charge more for end-to-end package tracking including when a package is delivered, signed and retrieved by the user. IoT systems are used in various home appliances today to monitor homes from a smart thermostat to a doorbell camera. A smart mailbox can be compared to a video doorbell but with additional requirements for working with scenarios when the user is not home, securely holding the mail and classifying the incoming mail for the user. In this paper, we implement a connected mailbox that can provide delivery confirmation to reduce repeated delivery attempts for couriers and detect spam allowing users to focus on important mail items.

2 Related Work

We surveyed the USPS and DHL visions for connected mailbox, VoIP based solution for users to remotely sign for packages, strategies for saving power, using GPU to speed up image processing and using color distribution to classify mail as flyer or letter. We explore some of the proposed strategies for smart mailbox below.

2.1 Internet of Postal Things (IoPT)

The United States Postal Services (USPS) issued a report in 2015 outlining the opportunities to use IoT to improve postal collection, handling, and delivery while providing new business opportunities for USPS and other branches of the Government through a connected neighborhood model. USPS envisions a smart mailbox with sensors that can collect and transmit postal and non-postal data such as delivery time, temperature, locking mechanism etc. These mailboxes will communicate with the mail person's smart watch or smart phone to auto lock/unlock or notify the owner for lock/unlock scenarios. In addition, the mailbox sensor would scan incoming mail's barcode to generate electronic delivery signatures [1] which would remove the need for users to arrange time off work to receive their parcels. The smart mailbox would increase the security of the delivery items by providing remote lock capabilities and 24/7 monitoring for the user. In addition, USPS believes the ability to notify senders when their mail or promotional materials are delivered and retrieved by the user would provide full visibility to the sender making it beneficial for advertisers, senders and receivers ultimately leading to increased profits for USPS [3].

The system proposed by the USPS would provide many functionalities to the user for secure delivery but does not provide an implementation and it also does not classify the incoming message into important documents or advertisements. In addition, the system would only work with USPS delivery. While USPS is the largest mail delivery provider in the US, this solution leaves DHL, UPS and other carriers without access to user mailbox unless it is provided by USPS as a platform to other courier companies.

2.2 IoT in Logistics

DHL and Cisco envision a smart mail box where delivery triggers an alert to the user's phone using GSM. With this method, the users can be notified of incoming mail and they can track mail when they are on vacation. DHL also envisions a Parcelbox which allows for bigger parcels to be delivered in a temperature controlled setting to account for the growth in ecommerce and allow for delivery of grocery and other sensitive items [4]. DHL anticipates connecting the smart Parcelbox to a user's connected fridge that tracks

- *R. Acharya is a student at the University of Washington, Bothell.*
- *Dr. Y. Peng is a professor at the University of Washington, Bothell.*

supplies that are running low to allow for new scenarios such as auto replenishment and anticipatory shipping. Although the DHL paper does not specify the technology it will use, it sets the goal for improving logistics using IoT by leveraging it's vast network of couriers worldwide and increasing the profitability of couriers by integrating with smart refrigerators and other IoT sensors already used in connected homes.

2.3 Smart VoIP Postbox with Confirmation Receipt using IoT Technology

The Smart VoIP Postbox with Confirmation Receipt using IoT Technology paper proposes a connected mailbox that sends notification when a parcel arrives, acts as the communication medium between the courier and the user, allows users to remotely unlock mailbox and sends confirmation of receipt for mail delivery [7]. The architecture involves a central system where couriers and users can register to get delivery notifications. Once a courier arrives at the connected mailbox, the courier presses a button to make a VoIP call to the user who can inspect the parcel and open the mailbox. The implementation detailed in the paper uses a Raspberry Pi 3 Model B board with built in Wi-Fi to connect to the central server as well as user's smart device. While this system allows for delivery notification and interaction with the user, it does not classify the incoming mail. It also poses problems when the user is driving or otherwise unable to check their phones while the delivery is happening and could increase cost for couriers because of lost time waiting for users to respond to their calls. However, this system creates a third-party service that can be used by various carriers such as USPS, DHL, UPS etc.

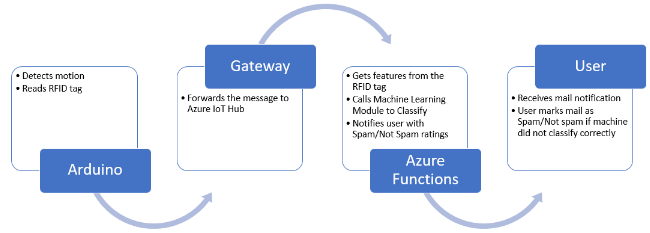
2.4 Intelligent Mailbox with Centralized Parallel Processing

IoT devices are constrained by their power utilization and the need to recharge them. In a mailbox that is attached to a user's home, it may be possible to easily recharge the IoT devices installed in the mailbox. However, for communities that centrally manage mailboxes, the power usage can become an economic and logistic issue. This paper proposes devices that can be turned off or put on standby when they are not in use [5]. In this experiment the Wi-Fi adapter was turned off by the Raspberry Pi board when not in use.

2.5 LetterTwitter: Smart Mailbox for Spam-filtered Notification of Received Letters

The authors propose a mail classification system that takes pictures of incoming mail when the postal service worker drops the mail in a special front slot in a connected mailbox. The front slot uses a magnetic switch to detect that a mail is present triggering a picture to be taken. Once the picture is taken, the system classifies the mail as letter or junk mail. The system uses the distribution of color to classify the mail as letter (mostly white) and junk mail (mostly colorful) [6]. When multiple mail pieces are inserted, the system compares additional images against prior image to calculate the change in number of high saturation pixels to determine if new mail is junk or letter. While the system is quite

Fig. 1. Serial logs from Arduino when motion is detected and mail is scanned



simple, it can identify letters vs advertisements. However, the system cannot distinguish between advertisements that look like letters where they are mailed in white envelopes. It also does not address signing for package delivery or mailbox security.

3 System Model

This project creates a smart mailbox system which saves couriers repeated delivery attempts by automating the delivery confirmation system and saves receivers time by identifying spam mail. Based on USPS IoPT report, we use RFID tags to automatically issue mail delivery confirmation. We provide a third party technology so that the system can be used by USPS, UPS, DHL and private couriers as well. The third-party system allows users to subscribe to events from a particular mailbox. In a community mailbox scenario such as apartments, the system allows the community manager permissions to subscribe/unsubscribe users from a mailbox.

We implement a mail classification technique to classify mail as Spam or Normal based on the sender and receiver information. This classification technique works at two levels, first we learn from all user's data to determine if a particular set of features relate to spam mails, in addition, we learn from the receiver's data to determine if a particular sender or receiver pattern is spam. This allows us to identify regular statements from a bank to existing customers vs. promotional messages sent to potential customers.

For the hardware, we use an Arduino Mega with motion sensor, RFID reader and Wi-Fi module. We conserve power by deactivating the Wi-Fi and RFID modules when there is no motion in the mailbox. We use Azure cloud offerings such as IoT hub, functions, container services for machine learning to setup our system.

Users are notified of incoming mail and the classification result using SMS notification. Users are also able to update the labels for each mail using an ASP.NET web intrface allowing the machine learning module to be re-trained with spam labels provided by users.

Figure 1 shows the data flow for the system.

4 Design and Implementation

This project implements the model defined above using an Arduino board connected to Azure cloud offerings.

4.1 Hardware

4.1.1 Arduino Mega 2560

Arduino Mega 2560 is a microcontroller board with both 3.3V and 5V support. This board was chosen because of the need to support both 5V and 3.3V sensors and because of the high number of input and output pins available in this board since the project uses multiple sensors. Although this project did not use any shields, this board is compatible with various shields designed for Arduino, particularly the Wi-Fi shields.

4.1.2 Passive Infrared Sensor

Passive Infrared Sensors detect the infrared light from objects in their view. These are mainly used as motion sensors and we use the HC-SR501 PIR Motion Sensor to detect the opening of a mailbox. Once the mailbox is opened, the system activates the Wi-Fi connection and RFID reader. The delay time of the module was set to 5 minutes to allow the courier time to finish scanning all the letters.

4.1.3 ESP8266 Wi-Fi module

ESP8266 is a low cost Wi-Fi module with a full TCP/IP stack and 1 M B flash storage. This module comes with the AT firmware by Espressif [12]. The AT firmware is basic and led to some of the connectivity issues such as inability to use certificates. The module was also slow in connecting to Wi-Fi even when transferring just a few bytes of data.

4.1.4 RFID Reader

Radio Frequency Identification (RFID) uses electromagnetic fields to identify and track objects. This project uses MFRC522 which is a contactless RFID reader and writer that operates at 13.56 MHz [9] The MFRC522 has a short range of 1-5 cm. We were unable to increase the range to 5cm and had to be less than 1 cm away from the reader to read the passive tags used in the project.

4.1.5 RFID Tags

RFID tags are cost effective ranging from $0.15 to $5+. This project uses passive RFID tags that were acquired for $0.15 each. The tags can easily be added to envelopes to track mail through the postal system.

4.2 Software

4.2.1 Gateway Web API

Since the ESP8266 Wi-Fi module does not support certificate based authentication for connection to Azure IoT Hub, we used a simple gateway API allowing the ESP8266 to simply post plain text GET messages to this server. This server simulates a gateway device with connection to Azure IoT Hub. The gateway API was built using ASP.NET and was deployed to an Azure Web App service to route the requests from the device to Azure IoT hub.

4.2.2 Azure Functions

Azure functions allow users to author programs that are triggered on various inputs such as time, messages in a queue, new files in a blob storage etc. The functions are serverless meaning that the user does not need to provide a virtual machine to run the code. This setup allows for faster development and deployment. Azure functions also have rich support for Azure IoT hub and was chosen for this project based on the examples and tutorials available to integrate with Azure IoT Hub. We had attempted to simulate an IoT Edge device on the gateway server but we were unable to run the latest IoT Edge device SDK for python on a MacOS. Therefore, we used Azure Functions to gather information about the mail tag received from the device, call a machine learning module and then notify the user. The Functions application was written in C# using the latest libraries for Azure Functions.

4.2.3 Machine Learning Module

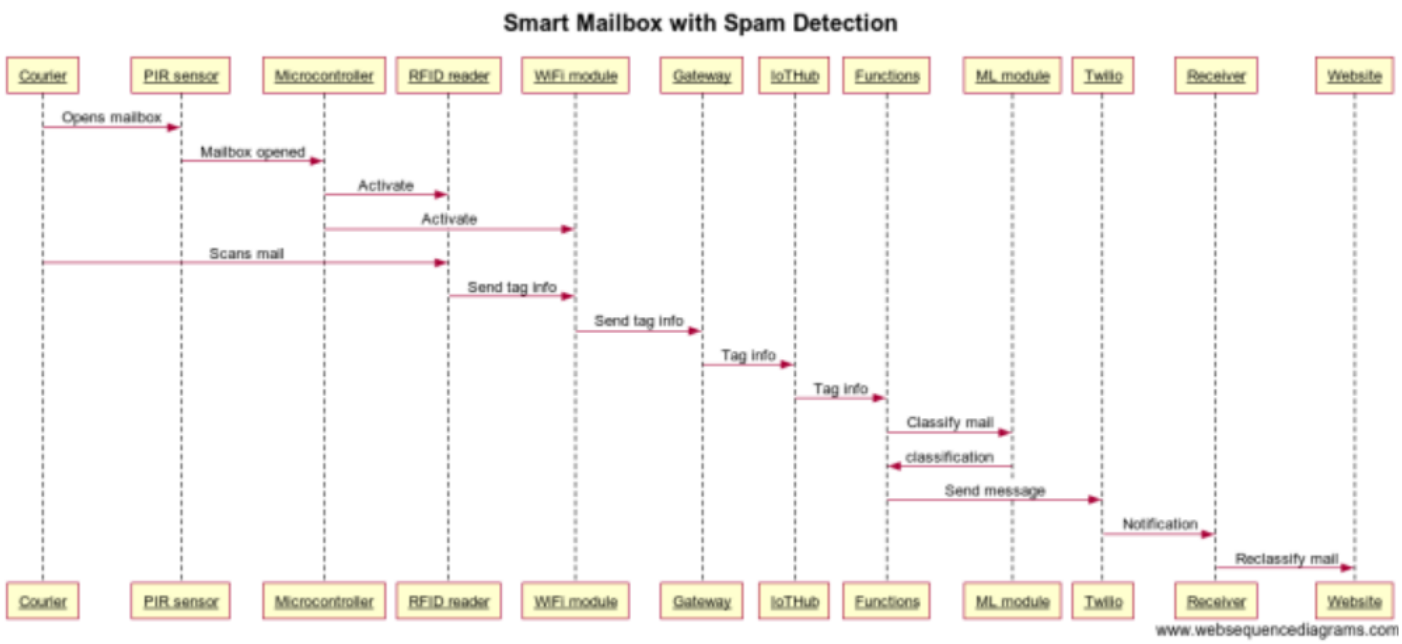
This project uses a decision tree classifier which was trained with randomly generated data. The dataset had mostly negative (non-spam) values. The model was trained with a 65/35 split with 65% of the data used for training. When the model was tested with the withheld data, it produced an accuracy of 75%. We then setup a docker image that was deployed to an Azure Container Services instance which triggers the prediction to run when it receives the REST API messages from the Functions application. The machine learning model was created using Python and the scikit-learn library [11].

4.2.4 User Notification and Interaction

This project uses the Functions application to send SMS notification once a mail has been labelled as spam or normal mail by the machine learning module. Once the user receives the notification, they are able to click on the provided link and mark a mail as spam or not spam allowing us to retrain the model based on the data provided by the user. This allows mail spam detection to be trained just like email spam detection which benefits from buttons users can click to report junk emails. Figure fig:website shows the website where users can rate the incoming mail.

The SMS notifications were done through a trail account from Twilio [10] and the website for users to rate mail was hosted on the same server as the gateway API and developed using ASP.NET. We chose Twilio because of the trail account offer and because of the rich APIs available for C# and Azure Functions. Figure fig:sms shows the SMS notifications received by the user.

Fig. 2. Sequence Diagram of the Smart Mailbox



4.3 System Architecture

Figure 2 shows the sequence diagram of the smart mailbox. Once a courier opens the mailbox, the PIR sensor detects motion and activates the RFID reader. It also initiates the Wi-Fi connection. The courier scans the mail item which is read by the RFID reader. For each mail item, the ESP866 module sends a HTTP GET message to the Gateway endpoint. The gateway simply forwards the message to an Azure IoT Hub account under the /messages/event path.

The Functions application is deployed in Azure which listens to messages in the IoT Hub channel under the /messages/event path. When the functions application receives the messages, it gets the sender, receiver and send date data from the database and sends an HTTP message to the Machine Learning module hosted in Azure Container Services to classify the mail as spam or normal. The machine learning module simply runs the prediction on a saved model to generate the classification.

Once the classification is received, the functions application sends an SMS notification using the Twilio platform [10] to the receiver. The receiver then clicks on the website URL in the SMS notification to view incoming mail message. The receiver can also mark a mail as spam or normal if he/she feels the message has been misclassified by the system. This allows the system to continue learning based on user input.

5 Evaluation and Result

For the evaluation we used randomly generated data for 150 mail pieces along with 5 new mail parcels represented by 5 different RFID tags. The Arduino board was powered through the USB connector which allowed us to view the messages sent by the board in the serial monitor. As mentioned in the Design and Implementation section, the system is not secure because of the need to send plain text GET requests from the ESP8266 module which doesn't support certificates. A gateway server was deployed to Azure Web Apps to overcome limits of ESP8266. The Azure functions and machine learning modules were also deployed to Azure. Figure 3 shows the messages received in the Arduino console when the motion sensor is activated and an RFID tag is scanned.

Fig. 3. Serial logs from Arduino when motion is detected and mail is scanned

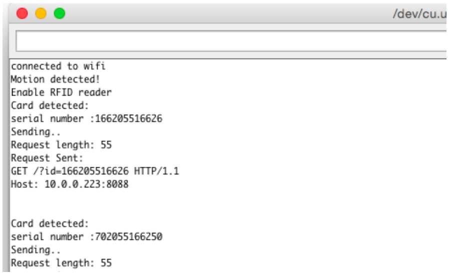
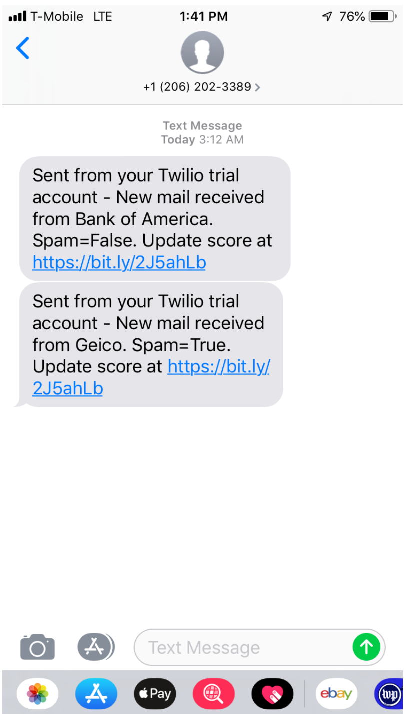


Fig. 4. SMS notification sent to user



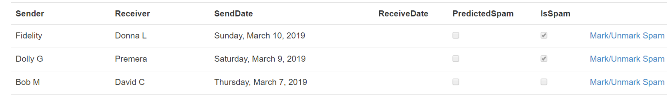
The hardware system was able to detect motion when the mailbox was opened and activated the RFID reader and initiated the Wi-Fi connection. Once the RFID tag is read and the information is sent by the Arduino device, it moves on to scanning other RFID tags.

The tag scanned was received by the Gateway API which forwarded it to an Azure IoT hub. Once IoT hub received a message, it triggers the Functions app which calls the Machine Learning module and notifies the user. Figure 4 shows the notification received by the user for two mail pieces received.

However, the RFID reader had a very short range and the tag had to be less than 1 cm away from the reader for it to be scanned. In addition, the Wi-Fi connection through ESP8266 was slow introducing delays in the system. This can be improved by using a more powerful Wi-Fi module such as ESP 32. Figure 3 below shows the message received in the Arduino console when the mailbox is opened, and a mail tag is scanned. The current system sends the user one SMS for each received mail, however, this led to too many messages sent through Twilio during the testing phase delaying some messages by more than 5 minutes.

Once the user is notified of incoming mail, they can

Fig. 5. Website for users to rate incoming mail



Sender	Receiver	SendDate	ReceiveDate	PredictedSpam	IsSpam	
Fidelity	Donna L	Sunday, March 10, 2019		☐	☑	Mark/Unmark Spam
Dolly G	Premera	Saturday, March 9, 2019		☐	☑	Mark/Unmark Spam
Bob M	David C	Thursday, March 7, 2019		☐	☐	Mark/Unmark Spam

classify mail as spam or normal using the ASP.NET website. This website was slow to load on cell phones such as Apple iPhone 7. This limits the usage of the application because a user receiving notifications on the cellphone will want to use the cellphone to rate the mail. Figure 5 shows the website used by the users to mark a mail item as spam or not spam.

6 Conclusion and Future Work

This project delivers a smart mailbox using the Arduino Mega 2560 board and Azure cloud services. We were able to use motion sensors, RFID reader solution that can detect spam. We are also able to pass the RFID tag information to the cloud which then processes the messages, runs classification and notifies the user of the received mail and label. This provides the users the ability to recognize spam mail as well as provides instant delivery confirmation to avoid repeated trips by couriers or the need for users to be present to sign for important mail documents.

Although we were able to scan letters individually, the RFID reader was not able to scan multiple messages at once and had a small range of less than 1 cm. We can allow multiple mail pieces by using a more powerful reader with longer reading ranges and by using active RFID tags.

The mail classification module was trained with random data. This module had an accuracy of 0.75 during our tests. In the future, data should be collected based on actual mail sent and received by users to improve the accuracy. In addition, the system allows users to reclassify mail items, but the model is not being retrained yet. We can write simple python modules to retrain the model every night using Azure Functions.

The device security should also be improved by using a more powerful Wi-Fi chip such as NodeMCU or ESP32 which support certificates so that the device can directly talk to Azure IoT Hub. This will allow us to remove the Gateway API. In addition, the machine learning module can run on the Edge device removing the need for the Functions application. The ability to deploy the machine learning module to the Edge device will speed up the classification process allowing computations to be done closer to the user's mailbox.

The future work mentioned above improve the security and performance of the system. The system functionality has been demonstrated in the Evaluation section and this project delivers a complete prototype with all the features needed in a smart mailbox.

REFERENCES

[1] USPS OIG Office. (2015).The Internet of Postal Things. [online] USPS IOG. Available at: https://www.uspsoig.gov/sites/default/files/document-library-files/2015/rarc-wp-15-013_0.pdf [Accessed 6 Feb. 2019].
[2] Waxer, C. (2019). Digital SOS: How technology can save the USPS. [online] Computerworld. Available at: https://www.computerworld.com/article/2855146/digital-sos-how-technology-can-save-the-usps.html [Accessed 6 Feb. 2019].
[3] USPS OIG Office. (2013). Strengthening Advertising Mail by Building a Digital Information Market. [online] USPS IOG. Available at: https://uspsoig.gov/sites/default/files/document-library-files/2015/rarc-wp-14-002_0.pdf [Accessed 6 Feb. 2019].
[4] DHL Trend Research. Internet of Things in Logistics. [online] DHL. Available at: https://discover.dhl.com/content/dam/dhl/downloads/interim /full/dhl-trend-report-internet-of-things.pdf [Accessed 6 Feb. 2019].
[5] Muhammad, A., & Ur Rehman, N. (2016). Intelligent mailbox with centralized parallel processing. Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2016 17th IEEE/ACIS International Conference on, 255-259.
[6] Tsukada, K., Mizushima, Y., Ogata, A., & Siio, I. (2010). LetterTwitter: Smart mailbox for spam-filtered notification of received letters. Proceedings of the 12th ACM International Conference Adjunct Papers on Ubiquitous Computing - Adjunct, 439-440.
[7] Sangkong, J., & Ongtang, M. (2017). Smart VoIP postbox with confirmation receipt using IoT technology. ACM International Conference Proceeding Series, 71-75.
[8] Dandavate, Dinanath. (2017). Smart Box System (POC) Using NB-IoT Technology. International Journal of Science and Research (IJSR). 6. 966-968.
[9] Standard Performance MIFARE and NTAG frontend. [online] NXP Semiconductors. Available at: https://www.nxp.com/docs/en/data-sheet/MFRC522.pdf [Accessed 9 Mar. 2019].
[10] Twilio Docs: API References. [online] Twilio.com. Available at https://www.twilio.com/docs/ [Accessed 9 Mar. 2019].
[11] Documentation of scikit-learn 0.20.3 [online]. Scikit-learn.org. Available at https://scikit-learn.org/stable/documentation.html [Accessed 9 Mar. 2019].
[12] ESP8266 Technical Reference. [online] Espressif. Available at https://www.espressif.com/sites/default/files/documentation/esp 8266technical_reference_en.pdf [Accessed 9 Mar. 2019].