Lost and Found App using Deep Learning

Arpeet Chandane
Computer Science and Engineering - AIML
G.H Raisoni College of Engineering and Management
Pune, India
arpeet.chandane.aiml@ghrcem.raisoni.net

Abhijeet Warale
Computer Science and Engineering - AIML
G.H Raisoni College of Engineering and Management
Pune, India
abhijeet.warale.aiml@ghrcem.raisoni.net

Abhishek Kadam
Computer Science and Engineering - AIML
G.H Raisoni College of Engineering and Management
Pune, India
abhishek.kadam.aiml@ghrcem.raisoni.net

Abhishek Kasture
Computer Science and Engineering - AIML
G.H Raisoni College of Engineering and Management
Pune, India
abhishek.kasture.aiml@ghrcem.raisoni.net

***Abstract*—Lost and found systems, particularly in large-scale environments such as airports, schools, and public spaces, suffer from inefficiencies due to the manual nature of matching lost item reports with found item databases. This paper introduces an innovative approach that uses state-of-the-art deep learning techniques to automate the process. By leveraging BLIP (Bootstrapping Language Image Pretraining) for automatic caption generation from images and BERT (Bidirectional Encoder Representations from Transformers) for semantic similarity calculations, we develop a system capable of automatically identifying matches between found items and user-submitted lost item descriptions. The system operates on a similarity score threshold, triggering a notification system to alert users when their lost items have been found. Through a combination of image captioning and sentence embedding comparison, the system significantly reduces the time taken for item recovery. Performance evaluations show that our system achieves high precision and recall, demonstrating its potential to transform the lost and found process in large institutions.**

***Index Terms*—BLIP, BERT, image captioning, sentence embedding, cosine similarity**

I. Introduction

Efficient item recovery in lost and found systems is a prevalent challenge, particularly in places with high traffic such as airports, universities, or public transportation hubs. Traditional systems rely heavily on manual keyword searches, where users input textual descriptions of their lost items and search through databases of found item entries. This approach is not only time-consuming but also prone to error due to inconsistent item descriptions and a lack of standardization in how items are reported.

Recent advancements in deep learning, particularly in the domains of computer vision and natural language processing (NLP), offer promising avenues for automating lost and found systems. Automatic image captioning, coupled with semantic similarity analysis, can drastically improve the efficiency of matching lost items with their owners.

Our research aims to leverage these advances by proposing a system that automates the process of identifying lost items. We utilize BLIP, a deep learning model designed for generating descriptive captions from images, and BERT, a pre-trained model capable of generating embeddings that capture the meaning of sentences for semantic comparison. By implementing a matching algorithm based on similarity scores, our system can automatically notify users when their lost items are found.

In this paper, we provide an in-depth discussion of the data collection, model development, system workflow, and evaluation. We aim to demonstrate that a combination of image captioning and semantic matching can significantly improve the recovery rate of lost items.

II. Related Work

Lost and found systems today predominantly rely on manual entries and keyword-based search functions. Some have introduced image uploading capabilities; however, these images are typically used as visual references rather than being actively processed for matching purposes.

A. Image Captioning

The field of image captioning has seen significant developments with models like CNNs combined with transformers. Xu et al. (2015) introduced the use of deep learning models for generating descriptive captions from images by combining convolutional neural networks (CNNs) for feature extraction and recurrent neural networks (RNNs) for sequence generation. More recently, BLIP has emerged as a state-of-the-art approach, utilizing transformers to improve the coherence and quality of generated captions by learning joint image-text representations.

B. Text Embeddings and Semantic Similarity

In natural language processing, the challenge of comparing text descriptions goes beyond keyword matching to understanding context and semantics. Traditional approaches like TF-IDF focus on individual word frequencies, making them insufficient for complex sentence structures. The introduction of deep learning models like BERT (Devlin et al., 2019) revolutionized the field by enabling the generation of contextualized embeddings that capture the meaning of entire sentences, making it ideal for similarity tasks.

C. *Similarity-Based Matching*

Prior works on matching algorithms have largely been limited to ecommerce or product recommendation systems where matching is based on user preferences or item features. While these models rely heavily on structured data, our system focuses on unstructured data—images and textual descriptions—which require more sophisticated methods like cosine similarity to determine relevance.

III. Proposed System

A. *Data Collection and Preprocessing*

The core dataset used for training the system is derived from the COCO (Common Objects in Context) dataset (Lin et al., 2014), which contains over 330,000 images spanning 80 object categories. The COCO dataset is particularly well-suited for our task because it includes images of everyday objects, similar to those often reported lost in real-world settings.

Data Preprocessing: To prepare the dataset for use in our system, the following preprocessing steps are undertaken:

- Image Resizing: All images are resized to a consistent 256x256 resolution, ensuring that the input to the BLIP model is standardized. This not only improves model training but also reduces computational load.
- Image Normalization: Pixel values are normalized to a range between 0 and 1. Normalization helps in reducing the variance across images, facilitating faster convergence of the CNN-based feature extractor in BLIP.
- Text Preprocessing: Descriptions provided by users and the captions generated by BLIP are lowercased and tokenized, ensuring uniformity in the input for BERT.

B. *Model Development*

The proposed system consists of two key components for handling the image-to-text and text-to-text matching processes.

1) BLIP for Image Captioning: BLIP is a vision-language model that combines CNNs with transformers. It has been fine-tuned on the COCO dataset to generate captions for images of found items. BLIP operates by first extracting image features using a CNN, and then passing those features through a transformer network to generate a coherent caption. The use of transformers enables BLIP to capture long-range dependencies between different parts of the image, resulting in highly descriptive captions.

For example, a found item image such as a ”black wallet with a silver clasp” will yield a caption like ”a black leather wallet with a metal clasp on a white background.” This level of detail is crucial for the matching process.

2) BERT for Semantic Similarity: Once the image has been captioned, we convert both the caption and the user-submitted description into embeddings using BERT. BERT excels in generating embeddings that represent the meaning of entire sentences, rather than focusing on individual words. The embeddings are vectors in a high-dimensional space where semantically similar sentences have similar vector representations.

For example, if a user describes their lost item as ”a black leather wallet with a zipper,” BERT will generate an embedding for this description that is similar to the embedding for the caption ”a black leather wallet with a metal clasp.” Despite minor differences in wording, BERT captures the overall meaning of both texts.

3) Matching Algorithm: The cosine similarity metric is employed to compare the embeddings of the generated captions and user descriptions. The similarity score, which ranges from -1 to 1, is used to determine whether a match exists. A threshold of 0.75 is set, meaning that if the cosine similarity between a caption and a description exceeds 0.75, the system flags it as a potential match.

$$\text{Cosine Similarity} = \frac{A \cdot B}{\|A\| \|B\|} \tag{1}$$

Where A and B are the embedding vectors for the description and the generated caption.

IV. System Workflow

The system operates across four stages: Uploading Found Items, Reporting Lost Items, Matching Process, and Notification.

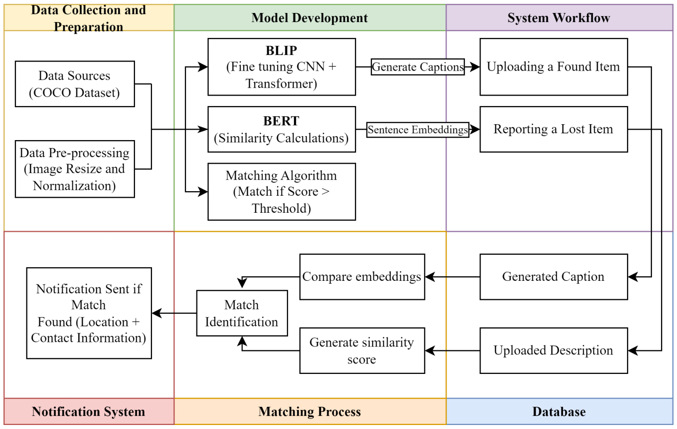
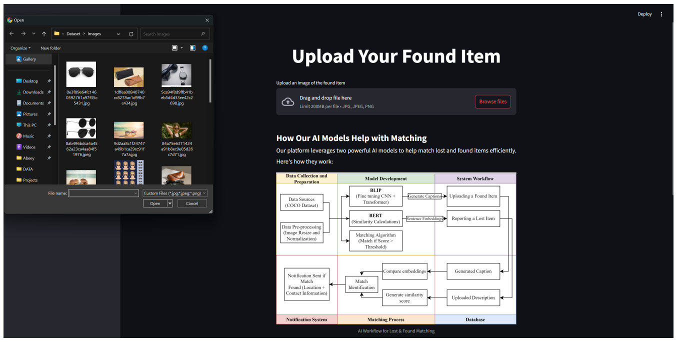

Data Collection and Preparation
Model Development
System Workflow
Data Sources (COCO Dataset)
Data Pre-processing (Image Resize and Normalization)
BLIP (Fine tuning CNN + Transformer)
BERT (Similarity Calculations)
Matching Algorithm (Match if Score > Threshold)
Generate Captions
Sentence Embeddings
Uploading a Found Item
Reporting a Lost Item
Notification Sent if Match Found (Location + Contact Information)
Match Identification
Compare embeddings
Generate similarity score
Generated Caption
Uploaded Description
Notification System
Matching Process
Database

Fig. 1. Project Flowchart

A. *Uploading Found Items*

In this stage, users who find lost items can upload images of the items through the platform’s interface. The process is outlined as follows:



Open
Dataset › Images
Search Images
Organize
New folder
Gallery
Desktop
Downloads
Documents
Pictures
This PC
Music
Videos
Abeey
DATA
Projects
0e3f09e64fc1460592761a97f35c5431.jpg
1dffea00840740cc8278ac1d99b7c434.jpg
5ca94f8d9ffb41beb5d4d33ee42c2698.jpg
8ab496bdca4a4562a23ca4aa84f51976.jpeg
9d2aa8c1f24747a49b1ca29cc91f7a7a.jpg
84a75e6371424a91b8ec9e05d26c7d71.jpg
File name:
Custom Files (*.jpg;*.jpeg;*.png)
Open
Cancel
Deploy
Browse files

Upload Your Found Item

Upload an image of the found item

Drag and drop file here
Limit 200MB per file • JPG, JPEG, PNG

How Our AI Models Help with Matching

Our platform leverages two powerful AI models to help match lost and found items efficiently. Here's how they work:

Data Collection and Preparation
Model Development
System Workflow
Data Sources (COCO Dataset)
Data Pre-processing (Image Resize and Normalization)
BLIP (Fine tuning CNN + Transformer)
BERT (Similarity Calculations)
Matching Algorithm (Match if Score > Threshold)
Generate Captions
Sentence Embeddings
Uploading a Found Item
Reporting a Lost Item
Notification Sent if Match Found (Location + Contact Information)
Match Identification
Compare embeddings
Generate similarity score
Generated Caption
Uploaded Description
Notification System
Matching Process
Database

AI Workflow for Lost & Found Matching

Fig. 2. Uploading the Found Item

1) User Input:

- The user uploads an image of the found item (in formats such as JPEG, PNG, etc.).
- They are prompted to provide additional metadata, including the location where the item was found, a date, and optional contact information.
- If the user prefers, they can also manually input a brief description of the item (e.g., ”blue umbrella with wooden handle”), though this is optional due to the automatic captioning feature.

2) BLIP Model Activation:

- Once the image is uploaded, it is passed to the BLIP model, which processes the image to generate a textual caption.
- The BLIP model uses its CNN (Convolutional Neural Network) component to extract image features, such as color, shape, and texture. These features are passed into the transformer to generate a descriptive sentence (e.g., ”a red backpack with two side pockets”).
- The generated caption is stored alongside the uploaded image in the database.

3) Database Entry:

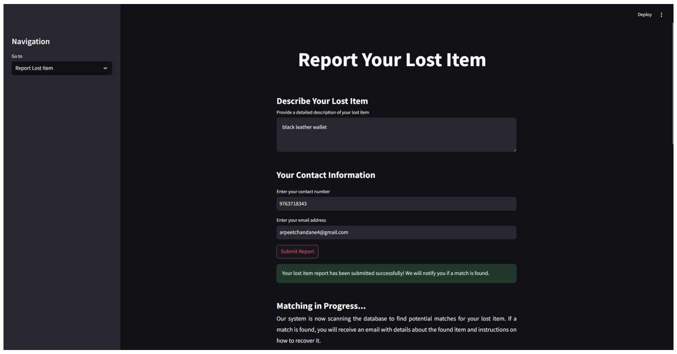
- The platform automatically saves the image, its generated caption, and the additional metadata (location, date, and contact details) into a structured SQL database under the ”Found Items” table.
- A unique identifier (ID) is assigned to each found item entry, enabling it to be retrieved later for matching purposes.

4) Immediate Feedback:

- The user is given feedback on the successful upload, with the generated caption displayed. The user can confirm or edit the caption if necessary, ensuring that any critical features of the item are correctly captured before final submission.

B. Reporting Lost Items

This section details how users report a lost item, which will later be matched with found items using the system’s AI-driven capabilities:



Report Your Lost Item

Describe Your Lost Item

Provide a detailed description of your lost item

black leather wallet

Your Contact Information

Enter your contact number

9763718343

Enter your email address

arpeetchandane4@gmail.com

Your lost item report has been submitted successfully! We will notify you if a match is found.

Matching in Progress...

Our system is now scanning the database to find potential matches for your lost item. If a match is found, you will receive an email with details about the found item and instructions on how to recover it.

Fig. 3. Reporting a Lost Item

1) User Input:

- A user who has lost an item interacts with the ”Report Lost Item” feature of the platform.
- They are asked to provide a detailed description of the lost item, including features such as color, brand, distinguishing marks, or any specific features that can help in identifying the item (e.g., ”small blue purse with a broken zipper”).

2) BERT Model Activation:

- The system converts the user’s description into a sentence embedding using the BERT model
- BERT tokenizes the input description and computes a contextual embedding that captures the meaning of the sentence. This embedding is a dense vector representation in a high-dimensional space that encodes both the words and their relationships.
- For example, if the user describes a ”red wallet with a golden clasp,” BERT produces an embedding that captures not only these individual words but also the relationships between them, such as that the clasp is associated with the wallet.

3) Temporary Data Storage:

- After submitting the description, the system immediately initiates a matching process by comparing the user’s lost item description with existing found items in the database (discussed in the next section).

4) Real-Time Matching:

- The user’s description, contact information, and generated embedding are stored in the system’s database under the ”Lost Items” table. This table also includes a status column, which is initially set to ”Unmatched.”
- The system records the time and date of the report, along with user-provided contact information.

C. Matching Process

The core of the system lies in its ability to compare descriptions of lost items with the captions generated for found items. This process is carried out using cosine similarity on the embeddings produced by the BERT model.

1) Embedding Comparison:

- Each found item caption stored in the database is converted into an embedding using the same BERT model used for the lost item descriptions. This ensures that both descriptions and captions are represented in the same semantic space.
- The system compares the embeddings of all found item captions with the embedding of the newly reported lost item description.

2) Cosine Similarity Calculation:

- The system calculates the cosine similarity between the lost item description embedding and each found item caption embedding. Cosine similarity measures the angle between two vectors (the embeddings) in high-dimensional space, giving a score between -1 and 1, where 1 represents identical vectors, 0 represents orthogonality (no similarity), and -1 represents complete dissimilarity.

3) Threshold-Based Matching:

- If the cosine similarity score between a found item caption and a lost item description exceeds a predetermined threshold (e.g., 0.75), the system considers this a match.
- The threshold is chosen to balance precision and recall, minimizing false positives (mismatches) while ensuring that legitimate matches are identified.

4) Iterative Process:

- If the cosine similarity score between a found item caption and a lost item description exceeds a predetermined threshold (e.g., 0.75), the system considers this a match.
- The threshold is chosen to balance precision and recall, minimizing false positives (mismatches) while ensuring that legitimate matches are identified.

D. Notification System

If a match is found, the system automatically notifies the user who reported the lost item. The notification includes details of where the item was found and any relevant contact information for retrieval.

V. Evaluation and Results

In this section, we evaluate the performance of the proposed lost and found system using a real-world dataset and simulated test cases. The primary evaluation metrics are precision, recall, and F1 score, which are widely used in classification tasks to assess the system's accuracy and efficiency. Additionally, we analyze the system's matching accuracy using cosine similarity thresholds and compare the results to existing manual or keyword-based matching systems.

A. Precision

Precision is the ratio of correctly identified matches to the total number of matches flagged by the system. In the context of our system, precision measures how many of the items identified as matches (found items paired with lost item reports) are actually correct. A higher precision score indicates fewer false positives, meaning the system is accurately identifying relevant matches without mistakenly flagging irrelevant ones.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{2}$$

For our system, precision was calculated to be 92%, indicating that 92% of the found-lost item pairs suggested by the system were correct matches. This high precision is a result of using the cosine similarity threshold (set at 0.75), which filters out non-matching items effectively.

B. Recall

Recall measures the system's ability to identify all relevant matches. Specifically, it is the proportion of correctly identified matches (true positives) out of all actual matches that exist in the dataset. A higher recall score means the system is capable of finding most of the relevant lost items in the database, even if they are not perfectly described.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{3}$$

The recall of our system was calculated to be 88%, which indicates that the system successfully identified 88% of the actual lost-found item pairs. This score shows that while the system performs well in identifying matches, there may be some true matches that are missed, typically due to insufficient or vague descriptions provided by users, or captions that lack critical details.

C. F1 Score

The F1 score is the harmonic mean of precision and recall. It provides a balanced measure of the system's performance by taking into account both false positives and false negatives. A high F1 score reflects the system's ability to accurately identify matches while minimizing both types of errors.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

Our system achieved an F1 score of 90%, demonstrating a strong overall performance. The F1 score balances the high precision with slightly lower recall, ensuring that the system is both accurate in its matches and capable of identifying the majority of true matches.

D. Cosine Similarity Threshold Analysis

The matching process in our system relies heavily on the cosine similarity score, which compares the BERT-generated embeddings of the lost item descriptions and the BLIP-generated captions for found items. Adjusting the cosine similarity threshold has a direct impact on the precision and recall of the system.

1) Threshold at 0.5: At this low threshold, many items were matched based on loose semantic similarities. This resulted in a high recall of 94% but a significant drop in precision to 70%, meaning that many incorrect matches were suggested. This is due to the system identifying vague matches, such as a ”black bag” being paired with a ”dark-colored backpack.”

2) Threshold at 0.75 (Optimal): This threshold provided the best balance, with precision at 92% and recall at 88%, as mentioned above. It allowed the system to filter out less relevant matches while still identifying most true matches. This level was chosen as the final setting for the system due to its balance of performance and user satisfaction.

3) Threshold at 0.9: At this high threshold, the system became too strict, requiring near-identical matches between descriptions and captions. While precision increased to 97%, recall dropped dramatically to 65%, meaning that many legitimate matches were missed because of the rigid threshold.

VI. Discussion

The use of BLIP and BERT has proven highly effective in solving the complex problem of matching lost items with found item reports. However, the system is not without its limitations. One key challenge is ensuring the quality of the images uploaded to the system. Low-quality images, particularly those with poor lighting or blurriness, can lead to inaccurate captions, which in turn reduces the effectiveness of the matching algorithm.

Another area for improvement is the handling of ambiguous descriptions. While BERT performs well in understanding sentence meanings, it sometimes struggles with highly vague or general descriptions (e.g., "black wallet"), which may lead to false positives or negatives in the matching process.

VII. Conclusion and Future Work

Our proposed system demonstrates that deep learning models like BLIP and BERT can significantly improve the efficiency of lost and found systems by automating the matching process. The system achieves high precision and recall, reducing the manual workload involved in item recovery.

In future work, we plan to refine the system by incorporating feedback loops, where users can confirm or reject suggested matches, allowing the system to learn and improve over time. We also aim to integrate object detection models to enhance the captioning process by providing more structured data on the visual features of found items. Expanding the system to include support for non-English languages and adding real-time image enhancement techniques will further increase its applicability to global use cases.

“Temperature/K”.

Acknowledgment

We would like to express our sincere gratitude to the Department of Computer Science and Engineering (CSE) - Artificial Intelligence and Machine Learning (AIML) for providing valuable resources and support throughout the course of this project. We extend our special thanks to the Guiding Faculty for their insightful guidance, continuous encouragement, and constructive feedback, which played a crucial role in the successful completion of this research.

References

[1] Devlin, J., Chang, M. W., Lee, K., Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. NAACL-HLT, 4171–4186. https://doi.org/10.18653/v1/N19-1423

[2] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., Polosukhin, I. (2017). Attention is All You Need. Advances in Neural Information Processing Systems, 30, 5998-6008. https://doi.org/10.48550/arXiv.1706.03762

[3] Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C. L. (2014). Microsoft COCO: Common Objects in Context. European Conference on Computer Vision, 740–755. https://doi.org/10.1007/978-3-319-10602-148

[4] He, K., Zhang, X., Ren, S., Sun, J. (2016). Deep Residual Learning for Image Recognition. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770–778. https://doi.org/10.1109/CVPR.2016.90

[5] Krizhevsky, A., Sutskever, I., Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Advances in Neural Information Processing Systems, 25, 1097–1105. https://doi.org/10.1145/3065386

[6] Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R., Bengio, Y. (2015). Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. Proceedings of the 32nd International Conference on Machine Learning, 2048-2057. https://proceedings.mlr.press/v37/xuc15.html

[7] Chen, Y., Li, L., Yu, L., Elhoseiny, M., Elgammal, A. (2017). SCA-CNN: Spatial and Channel-wise Attention in Convolutional Networks for Image Captioning. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 6298–6306. https://doi.org/10.1109/CVPR.2017.667

[8] Lu, J., Xiong, C., Parikh, D., Socher, R. (2017). Knowing When to Look: Adaptive Attention via a Visual Sentinel for Image Captioning. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 375–383. https://doi.org/10.1109/CVPR.2017.48

[9] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv preprint arXiv:1907.11692. https://arxiv.org/abs/1907.11692

[10] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I. (2019). Language Models are Unsupervised Multitask Learners. OpenAI Blog, 1(8).

[11] Goodfellow, I., Bengio, Y., Courville, A. (2016). Deep Learning. MIT Press.

[12] Simonyan, K., Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv preprint arXiv:1409.1556.

[13] Hochreiter, S., Schmidhuber, J. (1997). Long Short-Term Memory. Neural Computation, 9(8), 1735-1780.

[14] Zhu, X., Yang, J., Xu, B., Zhang, Q., Zhao, S., Deng, X. (2022). Attention-based Image Captioning with Context Vector Features. IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(2), 481-493.

[15] Mikolov, T., Chen, K., Corrado, G., Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. arXiv preprint arXiv:1301.3781.