Carrier App: Leveraging Machine Learning for Secure Crowdshipping

Mrs. M S Lakshmi Devi
Assistant Professor of Computer Science, Government First Grade College -VARTHUR, Bangalore 87

Abstract -This paper explores a new approach to delivery using crowd-shipping. Unlike traditional company-owned delivery vehicles, crowd-shipping utilizes everyday people with personal vehicles to deliver packages for compensation. This system offers greater flexibility and cost savings. The focus here is on a dynamic system where in-store customers are recruited as crowd-shippers to deliver online orders on their way home. However, these customers can choose to accept or decline the delivery offer. The challenge is to find the best way to match online orders with available crowd-shippers and determine the optimal compensation for each delivery. To address this, the paper proposes a two-step optimization model. This model first determines the best matches between orders and crowd-shippers, then calculates the optimal compensation scheme. Through computer simulations, the model demonstrates a significant cost reduction, averaging 7.30% lower compared to traditional delivery methods.

Keywords: Crowd-shipping, Machine learning, Image Processing, Acceptance Uncertainty

INTRODUCTION

Our daily commutes are often a frustrating trifecta: time wasted in gridlock, the stress of navigating congested roads, and the environmental burden of all those extra vehicles. This proposal tackles these challenges head-on, while simultaneously creating a new opportunity to earn income during your daily travels. Imagine a mobile app that connects people with errands to those who can deliver items quickly and conveniently. The app, built with a robust tech stack including Java, XML, PHP, SQL, Firebase, and developed within Android Studio, utilizes location data to create a seamless bridge between "Senders" with items to deliver and "Riders" already traveling in the same direction. Riders have complete control over their earnings potential. They can browse deliveries that seamlessly integrate with their existing commutes and choose the ones that best suit their route and time constraints. No detours, no extra hassle, just a chance to turn your daily commute into a productive money-making opportunity. The benefits extend far beyond Riders. "Senders" experience significantly faster delivery times compared to traditional methods, eliminating the need to travel by themselves and saving valuable time. Live tracking functionality provides peace of mind, while a secure OTP verification system and robust authentication protocols with industry-standard encryption and decryption techniques ensure a smooth, safe, and trustworthy experience for everyone involved. In essence, this innovative app offers a win-win-win situation. Riders can earn extra income by leveraging their existing commutes, our cities experience reduced traffic congestion and pollution, and everyone benefits from a convenient and time-saving solution for on-demand deliveries. It's a smarter way to navigate our daily lives, helping the environment, saving time, and putting some extra cash in your pocket.

LITERATURE SURVEY

This project draws inspiration from a variety of existing technologies and innovative ideas. Through a thorough literature review and careful comparison of different approaches, we've identified the most promising path forward for implementation. This process has opened up a range of possibilities for building the app and its functionalities.
A research paper titled "Matching Models for Crowd-Shipping Considering Shipper's Acceptance Uncertainty" by Shixuan Hou and Chun Wang [1] delves into the complexities of optimizing delivery costs in crowd-shipping systems that leverage in-store customers. The core challenge lies in identifying the most efficient pairings between customers and online orders to minimize overall delivery expenses. The authors provide a comprehensive review of existing

approaches to address this challenge. Archetti et al. propose a multi-start heuristic matching strategy, while Gdowska et al. present a heuristic algorithm focused on maximizing the number of orders offered to couriers for optimal profitability. Kafle et al. shift their focus to utilizing pedestrians and cyclists for last-mile deliveries, employing a mixed-integer programming model and a Tabu search algorithm to achieve optimal matching. Wang et al. introduce a two-pronged approach: first, they utilize pruning techniques to tackle large-scale delivery tasks by reducing network complexity. Second, they propose a heuristic algorithm specifically designed to assign deliveries to existing car trips, aiming to maximize the overall benefit of the crowd-shipping system. The effectiveness of these various methods is validated through experiments leveraging real-world data from major cities like Singapore and Beijing. This research highlights the ongoing efforts to optimize crowd-shipping systems, particularly when considering the inherent uncertainty of customer acceptance in such models. By exploring diverse approaches like multi-start heuristics, maximizing order allocation, and leveraging alternative delivery methods, researchers are paving the way for a more efficient and cost-effective crowd-shipping landscape.

The research article "The Digital Economy of Crowdsourcing: Crowd Shipping Model as E-Business" by Hassaan, Yaqot, and Menezes[2] explores the burgeoning field of crowd-shipping within the transportation industry. They acknowledge the rise of the sharing economy and how businesses are capitalizing on Information and Communication Technologies (ICT) and the Internet of Things (IoT) to create a more responsive delivery system that utilizes the "crowd." Beyond just describing the concept, the authors propose a generic business model for a typical crowd-shipping application. They delve deeper with a process flow for using commercial airplanes to deliver lightweight packages. This is further enhanced by introducing mathematical models that progressively build upon each other. The first model represents a basic one-to-one matching scenario. The second model incorporates additional factors like weight, delivery time, and desired date, leading to more nuanced matching. Finally, the third model offers senders flexibility by allowing them to choose from multiple potential carriers, potentially leading to more competitive pricing. The platform acts as a central hub, analyzing user data to identify the most suitable matches between senders and carriers. To optimize these matches, it employs a distance optimization technique, considering both pickup and delivery distances. This ensures efficient pairings and minimizes unnecessary travel, potentially revolutionizing the way we deliver goods by offering a cost-effective, adaptable, and potentially faster alternative to traditional methods.

PROPOSED SYSTEM

This research focuses on creating a real-time system that extracts key features from images of delivery items. This emphasis on feature extraction serves a critical purpose: ensuring the security of the delivered items. By automatically analyzing images in real-time, the system can potentially detect discrepancies or anomalies that might indicate tampering or damage during the delivery process. This innovative approach has the potential to significantly enhance security and peace of mind for both senders and receivers of delivered goods.

A. Image Acquisition

The very first step in our system's process is image acquisition. Here, users capture photos of the delivery item using their mobile phone or tablet cameras. These images are captured from both the sender's and receiver's sides to ensure a complete record of the item's condition throughout the delivery journey. It's important to note that the captured images may vary in resolution depending on the device used.

B. Image Processing

Following image acquisition, the system takes the initial RGB images captured from senders and receivers and prepares them for further analysis. To streamline processing and reduce computational demands, the system converts these images into a grayscale format. This essentially removes color information, focusing solely on the intensity variations within the image, which is often sufficient for feature extraction tasks. Additionally, all images are resized to a standard dimension of 400x400 pixels. This ensures consistency in the data fed into the subsequent stages of the image processing pipeline, allowing for more accurate feature comparisons and ultimately, more reliable security checks. However, real-world image capture can introduce imperfections that could hinder the system's ability to accurately assess the delivery item's condition. To address this challenge, the system incorporates a technique called

Contrast Limited Adaptive Histogram Equalization (CLAHE) [4]. Imagine dividing the image into a grid of smaller squares. CLAHE operates on these squares individually, adjusting the distribution of brightness values within each one. This targeted approach is particularly effective in situations where the image has uneven lighting or suffers from low contrast. By enhancing the image quality in this way, CLAHE ensures that crucial details about the item's condition are not obscured or misinterpreted during the feature extraction process. Finally, grayscale images are susceptible to a particular type of noise known as "salt and pepper noise." This manifests as random black and white pixels scattered throughout the image, potentially masking important details. The system combats this issue by applying a Median Filter [5]. This filter works in a localized manner, analyzing a small surrounding area of pixels around a specific point in the image. The filter then replaces the central pixel's value with the median value of all its neighbors. By systematically iterating through the entire image, the Median Filter effectively removes the unwanted noise, resulting in a clean and accurate representation of the delivery item. This preprocessing stage is critical for ensuring that the subsequent feature extraction step operates on high-quality data, ultimately leading to more robust security checks throughout the delivery journey.

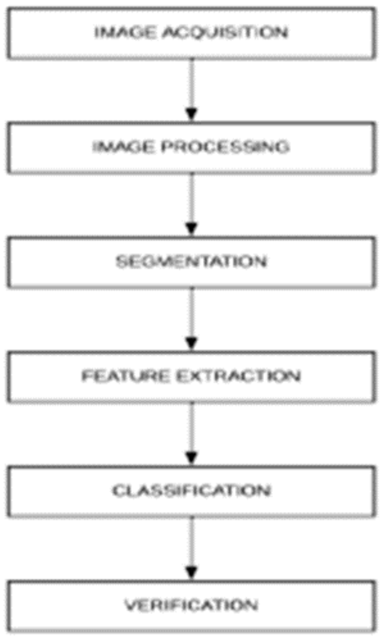

Fig.1.1: Proposed System

C. Image Segmentation

After the meticulous pre-processing stage, the system employs a sophisticated technique called segmentation to pinpoint the specific area of interest within the image. Imagine a scenario where a box is delivered and there's a small dent on its corner. Ideally, we want the system to focus only on that specific damaged area, not the entire box. Segmentation achieves this by essentially partitioning the image. Here, the system utilizes an advanced variant called "Improved Fuzzy C-Means (FCM)" for segmentation. Standard FCM algorithms can become overwhelmed by complex images with a high number of data points (high dimensionality). This can hinder their ability to effectively segment the image and isolate potential damage. The "Improved FCM" technique addresses this limitation by incorporating a data compression step at the outset. This step significantly reduces the dimensionality of the input data, making it more manageable for the segmentation process. Think of it as simplifying the image data without discarding crucial details. The compression involves two key operations working in tandem:

➔ Quantization: Here, a specific number of the least significant bits are masked out from the feature values within the image data. This essentially reduces the range of possible values without discarding crucial information about the image's intensity variations.

Imagine dimming a complex image down to a few distinct brightness levels, but cleverly preserving the details that signal potential damage.

➔ Aggregation: This step groups together feature vectors that share similar intensity values. By grouping these vectors, the system can more efficiently identify potential areas of damage within the segmented portion of the image. Continuing the box analogy, imagine the segmentation process has isolated the corner area of the image. Aggregation helps the system identify clusters of pixels within that corner area that share similar intensity levels, potentially corresponding to the dent on the box.

Through this two-pronged approach of quantization and aggregation, the Improved FCM technique effectively isolates the area of interest within the preprocessed image. This allows the system to focus its analysis on a more manageable set of data, enabling a more robust security assessment by pinpointing potential damage indicators with greater precision. This is a critical step in ensuring the integrity of the delivery process and providing peace of mind to both senders and receivers.

D. Feature Extraction

Following the image segmentation stage, the system employs a powerful technique called Grey Level Co-occurrence Matrix (GLCM) to delve deeper into the potential damage indicators. Imagine you have a zoomed-in view of the suspected dent on the box, courtesy of the pre-processing and segmentation steps. GLCM excels at analyzing the intricacies of texture within this specific area. It achieves this by creating a unique table that reveals how often pairs of pixels with similar gray levels appear adjacent to each other in the image. By meticulously examining this table, the system can extract valuable information about the structural patterns within the texture. This information can pertain to the randomness or smoothness of the surface, providing crucial clues about potential damage. A key strength of GLCM lies in its versatility. It can be applied at various scales and orientations. This flexibility allows the system to meticulously examine the texture from diverse perspectives, ensuring it doesn't overlook any subtle details that might signal damage. Think of it as having a high-precision magnifying glass that can also be tilted and adjusted to gain a comprehensive picture of the dent's texture. By analyzing the GLCM at different scales and orientations, the system can effectively identify irregularities in the texture that might be indicative of damage, even if they're not immediately apparent in the original image. This multifaceted approach to texture analysis plays a critical role in ensuring the accuracy of the system's security checks. It empowers the system to move beyond simply detecting the presence of a dent and delve deeper, potentially identifying characteristics of the dent (like its depth or sharpness) that could differentiate between minor cosmetic imperfections and more serious damage. This level of detail significantly enhances the system's ability to provide a more comprehensive assessment of the item's condition throughout the delivery journey.

$Pd=\{((a1,b1),(a2,b2)) : I(a1,b1)=r, I(a2,b2) = s\}||$

E. Classification

Culminating the meticulous image processing pipeline, the system leverages the power of artificial intelligence (AI) to deliver a final verdict on potential damage. This AI engine takes the form of an Artificial Neural Network (ANN), a sophisticated learning machine specifically trained to identify damage indicators within product images. Imagine the ANN as a vast network of interconnected processing units, mimicking the structure of the human brain. During a dedicated training phase, the system feeds the ANN a colossal dataset of images. This dataset serves as a treasure trove of knowledge, meticulously categorized to include examples of both undamaged and damaged items. Each image within the dataset is accompanied by critical information - the texture features extracted from the GLCM analysis (these features act as the inputs the network learns from) and the corresponding information about the actual damage status of the item (this is the target the network is trained to identify). By meticulously processing countless examples, the ANN progressively refines its ability to recognize the intricate patterns within textures that correlate with damage. It's akin to a student diligently studying a vast collection of examples, gradually developing the expertise to distinguish undamaged products from those with potential issues. Once this training phase is complete, the ANN transforms into a powerful diagnostic tool. Presented with a new, unseen image (like the one containing the suspected dent on the box), the ANN can analyze the extracted texture features. By drawing upon the knowledge gleaned from its training data, the ANN generates an output, essentially indicating the likelihood of damage within the image. This output serves as the system's final assessment, providing a reliable and data-driven verdict on the condition of the delivered item. It's a culmination of the entire image processing pipeline, translating the raw visual data into a clear and actionable conclusion about the product's integrity. This AI-powered approach significantly enhances the system's accuracy in detecting damage, offering a significant leap forward compared to traditional rule-based methods. It empowers the system to move beyond simply identifying the presence of a dent and delve deeper, potentially identifying characteristics of the damage (like its depth or sharpness) that could differentiate between minor cosmetic imperfections and more serious structural issues. This level of detail empowers both senders and receivers with a more comprehensive understanding of the item's condition throughout the delivery journey, fostering trust and transparency within the entire delivery process.

➔ Preparing the data: Following the intricate dance of image pre-processing and segmentation, the system arrives at a critical juncture: extracting the essence of potential damage from the isolated area of interest. Here, the Grey Level Co-occurrence Matrix (GLCM) steps into the spotlight. Imagine you have a high-resolution close-up of the suspected dent on the box, courtesy of the system's meticulous image manipulation. GLCM excels at analyzing the intricacies of texture within this specific region. It achieves this by creating a unique table that reveals how often pairs of pixels with similar gray levels appear adjacent to each other in the image. By meticulously examining this table, the system can extract a wealth of valuable information about the structural patterns within the texture. This information pertains to the randomness or smoothness of the surface, providing crucial clues about potential damage. However, GLCM doesn't stop at generating a generic report. The system is specifically programmed to focus on extracting a select group of five key texture features that are particularly sensitive to damage indicators. These features act like a fingerprint of the texture, uniquely identifying its characteristics:

● Correlation: This measures how closely pixels with similar gray levels are distributed within the image. In an undamaged area, you'd expect a natural correlation between neighboring pixels. Damage can disrupt this inherent relationship, causing the correlation value to deviate from the norm, potentially signaling trouble.

● Homogeneity: This captures the essence of smoothness or uniformity within the texture. Imagine a pristine, unblemished surface. The gray levels of its pixels would transition smoothly, resulting in a high homogeneity value. A dent or scratch, however, would disrupt this uniformity, causing the homogeneity value to plummet, indicating a potential anomaly.

● Contrast: This reflects the variation in gray levels within the image. An undamaged surface might exhibit a gradual shift in gray levels, leading to a specific contrast distribution. Damage, on the other hand, can introduce sharp variations in gray levels, throwing off the overall contrast distribution and acting as a red flag for the system.

● Entropy: This signifies the level of randomness or disorder within the texture. An undamaged surface typically exhibits a certain level of order in its gray level distribution. Damage can disrupt this order, introducing randomness and leading to a higher entropy value. By monitoring entropy, the system can identify potential signs of damage.

● Energy: This measures the local uniformity of the gray level distribution. Imagine a smooth, undamaged surface where neighboring pixels share similar gray levels. This local uniformity translates to a high energy value. Damage can disrupt this local uniformity, causing the energy value to fluctuate and potentially signaling the presence of an issue.

➔ Data Preprocessing for Neural Network Input: There's an important step before we unleash the power of the Artificial Neural Network (ANN). The data extracted from the image processing pipeline, specifically the five texture features like correlation and entropy, needs to be presented in a format that the ANN can understand. Here's where MATLAB's neural network toolbox comes into play. It has a specific preference for data organization: features along rows and samples along columns. Unfortunately, our current data might not comply with this preferred format. Imagine a table where each row represents a different image, and each column represents a specific texture feature. This flips the desired structure on its head. To bridge this gap, we employ a technique called transposition. Think of it as rotating the table by 90 degrees. This simple manipulation ensures the data aligns perfectly with the ANN's expectations. By meticulously arranging the features and samples in the correct format, we ensure the ANN receives the information it needs in a way it can readily process. This paves the way for the ANN to embark on its training journey, ultimately transforming it into a powerful tool for identifying damage within the delivered items.

➔ Constructing the Neural Network Classifier: Now that we have meticulously prepared our data by pre-processing the images and extracting key texture features, it's time to introduce the heart of the system's intelligence: the artificial neural network (ANN). This sophisticated learning machine will play a pivotal role in classifying the condition of the delivered items. The chosen architecture for this task is a cascade forward neural network. This type of network excels at processing information layer by layer, gradually extracting increasingly complex patterns from the data. To achieve optimal performance, the network will be equipped with a single hidden layer composed of 28 neurons. These neurons act as the information processing units within the network, working together to identify the subtle nuances within the texture features that signal potential damage. It's important to note that the initial weights assigned to the connections between these neurons will be randomized. This randomness helps the network avoid getting stuck in local minima during training and encourages it to explore the data landscape more effectively.

➔ Training and Testing the Neural Network: With the neural network architecture in place, we now focus on the critical process of training it to effectively distinguish between damaged and undamaged items. To achieve this, we strategically split our meticulously prepared data (containing the five texture features) into two distinct sets: the training set and the testing set. Think of the training set as the ANN's personal tutor. It contains a substantial portion of the data, complete with the corresponding information about the actual damage status of the items (remember, this data was meticulously labeled during the training phase described earlier). The ANN will relentlessly analyze this data, iteratively adjusting the weights between its neurons to progressively improve its ability to identify the patterns within the texture features that correlate with damage. This training process continues for a predetermined number of cycles or until the network reaches a point of diminishing returns, where further training no longer yields significant improvements. While the ANN jones it's skills on the training set, it's crucial to have an unbiased evaluation of its performance. This is where the testing set comes in. This set represents a completely independent selection of data, unseen by the ANN during training. By presenting the testing set to the trained network and comparing its predictions against the actual damage status of the items within the set, we can obtain a reliable measure of the network's generalizability and accuracy in real-world scenarios. In essence, the testing set acts as a final exam, allowing us to assess how well the ANN has learned to identify damage indicators and translate them into accurate classifications.

➔ Evaluating the Classifier's Performance: Once the neural network has completed its rigorous training phase, it's time to put its newfound knowledge to the test. This is where the testing set we meticulously carved out from our data comes into play. Remember, this set comprises entirely unseen examples, data the network has never encountered during training. By feeding these fresh images into the trained network, we can observe its performance in a scenario that mimics real-world application. The network will analyze the texture features extracted from the test images and generate its classifications – undamaged or damaged. We can then compare these predictions against the actual damage status of the items within the testing set. This comparison allows us to calculate the network's accuracy – essentially, how often it correctly identifies damaged and undamaged items. By analyzing these results, we gain valuable insights into the network's effectiveness in real-world situations. If the accuracy is high, it indicates the network has successfully learned the intricate patterns within textures that signal damage and can be reliably deployed in the delivery system. If the accuracy falls short, it might necessitate revisiting the network architecture, training parameters, or even the selection of texture features to optimize performance.

➔ Assessing Performance with the Confusion Matrix: Having evaluated the network's performance on the testing set with a simple accuracy metric, we can delve deeper using a powerful tool called the confusion matrix. Imagine a grid-like table where rows represent the network's predictions (damaged or undamaged) and columns represent the actual damage status of the items in the testing set. As the network analyzes each image, its corresponding prediction is recorded in the relevant cell of the confusion matrix. This matrix goes beyond just providing a basic accuracy score. By analyzing the distribution of predictions across the table, we can extract valuable insights into the network's performance. The confusion matrix breaks down the results into four key categories:

● True Positives (TP): These represent the number of images where the network correctly classified a damaged item as damaged.

● True Negatives (TN): These represent the number of images where the network correctly classified an undamaged item as undamaged.

● False Positives (FP): These represent the number of images where the network incorrectly classified an undamaged item as damaged (also known as a Type I error). This scenario might lead to unnecessary precautions being taken for an undamaged item.

● False Negatives (FN): These represent the number of images where the network incorrectly classified a damaged item as undamaged (also known as a Type II error). This is a critical error as it signifies a missed damaged item.

By analyzing these values, we can calculate key performance metrics like accuracy, precision, and recall. Accuracy, expressed as a percentage, reflects the overall proportion of correct classifications made by the network. Precision, also a percentage, indicates how often the network's prediction of "damage" was truly accurate (avoiding false positives). Recall, again a percentage, measures the network's ability to identify all the actual damaged items within the testing set (avoiding false negatives). A well-performing network will strive for a high overall accuracy, coupled with high precision and recall. This indicates the network is not only making a significant number of correct classifications but also minimizing both types of errors (false positives and false negatives). By scrutinizing the confusion matrix and the derived performance metrics, we gain valuable insights into the network's strengths and weaknesses, allowing for targeted improvements if necessary. This ensures the system maintains a high level of reliability and accuracy in real-world delivery scenarios.

➔ Leveraging AI for Secure Deliveries: This system takes a multifaceted approach to ensuring the security of delivered items. At its core lies the artificial neural network (ANN), a powerful learning machine adept at identifying damage indicators within product images. The ANN acts as the system's diagnostic tool, meticulously analyzing the texture features extracted from pre-processed images. These features, like correlation and entropy, capture the essence of the surface texture and provide crucial clues about potential damage. By drawing upon the knowledge gleaned from its comprehensive training data, the ANN generates a verdict on the condition of the item. This verdict isn't a simple binary classification (damaged or undamaged). The system delves deeper, potentially identifying characteristics of the damage (like depth or sharpness) that could differentiate between minor cosmetic imperfections and more serious structural issues. This nuanced analysis empowers the system to provide a more comprehensive assessment of the product's condition. These detailed damage assessments, along with other relevant property information, are then used to generate comparison measures. Imagine a scenario where the ANN detects a small scratch on a delivered phone. The system wouldn't just flag the damage; it would provide a comparison against established benchmarks or previous conditions of the phone, allowing for a more informed judgment about the severity of the issue. This multifaceted approach, combining AI-powered damage detection with insightful comparisons, ensures a more robust and reliable security system for deliveries.

This delivery app leverages a unique model that connects riders (travellers) with senders who need items delivered to destinations along the riders' planned journeys. Senders can pre-add items to the app, specifying details like product descriptions and shipping charges. Riders can then browse these listings and choose deliveries that fit their travel route. Once a rider selects a delivery, they can initiate their journey, pick up the item from the sender, and then continue on their route until they reach the receiver's location (which is a stop before the rider's final destination). Upon successful delivery and verification of the item's condition, the rider receives payment. To ensure item security, the app utilizes image recognition powered by Google's AI-based ML Kit. Upon reaching the destination, the receiver captures an image of the delivered item using the app. This image is then analyzed by object detection, which identifies and verifies the product's specifications. This comprehensive verification process helps ensure the item hasn't been tampered with during transit. For added transparency, the app tracks rider status using Google Maps API. The rider's status updates to "On the Way" after collecting the item from the sender, and then transitions to "Delivered" upon successful verification by the receiver. To ensure rider accountability, the app collects government-issued IDs (like Aadhaar cards in India) during the rider signup process. This multi-layered approach fosters trust and security within the delivery ecosystem.

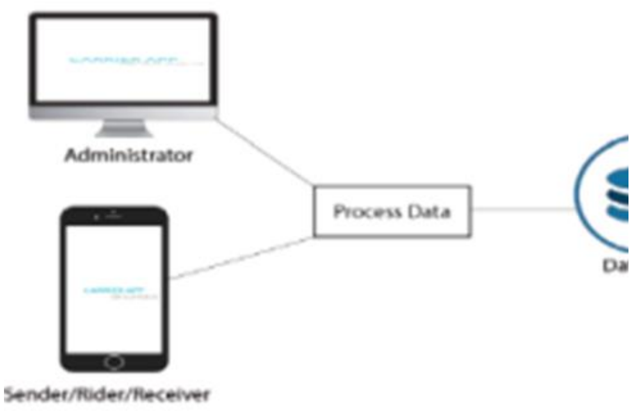

Fig.1.2: System Design - Architecture Design

The app caters to three distinct user roles: sender, rider, and receiver. Users choose their role upon signing up and can then perform actions specific to that role. For instance, senders can pre-list items for delivery, specifying details and pricing. Riders can browse these listings and select deliveries that align with their travel routes. Receivers can track deliveries, verify the condition of received items, and confirm successful deliveries. Additionally, the system incorporates an administrator role. This behind-the-scenes role handles the technical aspects of the app, including maintenance, updates, programming, monitoring, and service management. Data management is crucial for the app's functionality. All user roles (sender, rider, and receiver) interact with the database for various purposes. The database stores user information, delivery details, and other relevant data. It also facilitates data validation, tracking delivery progress, and ensuring overall system integrity. The app employs a combination of SQLite and Firebase for robust database management. SQLite provides a lightweight solution for local data storage on user devices, while Firebase offers a cloud-based backend for centralized data management and scalability. This hybrid approach ensures both offline functionality and real-time data synchronization. The administrator maintains and services both aspects of the database to guarantee smooth app operation.

The app's functionality can be visualized through a use case diagram, which details how different user roles interact with the system. Here's a breakdown of the key functionalities for each role:

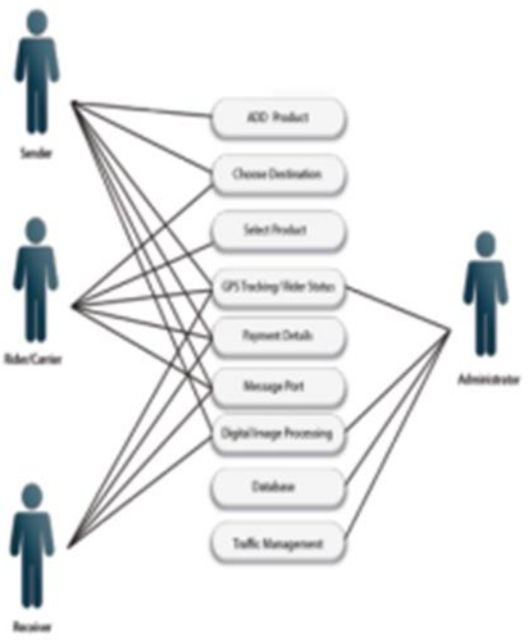

Fig.1.3: Use Case Diagram

Senders can manage the entire delivery process from their fingertips. They can add new items, specifying details and dimensions. They can choose the destination and track the delivery progress in real-time using GPS tracking and rider status updates. The app also facilitates payment management, allowing senders to specify payment details and potentially manage confirmation. Additionally, senders can communicate with riders through a message port for any delivery-related coordination. Some senders might also have access to optional image processing features to prepare or verify product images.

Riders can leverage the app to find and manage delivery opportunities. They can browse available deliveries and select those that align with their planned travel routes. The app provides GPS tracking functionality, allowing riders to track their progress towards the sender's location and the receiver's destination. Payment details for accepted deliveries are readily accessible within the app, ensuring riders receive proper compensation. Riders can also communicate with senders and receivers through the message port for any delivery-related inquiries.

Receivers have functionalities to track and manage deliveries upon arrival. They can track the rider's status to see when the delivery arrives and use the integrated payment system to settle the payment. The app empowers receivers with clear communication channels through a message port, allowing them to send messages to the sender or rider for any questions or concerns. A key feature for receivers is image processing, specifically product detection. This allows them to verify the received item's condition upon delivery. They can capture an image of the product and leverage the app's AI capabilities to confirm it matches the description.

Administrators play a critical behind-the-scenes role in maintaining the app's overall health. They handle system maintenance, updates, and technical aspects like programming and service management. Additionally, administrators oversee the entire delivery ecosystem using GPS tracking for all riders. They are responsible for maintaining the app's database (potentially a combination of SQLite and Firebase) to ensure data integrity, security, and smooth operation. Furthermore, they manage the app's image processing functionalities, ensuring their accuracy and effectiveness in tasks like product verification. Optionally, administrators might also have access to traffic management features, allowing them to optimize delivery routes or suggest alternative routes to riders based on real-time traffic conditions.

This use case diagram provides a comprehensive overview of how each user role interacts with the app and the features available to them. It highlights the distinct functionalities for senders, riders, and receivers, while also emphasizing the administrator's role in maintaining the system's technical health.

Image recognition, akin to showing a picture to a friend and asking them what it is, deals with classifying images into predefined categories ("cat," "car"). This fundamental computer vision (CV) task empowers machines to "see" and interpret the visual world. It serves as the foundation upon which more complex CV tasks are built, progressively refining a machine's understanding of visual data. Here's a breakdown of how these tasks build upon image recognition:

➔ Image Classification with Localization: Imagine taking a picture of your dog in the park. Image recognition with localization would not only identify the object as a "dog" but also pinpoint its exact location within the image using a bounding box, a rectangular outline around the object.

➔ Object Detection: This task tackles scenarios with multiple objects in a single image. Like finding all your friends in a group photo, object detection can categorize various objects within an image (e.g., people, bicycle) and highlight their precise locations using bounding boxes. It acts like a visual search engine, pinpointing all the relevant elements within an image.

➔ Object Segmentation: Moving beyond the realm of bounding boxes, object segmentation delves deeper into the image, identifying individual pixels that belong to each object. This intricate analysis allows the machine to not only recognize a cat in an image but also understand the shape and outline of the cat by classifying every pixel that contributes to the cat's form.

➔ Instance Segmentation: If you have a group photo with several friends wearing similar clothing, instance segmentation can differentiate between individual objects within the same category. It can distinguish between each person in the group, providing a more granular understanding of the image content.

By building upon the foundation of image recognition, these increasingly sophisticated CV tasks work together to give machines a progressively stronger grasp of the visual world. It allows machines to not only identify objects but also precisely locate them, understand their shapes and outlines, and even differentiate between multiple instances of the same object within a scene. This enhanced visual intelligence has far-reaching applications in various fields, from self-driving cars navigating through traffic to medical imaging analysis for disease detection.

CONCLUSION AND FUTURE SCOPE

This paper proposes a novel approach to delivery services, focusing on sustainability, efficiency, and economic benefits. Our project tackles the challenge of congested roads and expensive delivery costs by leveraging crowd-sourced shipping. We achieve this through a user-friendly mobile application, accessible on both Android and iOS platforms. The app boasts a simple and intuitive interface, making it easy for anyone to participate. Security is paramount in our system. We guarantee the safe delivery of products through robust security measures. Additionally, we incorporate cutting-edge machine learning techniques via image processing. This technology empowers users to verify the condition of delivered items, fostering trust and transparency within the delivery ecosystem. Furthermore, our project goes beyond just efficient deliveries. It embodies a strong social and environmental conscience. By creating opportunities for people to earn extra income through deliveries, we contribute to the economic well-being of our users. This social aspect is further amplified by our commitment to eco-friendly practices, promoting a sustainable delivery model that benefits both people and the planet.

REFERENCE

[1] Le, T.V., Stathopoulos, A., Van Woensel, T. and Ukkusuri, S.V., 2019. Supply, demand, operations, and management of crowd-shipping services: A review and empirical evidence. Transportation Research Part C: Emerging Technologies, 103, pp.83-103.

[2] Najafabadi, S., 2019. Designing an On-demand Dynamic Crowdshipping Model and Evaluating Its Ability to Serve Local Retail Delivery in New York City (Doctoral dissertation, The City College of New York).

[3] Punel, A., 2019. Understanding the "Crowd" in "Crowd-shipping": Evaluating the Performance of a Crowdsourced Delivery System–User Behavior, Agent Interaction, and Network Variables (Doctoral dissertation, Northwestern University).

[4] Dietmann, K., 2020. Crowdshipping the last-mile delivery-an empirical investigation into the crowd's willingness to participate as crowdshipping drivers.

[5] Buldeo Rai, H., Verlinde, S., Merckx, J. and Macharis, C., 2017. Crowd logistics: an opportunity for more sustainable urban freight transport?. European Transport Research Review, 9, pp.1-13.

[6] Bardasco San José, D., 2016. Analysis of Crowdsourcing Logistics in B2C e-commerce: Costs and Environmental perspective.

[7] Briffaz, M. and Darvey, C., 2016. Crowd-shipping in Geneva exploratory and descriptive study of Crowd-shipping.

[8] Manners-Bell, J. and Lyon, K., 2019. The logistics and supply chain innovation handbook: disruptive technologies and new business models. Kogan Page Publishers.

[9] Jaller, M., Otero, C., Pourrahmani, E. and Fulton, L., 2020. Automation, electrification, and shared mobility in freight.

[10] Nguyen, K., 2019. Analyzing the Influence of Perceived Attributes towards Consumer Adoption of Crowdsourced Delivery in Vietnam.

[11] Nguyen, K., 2019. Analyzing the Influence of Perceived Attributes towards Consumer Adoption of Crowdsourced Delivery in Vietnam.

[12] Cronkhite, I.S., 2015. Evaluate innovative strategies leveraging existing USPS capabilities and resources to advance as a vital 21st century service organization (Doctoral dissertation, Massachusetts Institute of Technology).

[13] Tang, C.S. and Veelenturf, L.P., 2019. The strategic role of logistics in the industry 4.0 era. Transportation Research Part E: Logistics and Transportation Review, 129, pp.1-11.

[14] Davis, A.W., Polydoropoulou, A., Stathopoulos, A., Farooq, B., Bhat, C.R., Antoniou, C., Zhang, J., Goulias, K.G., Mohammadian, K., Kamargianni, M. and Mokhtarian, P.L., 2020. Workshop summary and research themes. In Mapping the Travel Behavior Genome (pp. 635-673). Elsevier.

[15] Skilton, M., 2016. Building digital ecosystem architectures: A guide to enterprise architecting digital technologies in the digital enterprise. Springer.

[16] Cohen, B., 2017. Post-capitalist Entrepreneurship: Startups for the 99%. Taylor & Francis.

[17] Rimmer, P.J. and Kam, B.H., 2018. Consumer logistics: Surfing the digital wave. Edward Elgar Publishing.

[18] Eulaerts, O., Joanny, G., Giraldi, J., Fragkiskos, S. and Perani, S., 2020. Weak signals in Science and Technologies-2019 Report. Publications Office of the European Union. https://publications. jrc. ec. europa. eu/repository/bitstream/JRC118147/kjna29900en

n. pdf, accessed in January.
[19] Skilton, M. and Skilton, M., 2015. How to Build the Right Digital Enterprise for Payoff and Monetization. Building the Digital Enterprise: A Guide to Constructing Monetization Models Using Digital Technologies, pp.155-184.
[20] Wilson, K., 2015. Every crowd has its silver lining: How crowdsourcing is conceived, practised and how it creates value (Doctoral dissertation, University of Melbourne, Department of Management and Marketing).
[21] Moavenzadeh, J., 2015, October. The 4th industrial revolution: Reshaping the future of production. In World economic forum (p. 57). Amsterdam: DHL Global Engineering & Manufacturing Summit.

ANNEXURE (ALGORITHM)

```
import cv2 # Assuming OpenCV for image
processing
# Function for image preprocessing
def preprocess_image(image_data):
# Convert to grayscale and resize (replace with
specific functions) grayscale_image =
cv2.cvtColor(image_data,
cv2.COLOR_BGR2GRAY) resized_image =
cv2.resize(grayscale_image, (400, 400))
# Apply CLAHE and Median Filter (replace with
specific functions) preprocessed_image =
apply_CLAHE(resized_image)
preprocessed_image =
apply_median_filter(preprocessed_image) return
preprocessed_image
# Function for image segmentation (placeholder for
Improved Fuzzy C-Means) def
segment_image(preprocessed_image):
# Implement Improved Fuzzy C-Means with
quantization and aggregation segmented_image =
perform_improved_FCM(preprocessed_image)
return segmented_image
# Function for feature extraction (placeholder for
GLCM)
def extract_features(segmented_image):
# Apply GLCM and extract texture features
features =
calculate_GLCM_features(segmented_image)
return features
# Function for data preprocessing for neural network
def prepare_neural_network_data(features):
# Transpose features for ANN input
transposed_features = np.transpose(features)
return transposed_features
# Placeholder for pre-trained neural network model
def classify_damage(features):
# Load pre-trained ANN model and use it for
classification
# Replace with your specific model loading and
prediction functions classification_result =
pre_trained_model.predict(features)
return classification_result
# Main program flow (replace with actual image
acquisition logic) image_data =
cv2.imread("path/to/image.jpg") # Replace with
image capture logic
preprocessed_image =
preprocess_image(image_data) segmented_image =
segment_image(preprocessed_image) features =
extract_features(segmented_image) nn_data =
prepare_neural_network_data(features) classification
= classify_damage(nn_data)
print("Damage Classification:", classification)
```