Journal of Autonomous Intelligence (2024) Volume 7 Issue 2
doi: 10.32629/jai.v7i2.1381

ORIGINAL RESEARCH ARTICLE

Lost item identification model development using similarity prediction method with CNN ResNet algorithm

Jonathan Prawira, Theresia Ratih Dewi Saputri*

Informatics, School of Information Technology, Universitas Ciputra Surabaya, Surabaya 60219, Indonesia

*** Corresponding author:** Theresia Ratih Dewi Saputri, theresia.ratih@ciputra.ac.id

ABSTRACT

Background: Incidents of personal belongings being lost often occur due to our negligence as human beings or criminal acts such as theft. The methods used to address such situations are still manual and ineffective. The manual process of reporting lost items requires significant time and effort. Additionally, matching the information of lost items with the found ones becomes increasingly difficult, and finding the original owners can be time-consuming. **Objectives and Methods:** This research aims to develop an approach that aids the community in the management of lost items by incorporating a process of item identification. It proposes the creation of an iOS-based prototype model that implements image comparison and string matching. The ResNet-50 architecture extracts features from images, and the Euclidean Distance method measures similarity between these features. Natural language processing used for text pre-processing and employs the cosine similarity metric to assess textual similarity in item descriptions. **Result and Conclusion:** By combining Euclidean distance and cosine similarity values, the model predicts similar lost item reports. Image comparison provides an accuracy result of 29.96% correctness, while string matching with 97.92% correctness. Thorough testing and validation confirm the model's success across different reports.

Keywords: image comparison; Euclidean distance; string matching; natural language processing; cosine similarity

ARTICLE INFO

Received: 20 October 2023
Accepted: 16 November 2023
Available online: 13 December 2023

1. Introduction

The incidents of personal belongings being lost often occur due to our negligence as humans or as a result of criminal acts such as theft. However, the resulting impact can be fatal. In 2004, a survey conducted by a marketing company in Boston on American adults revealed that, on average, Americans spend 55 minutes per day searching for misplaced items, even though they know their whereabouts but cannot find them[1]. A survey conducted in the UK with 3000 participants in early 2012 found that people can lose up to nine items per day[1]. Another survey conducted by newswire PR in 2013 showed that the time wasted searching for lost items can result in financial losses comparable to throwing away money[1]. The survey indicated that approximately $177 billion is wasted annually in searching for lost or misplaced items in the workplace. In an era of advanced technology, people report lost items through various media platforms[2,3], but the methods are still manual and ineffective. Manual reporting is time-consuming and discourages reporting[4], leading to an increasing number of lost items. Traditional methods can't cope with the demand, making it challenging to match and identify original owners[2,5]. For example, the FBI reports a high number of stolen bicycles with a low 5% return rate to owners[1]. To address these issues, researchers aim to create a model that facilitates lost item management by implementing an item identification process, ensuring proper identification for item return to rightful owners.

Zhou et al.[6] proposed a study aimed to address the issue of increasing lost items and inefficient manual search methods. It introduced a method to simplify comparing photos of lost objects with a database, resulting in a practical search approach. The study conducted experiments using transfer learning models, combining MobileNetv2 with Convolutional Block Attention Module (CBAM) Attention to create a photo matching network. The system achieved a 96.8% test accuracy. Even though this work achieves high accuracy, the system only can be run in laptop which makes it inefficient. Jeong et al.[7] developed a hierarchical deep learning-based lost item management system with image classification and ranking systems. The ranking system prioritizes registered properties based on weights, and the classification system utilizes TensorFlow and Inception-v3. Experimental results showed convenience and user-friendliness, with higher reliability compared to non-hierarchical approaches. Suryani and Edy[2] proposed a study of Term Frequency-Inverse Document Frequency (TF-IDF) and cosine similarity methods were used in an Android "Lost & Found" app. TF-IDF weighs word-document relationships, while cosine similarity measures similarity between objects using keywords. The study found that these methods can be effectively used for searching purposes. Using a confusion matrix, they achieved an 88% accuracy rate with a 12% error rate, indicating the app's good performance. However, this proposed approach only focusing on the description of the item. Relying only in text description can be difficult for finding item because people may use different wording in describing same item.

Therefore, this study proposes an iOS-based prototype using image comparison and string matching techniques. It utilizes the ResNet-50 architecture for image feature extraction due to its advantages, as highlighted later in the study, and its better performance when compared to the architecture used in a previous study. Euclidean distance is used to calculate image similarity. In string matching, natural language processing pre-processes text, and cosine similarity measures text similarity in lost item reports. The model combines Euclidean distance and cosine similarity scores to predict report matches with stored records.

The remainder of this paper is organized as follows. The section 2 discussed the materials and methods that use to address the research problem. Next, in the section 3 the implementation result is presented. The section 4 covers discussion on the result of implementing the proposed method. Finally, the conclusion and future works are explained in section 5.

2. Materials and methods

In this research, the model is divided into 2 parts, namely image comparison and string matching. The model cannot rely solely on image matching. This is to address the possibility of two different items having a similar appearance. To increase the accuracy, additional information in the form of descriptions or item details is required. With the addition of a description of the item and information on how the item could be lost, it will make it easier to match the found and lost items. Therefore, the proposed model will combine image matching with descriptions matching.

Photos of different item types were used as the dataset. In this proposed approach, image pre-processing and convolutional neural network (CNN) with the ResNet-50 algorithm were employed. Euclidean distance measured similarity values for extracted images, while text pre-processing and cosine similarity calculated similarity between texts. The overall process conducted in this research can be seen in **Figure 1**. The whole methodology process can be seen in **Figure 2**.

The model cannot rely solely on image matching. Additional information in the form of descriptions or item details is required. This is to address the possibility of two different items having a similar appearance. Thus, confirmation can be made through comprehensive descriptions.

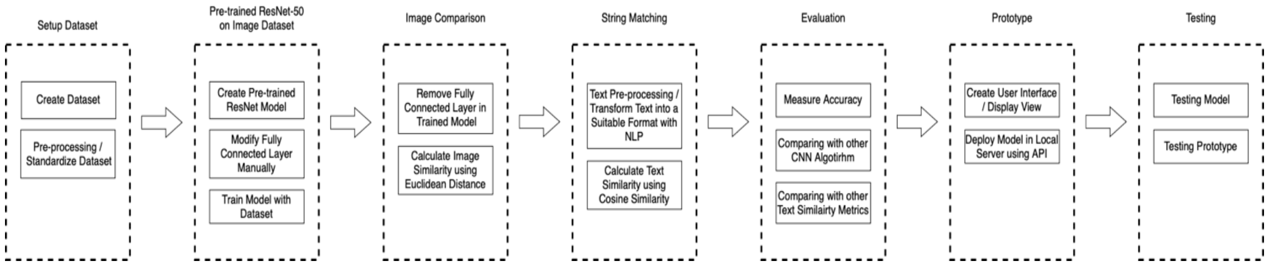

Setup Dataset
Create Dataset
Pre-processing / Standardize Dataset
Pre-trained ResNet-50 on Image Dataset
Create Pre-trained ResNet Model
Modify Fully Connected Layer Manually
Train Model with Dataset
Image Comparison
Remove Fully Connected Layer in Trained Model
Calculate Image Similarity using Euclidean Distance
String Matching
Text Pre-processing / Transform Text into a Suitable Format with NLP
Calculate Text Similarity using Cosine Similarity
Evaluation
Measure Accuracy
Comparing with other CNN Algotirhm
Comparing with other Text Similairty Metrics
Prototype
Create User Interface / Display View
Deploy Model in Local Server using API
Testing
Testing Model
Testing Prototype

Figure 1. The process and stages of model creation research.

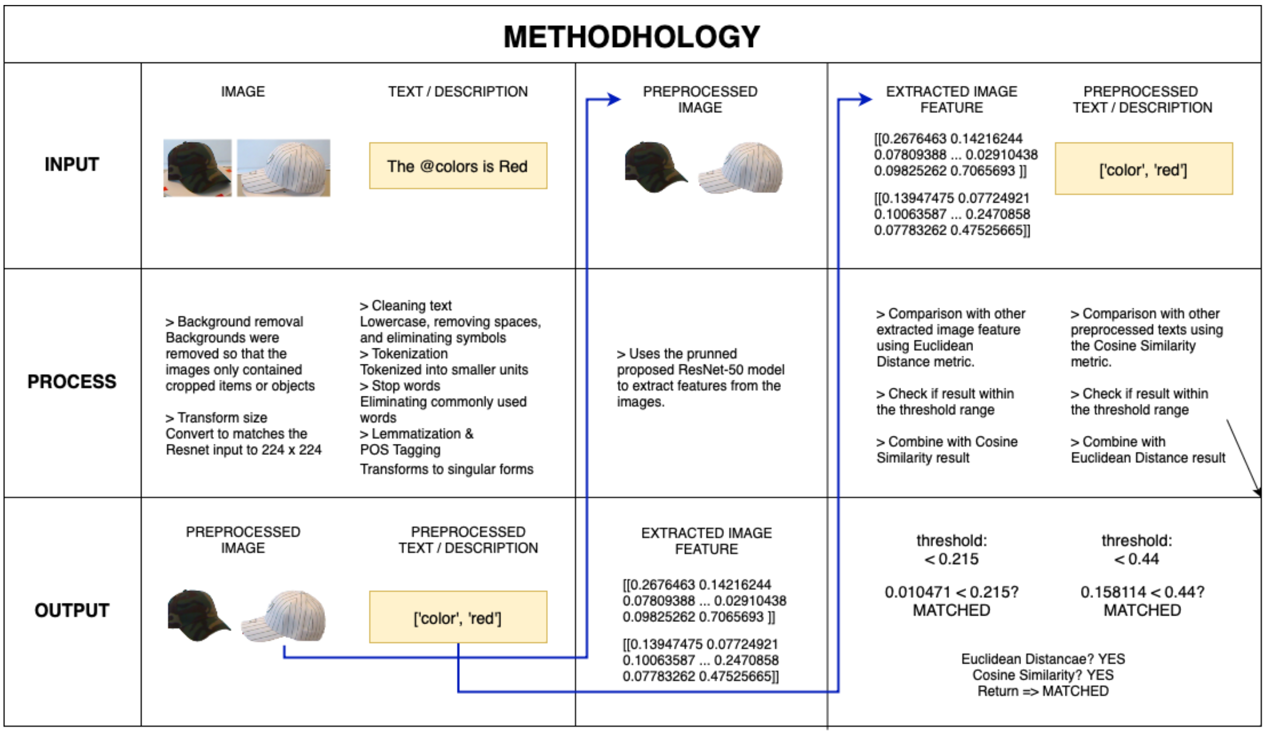

METHODHOLOGY
INPUT
IMAGE
TEXT / DESCRIPTION
The @colors is Red
PREPROCESSED IMAGE
EXTRACTED IMAGE FEATURE
[[0.2676463 0.14216244 0.07809388 ... 0.02910438 0.09825262 0.7065693]]
[[0.13947475 0.07724921 0.10063587 ... 0.2470858 0.07783262 0.47525665]]
PREPROCESSED TEXT / DESCRIPTION
['color', 'red']
PROCESS
> Background removal Backgrounds were removed so that the images only contained cropped items or objects
> Transform size Convert to matches the Resnet input to 224 x 224
> Cleaning text Lowercase, removing spaces, and eliminating symbols
> Tokenization Tokenized into smaller units
> Stop words Eliminating commonly used words
> Lemmatization & POS Tagging Transforms to singular forms
> Uses the prunned proposed ResNet-50 model to extract features from the images.
> Comparison with other extracted image feature using Euclidean Distance metric.
> Check if result within the threshold range
> Combine with Cosine Similarity result
> Comparison with other preprocessed texts using the Cosine Similarity metric.
> Check if result within the threshold range
> Combine with Euclidean Distance result
OUTPUT
PREPROCESSED IMAGE
PREPROCESSED TEXT / DESCRIPTION
['color', 'red']
EXTRACTED IMAGE FEATURE
[[0.2676463 0.14216244 0.07809388 ... 0.02910438 0.09825262 0.7065693]]
[[0.13947475 0.07724921 0.10063587 ... 0.2470858 0.07783262 0.47525665]]
threshold: < 0.215
0.010471 < 0.215? MATCHED
threshold: < 0.44
0.158114 < 0.44? MATCHED
Euclidean Distancae? YES
Cosine Similarity? YES
Return => MATCHED

Figure 2. The process of model methodology stage.

2.1. Dataset

This research selected data based on specific criteria, requiring various photos of the same object from different angles or backgrounds, as well as different photos of similar objects with varying characteristics. The dataset used for model training included photos of specific item type, with a focus on smartphones, cameras, glasses, and hats. Each item type contained various objects with different backgrounds or angles to enable the model to identify items regardless of variations, simplifying image matching. A sample of the dataset is shown in **Figure 3** to illustrate the criteria.

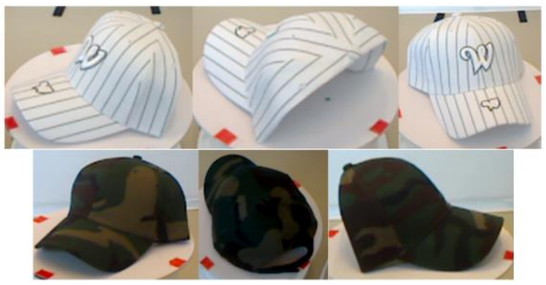


Figure 3. Photo of caps with various angles and backgrounds as a sample dataset.

In this study, the dataset combined the RGB-D Object Dataset[8], ABCI Dataset[9], and self-captured images. The proposed model is specifically targeted at certain items only and will be determined based on the search conducted on items that are most frequently lost[1,10,11]. The dataset consisted of four kinds of items:

handphones, cameras, sunglasses, and hats, each with five types, in total 800 photos for each type. To meet this number, some types of items were captured independently, and image augmentation (rotation, flip, brightness) was applied due to dataset limitations. Therefore, the final dataset consisted of 3400 photos.

2.2. Pre-processing data

This section discussed the pre-processing methods used to clean the image dataset so that it can be ready to be used in the further process.

2.2.1. Image pre-processing

Before using the dataset for model training, the dataset went through a pre-processing stage to meet the required criteria as input images. Furthermore, pre-processing is also applied to the input data of report submissions. First, images with backgrounds were removed so that the images only contained cropped items or objects. This was necessary because some data had a white background while others did not. Therefore, to standardize the data used for training, manual background removal was performed. Meanwhile, the removal of the background in user image inputs is carried out using a third-party tool called "removebg". Additionally, due to varying input image sizes, the images were resized to fit the input image size required by the ResNet method, which was 224 × 224. Example input and output of the pre-processing stage can be seen in **Figure 4**.

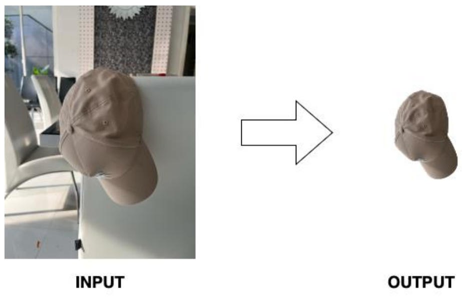


Figure 4. Before and after pre-processing.

2.2.2. Text pre-processing

To enhance the quality and precision of text-based models and algorithms, it's necessary to transform text into a suitable format as part of the preparation process for analysis because natural language is a form of communication with significant power. Natural language processing (NLP) is a computer science branch that aims to bridge human language and computers, transforming natural language into computer-understandable instructions. It draws knowledge from computer science, linguistics, and mathematics, evolving with improved algorithms and datasets for accurate text analysis[12]. NLP involves mining entities, extracting knowledge, language translation, summarization, inference, classification, and clustering[13]. In this research, NLP is employed for text pre-processing.

Before conducting the String-Matching process, the researcher performs text pre-processing to enhance effectiveness and efficiency. This process involves converting letters to lowercase, removing spaces, and eliminating symbols. The clean text is then tokenized into smaller units known as tokens, which serve as learning features. Stop words removal further reduces the text size by eliminating commonly used words from the Natural Language Toolkit (NLTK) library.

The researcher employs Lemmatization with WordNet Corpus for English word transformation, using the NLTK library's WordNetLemmatizer. This choice is preferred over Stemming due to potential word ambiguity. Lemmatization ensures accurate transformation from plural to singular forms without undesired fragments. It is combined with Part-of-Speech (POS) Tagging to determine word meanings in context, assigning tags like Verb, Noun, Adjective, or Adverb to avoid truncation and preserve precise meaning and context.

2.3. Model generalization

This section discussed the modelling methods used to determine the pattern in the data. The model generalization include the experiment process to identify the optimum parameter setting for the neural network model.

2.3.1. ResNet-50

To determine image similarity, it requires tasks such as image recognition and a solution capable of efficiently capturing spatial patterns and hierarchical features in visual data. Convolutional neural network (CNN) is a deep learning method that performs well in image recognition tasks[14]. CNN processes data with a grid pattern, such as images, which are designed to be adaptive. CNN automatically learns hierarchical spatial features through backpropagation using multiple building blocks[15]. The deep CNN image classification process can be generally divided into three stages or layers. The image passes through the convolutional layer, pooling layer, and fully connected layer[14].

ResNet, developed by He et al. in 2016[16], specifically designed to address challenges encountered in Deep Learning training, where training typically demands a significant amount of time and is constrained by a fixed number of layers. What sets the ResNet model apart is its capacity to sustain high performance even as the architecture grows more complex. Furthermore, ResNet excels in training networks more efficiently and cutting down on computational expenses. The intricacies introduced by ResNet are managed through the implementation of skip connections or shortcuts. In the ResNet model, connections are bypassed between two to three layers that include ReLU and batch normalization within the structure. ResNet is recognized as a skip connection because it enables the network to skip training through multiple layers and directly link to the output. According to He et al.[16], ResNet consistently delivers superior results in image classification when compared to other models, underscoring its proficiency in effectively extracting image features. The formula for ResNet can be seen in Equation (1).

$$y = F(x, W + x) \tag{1}$$

where x is the input layer, y is the output layer, and F represents the residual mapping. A residual block in ResNet is achieved when the input data dimension is the same as the output data dimension. Each ResNet block consists of two layers (for ResNet-18 and ResNet-34) or three layers (for ResNet-50 and ResNet-101)[17]. Utilizing ResNet as a pre-trained model for medical image classification has shown promising results[18].

ResNet-50 is a residual network with 50 layers. ResNet-50 demonstrates good performance in image classification, as it excels at extracting high-quality image features when working with ImageNet data. ResNet-50 can easily achieve high accuracy with increased depth[19].

The research utilizes TensorFlow and Keras, which are user-friendly machine learning tools known for their simplicity and extensive resources. TensorFlow offers a flexible ecosystem[20], while Keras prioritizes ease of use with a simple Application Programming Interface (API), clear error messages, and thorough documentation[21].

The first part of the proposed model is image comparison, which aims to compare the similarity between photos in a report and other report photos. Image comparison is performed using the CNN algorithm, specifically ResNet-50, to extract features from the images. In using the Keras API to create the basic ResNet-50 model, the researcher did not use the original fully connected layer from its architecture. There are 49 layers ($1 + 16 \times 3 = 49$ conv layers) with the output size of the 49th layer being 2048. Then, the fully connected layer was manually added to the final layer.

The researchers conducted experiments with a limited resource setting of 10 epochs to train the model. They determined the best parameters based on validation loss values. The final parameters chosen include a batch size of 10, dropout of 0.2, a Fully Connected Layer with 64 units and Relu activation. Lastly, a fully connected layer with parameters corresponding to the number of classes in the dataset is added. This layer serves as a classifier with Softmax activation, as it is used for multi-class classification. The architecture of the presented model can be seen in **Figure 5**.

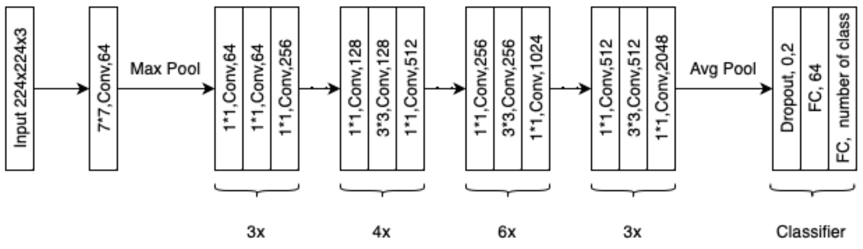


Figure 5. Architecture of the proposed ResNet-50 model.

The model will be trained using a pre-processed dataset with a setting of 100 epochs. The dataset is divided into 80% for training data and 20% for validation data. Then the model is saved in the h5 format.

The output of the model is then used to calculate image similarity. To achieve this, the model undergoes a pruning process where the fully connected layer is removed. This is done to ensure that the output of the model is the image extract.

2.3.2. Euclidean distance

In image search systems, similarity or difference between images is measured by calculating the distance between the query image and the feature vector of images in the database[22]. This metric is needed is to use mathematical distance metric that quantifies the difference between two images in a multi-dimensional space. In this study, Euclidean Distance is used as a measure of similarity between image feature vectors. The formula for Euclidean distance (E_D) can be seen in Equation (2).

$$E_D(Q,D) = \sqrt{\sum_{f=1}^{n} |\,[F_Q(f) - F_D(f)]^2|} \tag{2}$$

where 'n' represents the size of a feature vector. F_Q correspond to the feature vectors of a query image and F_D correspond to the image features in the database. The smaller the result of the Euclidean distance, the greater the similarity. Conversely, the larger the result of the Euclidean distance, the smaller the similarity[22,23].

The Euclidean distance metric is used in a research study as a measure of similarity in the quantitative analysis of Content-Based Image Retrieval (CBIR) usage[23]. In addition to Euclidean distance, the study explores metrics like city block distance, Canberra distance, and statistical distance. However, Euclidean distance is preferred in image search systems for its superior performance. Experimental results show significantly higher average precision with Euclidean distance, indicating its effectiveness in image retrieval. The proposed Fused Information Feature-based Image Retrieval System (FIF-IRS) method also exhibits a lower average error rate with Euclidean Distance compared to other methods across three databases.

To calculate image similarity, the image extracts from the model are compared. A threshold is needed to conclude that an image is similar to another image. The threshold determination is based on the measurement results by observing the highest value or maximum Euclidean distance from the experimental results that predict dozens of prepared item photos by the researchers. Since the highest value or maximum Euclidean distance is 0.0215, the image matching threshold has been determined to be smaller than 0.0215 based on the measurements. The calculated similarity results between photos within the threshold range are considered a match and are included in an array.

2.3.3. Cosine similarity

Text similarity is needed to measure of how similar or related two pieces of text are in terms of their content, meaning, or structure. What is needed is to use metric that measures the similarity between two pieces of text, typically represented as vectors in a high-dimensional space. Cosine similarity is a similarity function or metric assigns a numerical value to measure the similarity degree between two objects. Cosine similarity, a metric, assesses the similarity extent between two vectors by considering the cosine of the angle between them in a multidimensional space, focusing on vector orientation rather than magnitude. The cosine similarity value is calculated by determining the cosine of the angle between the vectors. When the angle is 0, the Cosine Similarity becomes 1, signifying perfect similarity, while any non-zero angle results in a value less than 1, indicating reduced similarity. Thus, a cosine similarity value of 1 implies high similarity. The formula for calculating cosine similarity is provided in Equation (3).

$$\mathrm{Cos}\alpha = \frac{Q \cdot D}{|Q||D|} = \frac{\sum_{i=1}^{n}(Q_i \times D_i)}{\sqrt{\sum_{i=1}^{n}(Q_i)^2} \times \sqrt{\sum_{i=1}^{n}(D_i)^2}} \tag{3}$$

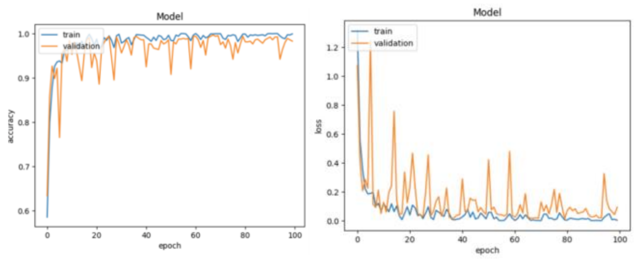
where Q and D is two vectors of attributes, $|Q|$ and $|D|$ is the Euclidean norm or length, $Q \bullet D$ is the dot product, ‘n’ is the quantities of vector, and ‘i’ is the total term in a sentence[2,24].

The second part of the proposed model is string matching to calculate the similarity between texts in the reports. String matching is performed on the text within the report, specifically, the description or characteristics of the item and the location where the item was found. Cosine similarity method was chosen because it has been proven to provide a high level of accuracy. The main advantage of the cosine similarity method is its insensitivity to the document length. Therefore, by comparing the generated keywords, the proximity level between documents can be determined[24].

Text similarity is determined using the cosine similarity method after text pre-processing. Various item descriptions with different patterns were measured to establish a threshold for string matching, which was set at 0.44 based on the lowest value from tests. Similarity results exceeding this threshold (greater than 0.44) are considered matches and included. A match is confirmed when there are similarities in type, description, time, and location between items. The final prediction combines string matching and image similarity, and a report is declared a match if both predict correctly.

2.3.4. Evaluation

Accuracy, a widely used evaluation metric, quantifies the proportion of correctly classified instances in a dataset. Accuracy is appropriate for many studies, especially when the aim is to assess the overall correctness of a model’s predictions. This work utilized accuracy as the evaluation metrics due to the size of dataset. After training the model with the dataset, the test data is used to measure the accuracy and loss of the trained model using ResNet-50 and the specified settings of 100 Epochs. It shows an average Val accuracy of 0.96 and Val loss of 0.14. The progress of accuracy and loss during model training is depicted in **Figure 6**.



(a) Accuracy value. **(b)** Loss value.

Figure 6. Model training process.

Next, validation accuracy and validation loss of the model trained with ResNet-50 is compared against other CNN algorithms, which can be seen in **Table 1**. This analysis aims to identify any superior performance among these methods. The results will help us determine the most promising approach.

Table 1. Comparison of model accuracy using other CNN algorithm.

Method	Val accuracy	Val loss
MobileNet v2	0.602	5.582
Inception v3	0.938	0.487
ResNet-50	0.964	0.14

Dummy data was generated and stored in a MySQL database, comprising details such as item types, descriptions or characteristics, the location and time of item discovery or loss, images of the items, information about the discoverer, and contact details. With the presence of dummy data, the matching process between reports could be conducted.

Furthermore, the researcher performs experiments on string matching of item descriptions by comparing different similarity metrics. The objective is to find the best method, enabling us to select the most promising approach for this study. Testing on String Matching begins with comparing the similarity values of each metric before and after text pre-processing. This will help in understanding the purpose behind text pre-processing. The test results can be seen in **Table 2**.

Table 2. The similarity value before and after using text pre-processing.

Tester	Scenario	Test result
Before	"the color is red" "red color"	0.707
	"the color is red and the stripes color are black" "red with black stripes"	0.452
After	"the color is red" "red color"	1.0
	"the color is red and the stripes color are black" "red with black stripes"	0.866

After seeing the results of testing before and after text pre-processing in string matching, significant differences can be concluded. This is because the text has not been cleaned from irrelevant elements, which removes the essence or core of the information. With text pre-processing, the text will be easier to understand and process to match only important information. So, it can be seen that the results after text pre-processing produce better values.

Furthermore, testing continues by looking at the similarity values between texts that contain the same meaning but have different writing styles. The aim of this analysis is to identify any superior performance among this method. The outcomes will assist in establishing the most promising approach. An example of text is the original description obtained by researchers from two different users. The way of describing an item by each user can be different even though the item is the same. The test results can be seen in **Table 3**.

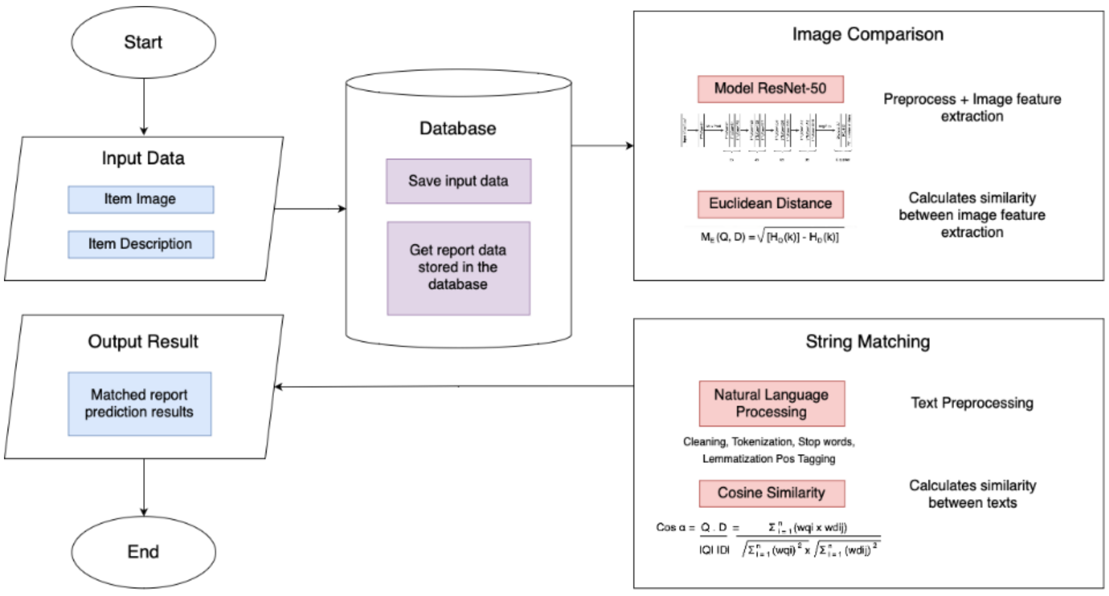
The results obtained from testing the similarity between texts that contain the same meaning but have different writing styles show a good value in predicting that both texts are the same. Cosine similarity shows a superior value compared to other metrics, which is 0.816 in the first scenario and 0.67 in the second scenario. With this testing, the model uses cosine similarity as a metric to calculate string matching.

Table 3. Experiment using multiple metrics to calculate text similarity.

String	Metric	Similarity
"the phone's color is gray, it has the apple logo behind it" "the color is gray, it has apple logo"	Cosine Similarity TF-IDF	0.709
	Cosine Similarity	0.816
	Jaccard Similarity	0.666
	Dice Coefficient	0.8
"the glasses has a silver frame, it has a round shaped" "round glass, silver color"	Cosine Similarity TF-IDF	0.51
	Cosine Similarity	0.67
	Jaccard Similarity	0.5
	Dice Coefficient	0.666

2.3.5. Prototype

The prototype's algorithm, can be seen in **Figure 7**, begins with the user uploading an item report containing a photo and description. This input is stored and compared with other reports in the database. The model predicts similarities using image comparison for photos and string matching for descriptions. The prediction results in a list of matching item reports, displayed in the prototype.



Start
Input Data
Item Image
Item Description
Database
Save input data
Get report data stored in the database
Image Comparison
Model ResNet-50
Preprocess + Image feature extraction
Euclidean Distance
Calculates similarity between image feature extraction
Output Result
Matched report prediction results
String Matching
Natural Language Processing
Text Preprocessing
Cleaning, Tokenization, Stop words, Lemmatization Pos Tagging
Cosine Similarity
Calculates similarity between texts
End

Figure 7. Overall flow of a lost and found application.

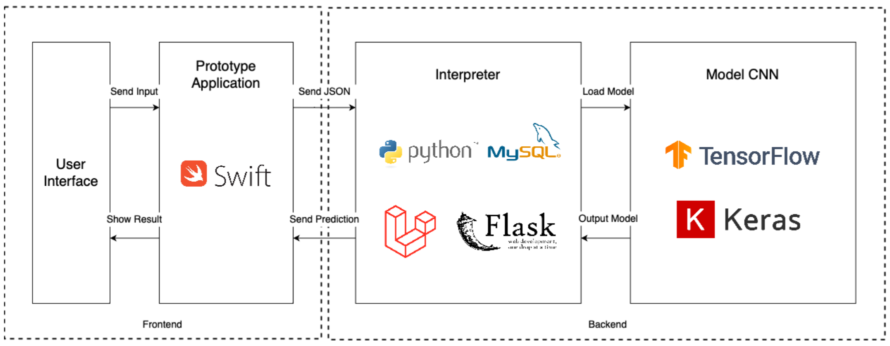


Figure 8. System application architecture.

In the system architecture, various algorithms, frameworks, and programming languages are utilized. The Swift programming language is used for the frontend of the prototype. The Laravel and Flask frameworks are employed as the system's framework for the prototype and to run the APIs. The MySQL database and Python programming language are utilized as the interpreter to execute the model. TensorFlow and Keras are employed for the creation and utilization of the ResNet-50 model. The system's flow or workflow can be visualized in the form of a diagram, as shown in **Figure 8**.

The prototype's design process utilized Figma as a design tool and includes multiple pages. Users can view saved reports and add new ones, providing information and photos of lost or found items. The matching process occurs upon completing the report. The iOS application was developed using XCode, Apple's integrated development environment. A preview of the prototype is shown in **Figure 9**.

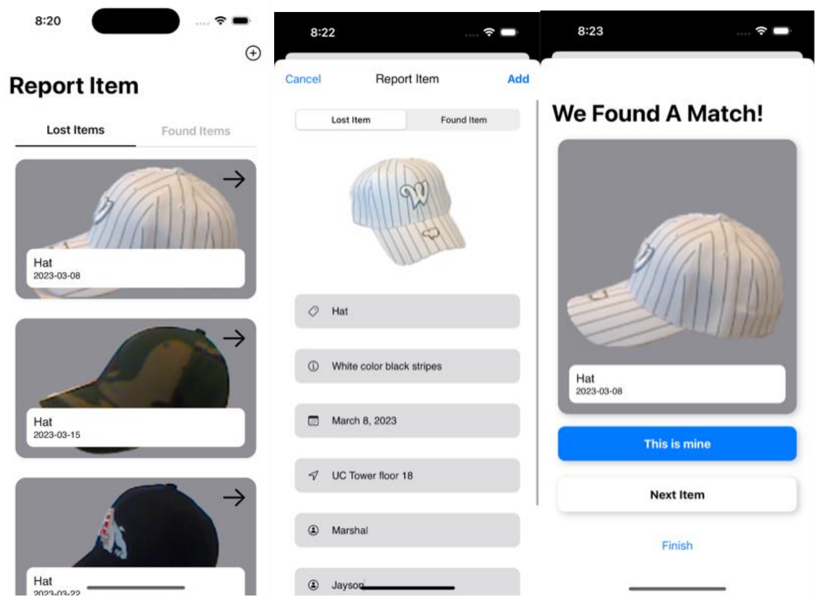


Figure 9. Preview of iOS-based prototype to run the model.

The user is asked to provide a description accompanied by a photo of the lost or found item to submit a new report. Following that, the application displays items that are predicted to match or resemble the one being searched for or reported. In the current state, users are restricted to using images from their mobile devices and predefined item types. If the image is replaced with an image of a different type of item, it may result in suboptimal predictions from the model. Users can view all previously submitted reports on the report list page, whether they are reports of lost or found items. To ensure the user privacy, the system only store a non-sensitive information that is used to identify the item not the person. Therefore, there is no phone number or personal identification number stored in the system. The system only save the name as the key to determine the ownership.

The anticipation of a potential decline in system performance correlating with an increase in the volume of lost item reports has led to the strategic decision to deploy the proposed system on a cloud platform. Using the cloud service technology, the performance aspects related to the effectiveness and accuracy in identifying and matching similar lost items can be maintained. The cloud service technology allows the system managers to perform load balancing based on the system usage. This strategy ensures the optimization of resources and the scalability of the system to handle fluctuating demands.

3. Results

This section discussed the result in applying the proposed methods. This section includes the discussion on model testing and prototype test.

3.1. Model test

Furthermore, testing is conducted on the predetermined threshold of the model based on the measurements of Euclidean distance and cosine similarity. Through this testing, the accuracy of correctly predicting similarity can be measured. The scenario in this testing involves calculating the similarity values of image comparison and string matching for various item reports using a threshold as a determinant of the prediction results. A value of 1 indicates a predicted match, while 0 indicates a predicted mismatch. The heatmap for the cosine similarity threshold testing can be seen in **Figure 10**.

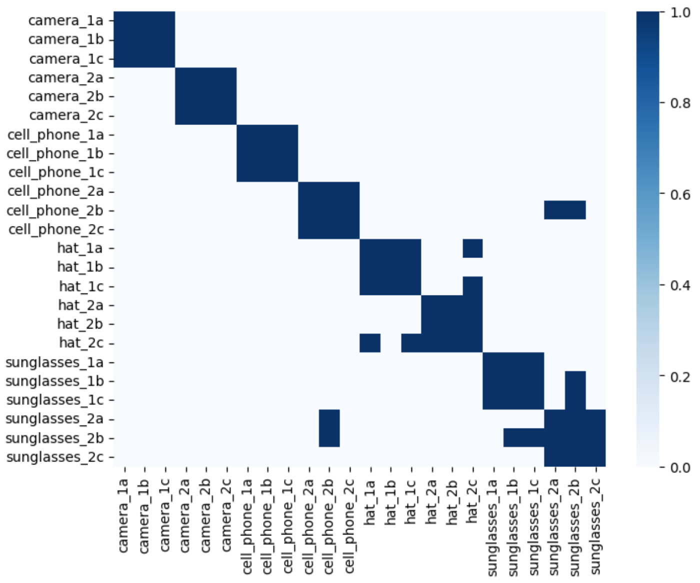


Figure 10. Results of cosine similarity threshold testing.

The accuracy measurement involves calculating the percentage of correct predictions in different scenarios. For predicting image matches, the accuracy was 29.96% with Euclidean distance threshold, while predicting text matches had 97.92% accuracy with cosine similarity threshold. The model’s responses indicate that both image and text must match for predictions to align, with specific criteria for each part. Cosine Similarity performed well, while Euclidean distance yielded less satisfactory results.

3.2. Prototype test

Testing on the prototype or application is carried out to ensure that the application operates in accordance with the specified specifications and requirements. The performance of the prototype will be measured by assessing the success of its functions and features as a whole without encountering any errors or bugs. The method used in testing is performance testing of the model. Scenarios in this testing begin by observing the model’s response to various reports with input variations provided by users. The various reports tested include: same image with the same description, same image with a different description, different image with the same description, and different image with a different description. The prediction for all reports only shows the same image with the same description, indicating a match result. During the testing process, potential users are also involved. This involves two prospective users submitting lost item reports, followed by feedback collection to ensure application functionality and gather user input. The testing results conducted by prospective users can be seen in **Tables 4** and **5**.

The test results for the functional prototype confirm its successful implementation and smooth performance, ensuring a positive user experience. The results demonstrate successful testing across multiple aspects of the system. Users are able to view a list of stored reports and access detailed information easily. The user interface provides a smooth and intuitive navigation experience, allowing users to switch between pages, menus, and features effortlessly. Creating new reports is successful, with accurate storage of entered information in the database. The image comparison and string matching prediction models run successfully, calculating the Euclidean distance and cosine similarity values effectively. Users can view combined prediction results accurately and appropriately, ensuring a comprehensive user experience.

Table 4. Model response testing.

Tester	Scenario	Expected result	Test result	Total score (%)
Tester A	There are reports with similar items that have already been submitted.	Found	Found	2/2 × 100 = 100%
	There are no reports with similar items that have already been submitted.	Not Found	Not Found	
Tester B	There are reports with similar items that have already been submitted.	Found	Found	2/2 × 100 = 100%
	There are no reports with similar items that have already been submitted.	Not Found	Not Found	

Table 5. User experience testing.

Tester	Question	Score	Total score (%)
Tester A	Can the report data be viewed properly?	5	28/30 × 100 = 93.33%
	Is the navigation functioning correctly?	5	
	Is the report addition process user-friendly?	3	
	Does the image comparison model predict accurately?	5	
	Does the string matching model predict accurately?	5	
	Are the prediction results displayed and visible?	5	
Tester B	Can the report data be viewed properly?	5	29/30 × 100 = 96.66%
	Is the navigation functioning correctly?	5	
	Is the report addition process user-friendly?	4	
	Does the image comparison model predict accurately?	5	
	Does the string matching model predict accurately?	5	
	Are the prediction results displayed and visible?	5	

4. Discussion

Based on the entire process that has been conducted, which focuses on the development of a predictive model for similarity in lost item reports, it can be concluded that the built model is capable of providing good and significant results in identifying the similarities between the reports. In conclusion, the use of the ResNet-50 algorithm with CNN is effective in handling the complexity of variations in lost item report photos. Using ResNet-50, the model can automatically and optimally extract important features for identifying image similarities. The Euclidean distance metric can calculate the similarity between the extracted images. To address the risk of two distinct items sharing a similar visual appearance, it is essential to include additional information in the form of descriptions or item details. This step aims to increase accuracy, and hence, the proposed model will integrate image matching with description matching. By using cosine similarity with text pre-processing, the model can effectively process text with different writing styles and calculate text similarity accurately.

It’s important to note that this model is not entirely without limitations. Despite the researcher’s efforts to address input variations with thorough processing and suitable datasets, there may still be cases where the predicted reports do not match expectations. It was found that the range of similarity values measured using Euclidean distance varies, leading to inaccuracies in determining the threshold. Currently, users are limited to utilizing images from their mobile devices and predefined item types. Going outside these limitations could lead to less-than-optimal predictions from the model. Scaling up and retraining the model is necessary to improve performance. Here are some recommendations to support or improve the development of the model for predicting the similarity of lost item reports in the future:

- Increase the collection of appropriate dataset and resources. This can be facilitated through the use of cloud technology.
- Periodically evaluate the model's performance and make improvements if errors are found.
- Enhance complexity by considering variations in input provided by users, including variations in language, structure, and format of user-generated lost item reports.
- Consider the possibility of time lapses between when an item is lost and found.
- Explore the use of deep learning methods or more advanced similarity metrics.
- Integrate the model into a security system within an institution or organization to streamline the reporting and searching for lost items.

By implementing these recommendations, it is expected that the model for predicting the similarity of lost item reports will experience improvements in effectiveness and efficiency in the reporting and search processes for lost items. Additionally, this model is expected to provide practical benefits to the community.

5. Conclusion

In summary, the developed predictive model for identifying similarities in lost item reports has demonstrated its capability. It utilized image pre-processing and CNN with the ResNet-50 algorithm for image extraction, as well as text pre-processing with Natural Language Processing for textual data. The model effectively handled photo complexities, extracted important features, and calculated similarity values using Euclidean distance and cosine similarity. The evaluation process shows that image comparison method provides an accuracy result of 29.96% correctness, while string matching method with 97.92% correctness.

However, there are limitations such as variations in similarity values, small dataset size, and specific item recognition. To address these, future improvements include expanding the dataset, periodic model evaluation, considering time intervals, accommodating user input variations, exploring advanced deep learning methods, and integration with existing security systems for streamlined lost item reporting and search processes.

Author contributions

Conceptualization, JP and TRDS; methodology, JP; validation, JP and TRDS; writing—original draft preparation, JP; writing—review and editing, TRDS; visualization, JP; supervision, TRDS; funding acquisition, TRDS. All authors have read and agreed to the published version of the manuscript.

Funding

We acknowledge the funding from the Ministry of Research, Techology, and Higher Eduction under the Research Funding Scheme between LLDKTI VII with Universitas Ciputra Surabaya on 19 June 2023 (contract number: 044/SP2H/PT/LL7/2023) which was obtained by Theresia Ratih Dewi Saputri to supporting the data collection, analysis process and publication.

Conflict of interest

The authors declare no conflict of interest.

References

1. Ahmad S, Ziaullah M, Rauniyar L, et al. How Does Matter Lost and Misplace Items Issue and Its Technological Solutions in 2015—A Review Study. *IOSR Journal of Business and Management* 2015; 17(4).
2. Suryani L, Edy K. Pengembangan aplikasi "lost & found" berbasis android dengan menggunakan metode term frequency—Inverse document frequency (TF-IDF) dan cosine similarity. *Jelekn* 2020; 6(2): 190-204. doi:

10.32531/jelekn.v6i2.232
3. Setiawan MA, Andryana S, Gunaryati A. Penerapan Algoritma Boyer Moore Dalam Pencarian Barang Hilang pada Aplikasi FindIt Berbasis Android. *Jurnal Media Informatika Budidarma* 2021; 5(3): 945. doi: 10.30865/mib.v5i3.3093
4. Ghazal M, Al Khalil Y, Haneefa F, et al. Archival and Retrieval of Lost Objects using Multi-feature Image Matching in Mobile Applications. *International Journal of Computing and Digital Systems* 2016; 5(1): 73-83. doi: 10.12785/ijcds/050107
5. Liu Y, Fang K, Yang E, et al. Lost-Found Item Net for Classification Based on Inception-Resnet. In: Proceedings of 2022 16th IEEE International Conference on Signal Processing (ICSP); 2022. doi: 10.1109/icsp56322.2022.9965294
6. Zhou M, Fung I, Yang L, et al. Lostnet: A Smart Way for Lost and Find. *SSRN Journal* 2023. doi: 10.2139/ssrn.4391555
7. Jeong H, Yoo H, You T, et al. Lost and Found Registration and Inquiry Management System for User-dependent Interface using Automatic Image Classification and Ranking System based on Deep Learning. *Journal of convergence security* 2018; 18(4): 19-25.
8. RGB-D Object Dataset. Available online: https://datasets.abci.ai/dataset/miro/ (accessed on 17 June 2022).
9. MIRO: Multi-view Images of Rotated Objects. Available online: http://rgbd-dataset.cs.washington.edu (accessed on 17 June 2022).
10. detikTravel. 10 Barang yang Paling Sering Hilang Saat Liburan. Available online: https://travel.detik.com/travel-news/d-2267276/10-barang-yang-paling-sering-hilang-saat-liburan (accessed on 26 February 2023).
11. Chipolo. Lost and Missing Items: What We Lose the Most and Where We Lose It. Available online: https://chipolo.net/en/blogs/lost-and-missing-items-what-we-lose-the-most-and-where-we-lose-it (accessed on 26 February 2023).
12. Kang Y, Cai Z, Tan CW, et al. Natural language processing (NLP) in management research: A literature review. *Journal of Management Analytics* 2020; 7(2): 139-172. doi: 10.1080/23270012.2020.1756939
13. Hussen Maulud D, Zeebaree SRM, Jacksi K, et al. State of Art for Semantic Analysis of Natural Language Processing. *Qubahan Academic Journal* 2021; 1(2): 21-28. doi: 10.48161/qaj.v1n2a44
14. Srinivasan K, Garg L, Datta D, et al. Performance Comparison of Deep CNN Models for Detecting Driver's Distraction. *Computers, Materials & Continua* 2021; 68(3): 4109-4124. doi: 10.32604/cmc.2021.016736
15. Yamashita R, Nishio M, Do RKG, Togashi K. Convolutional neural networks: An overview and application in radiology. *Insights Imaging* 2018; 9(4): 611-629. doi: 10.1007/s13244-018-0639-9
16. He K, Zhang X, Ren S, Sun J. Identity mappings in deep residual networks. In: Proceedings of the Computer Vision–ECCV 2016: 14th European Conference; 11–14 October 2016; Amsterdam, The Netherlands. pp. 6302–645.
17. Sarwinda D, Paradisa RH, Bustamam A, Anggia P. Deep Learning in Image Classification using Residual Network (ResNet) Variants for Detection of Colorectal Cancer. *Procedia Computer Science* 2021; 179: 423-431. doi: 10.1016/j.procs.2021.01.025
18. Reddy ASB, Juliet DS. Transfer Learning with ResNet-50 for Malaria Cell-Image Classification. In: Proceedings of 2019 International Conference on Communication and Signal Processing (ICCSP); 2019. doi: 10.1109/iccsp.2019.8697909
19. Wen L, Li X, Gao L. A transfer convolutional neural network for fault diagnosis based on ResNet-50. *Neural Comput & Applic* 2019; 32(10): 6111-6124. doi: 10.1007/s00521-019-04097-w
20. Tensor Flow. Available online: https://www.tensorflow.org/ (accessed on 17 June 2022).
21. Keras. Available online: https://keras.io/ (accessed on 17 June 2022).
22. Zenggang X, Zhiwen T, Xiaowen C, et al. Research on Image Retrieval Algorithm Based on Combination of Color and Shape Features. *J Sign Process Syst* 2019; 93(2-3): 139-146. doi: 10.1007/s11265-019-01508-y
23. Thusnavis Bella MI, Vasuki A. An efficient image retrieval framework using fused information feature. *Computers & Electrical Engineering* 2019; 75: 46-60. doi: 10.1016/j.compeleceng.2019.01.022
24. Sitikhu P, Pahi K, Thapa P, Shakya S. A Comparison of Semantic Similarity Methods for Maximum Human Interpretability. In: Proceedings of 2019 Artificial Intelligence for Transforming Business and Society (AITB); 2019. doi: 10.1109/aitb48515.2019.8947433