

Efficient Scalable Template-Matching Technique for Ancient Brahmi Script Image

Sandeep Kaur* and Bharat Bhushan Sagar

Department of CSE, Birla Institute of Technology, Mesra, Ranchi, 835215, India

*Corresponding Author: Sandeep Kaur. Email: sandeepsandhu03@gmail.com

Received: 31 May 2022; Accepted: 05 July 2022

Abstract: Analysis and recognition of ancient scripts is a challenging task as these scripts are inscribed on pillars, stones, or leaves. Optical recognition systems can help in preserving, sharing, and accelerate the study of the ancient scripts, but lack of standard dataset for such scripts is a major constraint. Although many scholars and researchers have captured and uploaded inscription images on various websites, manual searching, downloading and extraction of these images is tedious and error prone. Web search queries return a vast number of irrelevant results, and manually extracting images for a specific script is not scalable. This paper proposes a novel multistage system to identify the specific set of script images from a large set of images downloaded from web sources. The proposed system combines the two most important pattern matching techniques-Scale Invariant Feature Transform (SIFT) and Template matching, in a sequential pipeline, and by using the key strengths of each technique, the system can discard irrelevant images while retaining a specific type of images.

Keywords: Brahmi script; SIFT (scale-invariant feature transform); multi-scale template matching; web scraping

1 Introduction

Across diverse cultures and civilizations, writing is the most important key tool for recording historical information. In today's time, paper is the most prominent mode of writing. However, before the creation of paper, content was mostly written on rocks, pillars, and slabs called inscriptions. These inscriptions provide an insight into the socio-economic history of the time and are also a reliable source of historical knowledge [1]. Around the era of Ashoka, the Kharosthi script and the Brahmi script were in use [2]. The Brahmi script belonging to the Indo-Aryan period is one of the earliest writing systems. This script is considered a root script which led to the development of many modern Indian and Asian scripts [3]. A multitask learning framework that converts ancient Brahmi characters to modern-day Tamil digital format by employing features based on shape and writing pattern has been proposed by authors [4]. Enhancement of the study and analysis of ancient scripts can be done by developing an optical recognition system for these scripts. Development of OCR systems for native languages like



This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Hindi and Telugu is gaining momentum [5], but there is still a significant need for the development of OCR for ancient scripts. Such a system requires a collection of datasets that includes digital text or images of the specific scripts, which can be used to develop a recognition system [6]. Collecting data for modern or present-day scripts is not a complex task as books, and written text images are readily available. However, ancient script inscriptions are available only on rocks, pillars, and leaves, which makes data collection a tedious task. With the wide growth of web and image capturing devices there is significant effort spent by researchers, historians, and scholars in creating digitized camera images of the inscription and sharing or storing these on web. If we consider Brahmi, inscription images from various historical sites are available on web, which can be helpful in training an efficient and accurate Brahmi recognition system.

Standard datasets are extremely useful in evaluating existing image processing algorithms and techniques and in the research and development of new systems [7]. But there is a lack of a standard dataset availability for Brahmi Script. Web sources and camera-captured images can be used to create a local dataset as commonly available datasets may not be public or cannot be directly used due to issues like lack of useful labels [8]. Creation of a standard dataset from inscription images has been explored by creating Bag-of-Visual words feature-set for an inscription image using SIFT and K-mean clustering, but this requires collecting each inscription image from multiple sources and then processing using the proposed technique [9,10]. As inscription images of Brahmi are available on web, search followed by download of these can be done using some form of web scraping tools (Web scraping or Web data extraction is a process of going through web pages and extracting documents, selected text, or images) [11,12]. But this process can return many non-relevant images and manual annotation to identify a specific type of image from the image metadata is a tedious and time-consuming process. Content-based image analysis leveraging features such as shape, texture, color, spatial layout, intensity, edges, etc. has been developed to overcome this drawback. Comprehensive survey on different content-based image retrieval approaches and image representation using features ranging from low level to high level has been discussed by various authors [13,14]. Method of identifying a specific set of vehicle images from a vast set with the images having different camera views has been purposed by authors, combining the global features from an image and enhanced local features [15]. However, Image retrieval systems for ancient languages are limited, and most of these retrieve a specific type of images from an existing dataset [16–18]. Further considering Brahmi there is no significant prior research on the Brahmi Image Retrieval system from an existing dataset or internet. The scraping of images through web searching poses another challenge that all the images returned from a search query may not be relevant to the requirement or may be of an entirely different context. The system proposed in this paper aims to identify images relevant to Brahmi Script from the large number of images obtained from web search queries. Images that have Brahmi characters in them are considered relevant images. We have used a sequential fusion of feature descriptor, with the first stage using scale-invariant feature transform (SIFT) to identify documents having inscription images or printed characters. This step filters out the documents having no relevance to inscriptions and printed characters. Multi-scale Template matching using individual Brahmi characters is then applied to further identify the Brahmi specific inscription and character set images and filter out images related to other scripts. The salient features of the proposed system are:

- Multistage pipeline to improve the accuracy of Brahmi image retrieval.
- Key-point feature descriptor using SIFT algorithm [19–22] and brute force matching to discard irrelevant images.
- Multi-scale Template matching [23] using printed Brahmi characters to exclude non-Brahmi images.

2 Brahmi Script and Characters

Brahmi is an important ancient writing system that is considered a parent script of modern Indian and some Asian scripts. Brahmi was also used to record the teachings of Buddha by creating stone inscriptions. These inscriptions, called Ashoka edicts, are present in multiple places throughout the Indian sub-continent. Fig. 1 shows a sample Brahmi inscription from Ashoka Edict located at Sarnath [24] and Fig. 2 shows the characters of Brahmi script [25].

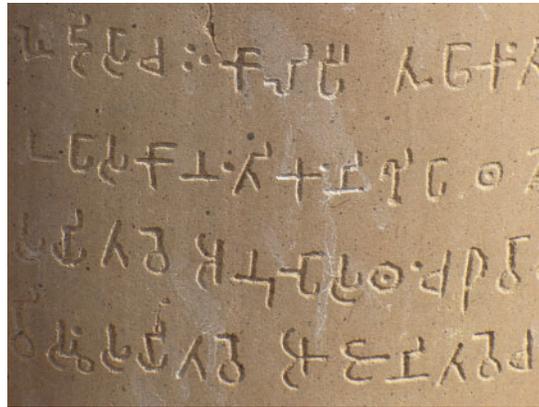


Figure 1: Ashoka edict–sarnath

𑀀 a	𑀁 ā	𑀂 i	𑀃 ī	𑀄 u	𑀅 ū
𑀆 e	𑀇 ai	𑀈 o			
𑀉 ka	𑀊 kha	𑀋 ga	𑀌 gha	𑀍 ṅa	
𑀎 ca	𑀏 cha	𑀐 ja	𑀑 jha	𑀒 ṅa	
𑀓 ṭa	𑀔 ṭha	𑀕 ḍa	𑀖 ḍha	𑀗 ṇa	
𑀘 ta	𑀙 tha	𑀚 da	𑀛 dha	𑀜 na	
𑀝 pa	𑀞 pha	𑀟 ba	𑀠 bha	𑀡 ma	
𑀢 ya	𑀣 ra	𑀤 la	𑀥 ḷa	𑀦 va	
𑀧 śa	𑀨 ṣa	𑀩 sa	𑀪 ha		

Figure 2: Brahmi character

2.1 Challenges in Identifying Brahmi Script

Brahmi is a very ancient script so the only source of the character-set images of Brahmi is inscription images captured by scholars and historians from pillars and stones. These images are mostly low-resolution images, which pose significant challenges due to visual defects. Some common defects are cuts and cracks on the inscriptions, which appear as part of the character on the image. Varied illumination due to sunlight and the angle of capturing the image causes shadows and bright spots. Fig. 3 [26,27] displays a close-up image of the Brahmi Inscription pillar, which has cracks, spots, and illumination variations created due to sunlight and skewness or camera angle.

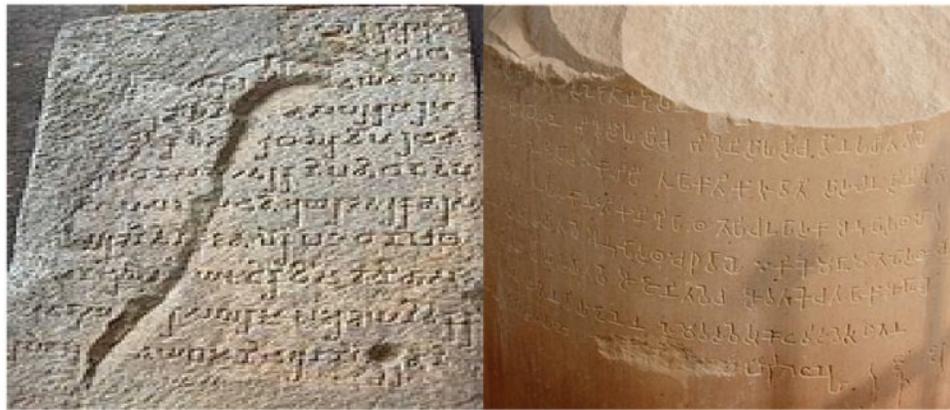


Figure 3: Camera image of Brahmi inscription

In addition to these inscription images, scholars and historians have also prepared estampages of the inscriptions in the form of printed character-set images. These have better clarity than inscription images but pose another significant challenge when trying to identify and separate the Brahmi images from other scripts. This is because some of the characters in Brahmi have similarity with characters in other scripts e.g., straight line and circle or arc is commonly used in Brahmi characters, which may result in false-positive identifications. Some of these are displayed in Tab. 1 below:

Table 1: Brahmi characters similarity to English and Devanagari characters

Brahmi character	Type of similarity	English character	Devanagari character
┆, 7, 1, +	Straight-line constitutes most of the characters, which is easily present in multiple scripts.	I	ी
⊆, ⊙	Circular or arc which can easily match with circular shapes character in other languages	O	ठ
┆, □	Horizontal and vertical straight lines, or any closed shape character	F, U, E	प, म

3 The Proposed Brahmi Image Matching System

The motivation for the development of Brahmi image matching system is the lack of a dataset collection mechanism for Indian inscription, which can be utilized for further study of inscription or to design an OCR for a Brahmi script. The proposed system utilizes a sequential matching pipeline employing two stages—scale-invariant feature matching and template matching. The reason for using a two-stage pipeline is because of the shape of the Brahmi characters. Most image matching systems employ only the scale-invariant features to measure similarity across different images; like SIFT feature has been used to generate feature set based on key areas like loops, arcs, curves, etc. for classifying Tamil script [28]. In Brahmi character images, sufficient scale-invariant key points and features are not extracted. This is because most of these methods extract key points based on corners and edges present in the image. Brahmi characters mainly include elements like a straight line, arcs, or circle; hence scale-invariant features are not useful if we consider individual Brahmi characters. To improve the overall accuracy of the system, the two-stage matching pipeline has been proposed, as shown in Fig. 4.

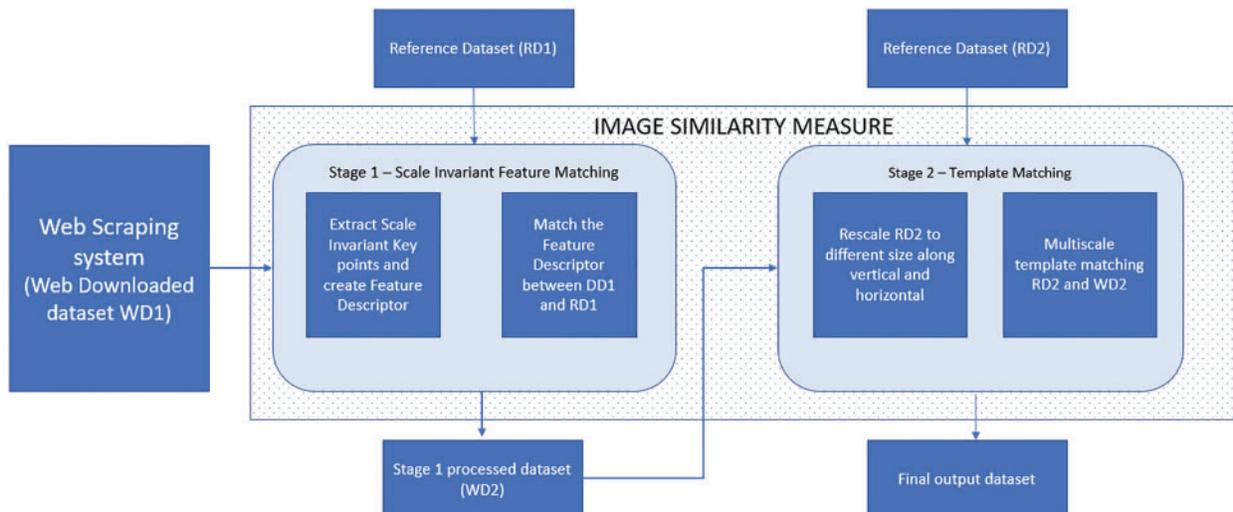


Figure 4: Proposed system workflow

To identify the similarity between images, ‘Image Similarity Measure’ is divided into two stages based on the key methods used:

- Stage-1: Key points-based feature descriptor matching using SIFT.
- Stage-2: Area/perimeter-based template matching.

The wide variety of the image dataset returned implies that the similarity measure system would need to incorporate both these matching techniques rather than applying a single matching technique. The two stages work as a pipeline such that the first stage filters specific results, which are improved by the second stage. Each stage requires a certain set of reference images against which the downloaded image is compared. Since the system uses different image datasets at different stages, these are referenced as mentioned below—

- RD1—Reference dataset for stage-1
- RD2—Reference dataset for stage-2

- WD1–Images downloaded from web using web scraping tool for comparison at stage-1
- WD2–Output of stage1 comparison, which acts as input for stage-2

3.1 Reference Dataset (RD1 and RD2)

3.1.1 Reference Dataset for Stage-1 (RD1)

The purpose of stage-1 is to discard completely irrelevant images like the image of Brahmi plants, products etc. The stage is intended to extract images of inscriptions and printed character sets. So, the reference dataset RD1 includes some Brahmi Inscription images and printed Brahmi character set images as shown in Fig. 5, which include multiple Brahmi characters. It is important to note that the RD1 set should not include images having only one or two characters.

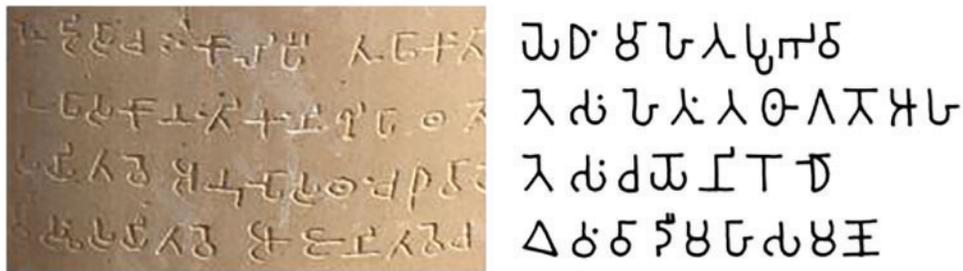


Figure 5: Inscription image and printed characters image constituting RD1

3.1.2 Reference Dataset for Stage-2 (RD2)

The purpose of stage-2 is to further refine the results of stage-1 and extract only the images related to the Brahmi character set. The RD2 set includes individual Brahmi character set images. But using the character image directly is not sufficient. The stage-1 dataset can contain images and characters which have variations in the size of characters. The reference dataset RD2 is enhanced to improve the identification by scaling each character image horizontally and vertically. This generates the RD2 character set, which now includes multiple images of each Brahmi character stretched in the horizontal or vertical direction. The steps used to create the RD2 dataset are as follows. For each Brahmi character image, $b_i \in$ printed Brahmi character set–

- Scale the image in x and y directions by a factor of 50% to 130% in increments of 10.
- Scale the image only in x-direction by a factor of 50% to 130% in increments of 10.
- Scale the image only in y-direction by a factor of 50% to 130% in increments of 10.

Each Brahmi character image is now stored as 25 different scaled images (Fig. 6). These varied size images form the basis of the multi-scale template matching of this stage.

3.2 Web Download Dataset (WD1)

By employing a Web Scraping system, which can scrape/retrieve images from various web sources based on a text query, test dataset WD1 is generated for input to stage-1. Fig. 7 explains the workflow for this system.

Input Brahmi Image	Scale %age	Scale X-Y	Scale X	Scale Y
	50			
	60			
	70			
	80			
	90			
	110			
	120			
	130			

Figure 6: Scaled images of Brahmi character image

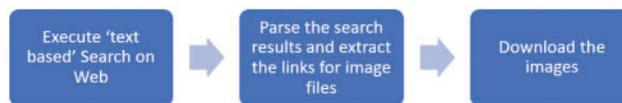


Figure 7: Web scraping system

3.2.1 Execute Image Search on Web

The most common mode of searching any image on the internet is using a ‘Text Query’. Text Query-based search includes a set of keywords as per the user’s understanding and need. There can be a various combination of keywords, and some such combinations may return similar pages in the result [29].

- Single keyword query like-‘Brahmi’, ‘Ashoka’, ‘inscriptions’.
- Multi Keyword queries like-‘Brahmi characters’, ‘Brahmi images’, ‘Brahmi script edicts’, ‘Ashoka Brahmi’, ‘Ashoka inscriptions’.

Some key points for the web scraping system are:

- Users can have multiple keywords to search or use a combination of similar keywords.
- Some of the results return entirely different objects, e.g., the search query ‘Brahmi’ returns mostly images of the Brahmi plant and another medicinal product with the same name.

- Results improve as the keywords change or more keywords are added, but the results may include some irrelevant images also.
- Some of the search queries may return similar results.
- Search results contain the similar images but not necessarily of the user’s intent e.g., using a printed Brahmi character set, the result contains images of different alphabets and scripts in similar calligraphic font.

3.2.2 Download and Save the Images

This step involves parsing the list of URLs returned by the search query and downloading the images mentioned in the links. The downloaded images from the dataset WD1 are submitted to stage-1 for matching with the reference dataset RD1 and generating the relevant image-set WD2. The dataset WD2 is the output of the stage-1 matching. The web search query returns many results shown in Fig. 8 that contain various irrelevant images. Mainly there are two types of images returned:

- Irrelevant images like Brahmi plant, some products having name as Brahmi, some images of places relevant to Brahmi and some book cover.
- Inscription images and printed character set images which could be of Brahmi or for some other script.

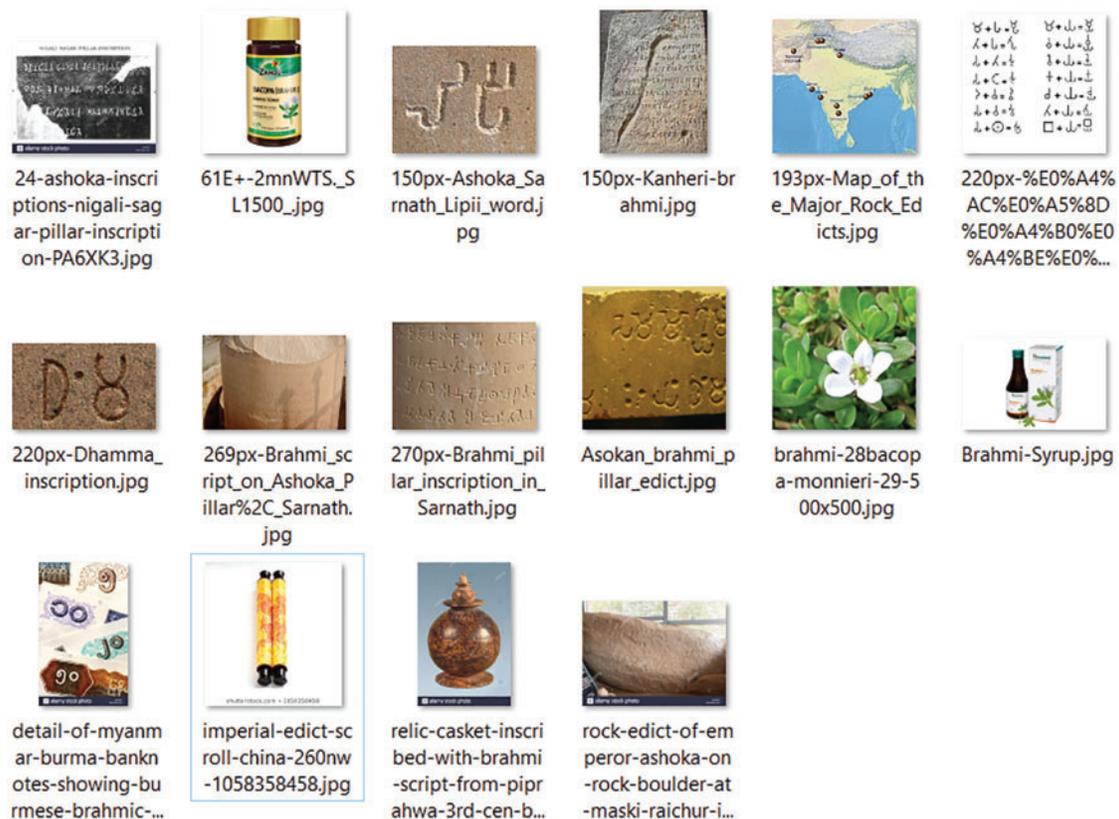


Figure 8: Image downloaded by different search queries using a keyword like “Brahmi” “Inscription” “Brahmi Character set”

3.3 Image Similarity Measure

The main and crucial part of the proposed system is to find the similarity of images with Brahmi Script. As mentioned above, we used a sequential matching pipeline to find the accurate Brahmi inscription image.

3.3.1 Stage 1–Key Points-based Feature Matching Using SIFT

The purpose of this stage is to filter out the images which are not related to inscriptions or any script. There are many images downloaded from the web pages like plants, products, etc. which are named like Brahmi but are not relevant to the Brahmi script. This stage will filter out all such images. The RD1 reference images used in this stage are images containing multiple characters from Brahmi script. Moreover, the RD1 set includes both inscription images and printed character set images which have been gathered from the Brahmi scholars and experts. Sample RD1 dataset is shown in Fig. 9.



Figure 9: RD1 reference dataset containing multiple Brahmi character images

As most of the images downloaded from the internet and the reference images contain camera snapshots with a lot of noise, the images in both datasets RD1 and WD1 are first pre-processed to reduce noise. Then, the key-point features are extracted, and key point descriptor are created using SIFT. Brute force matching is then used to match the feature descriptor of the RD1 images with each of the WD1 images. Euclidean distance is used for measuring feature similarity between the reference image feature descriptor, and the downloaded dataset images feature descriptor. Fig. 10 shows the flow diagram of stage-1.

- **Pre-Processing**–The first step is to convert the image to a greyscale image. The handwritten images mostly have a clear background, but for the inscription image, the background can contain noise. So, for the inscription or the edicts image, we need to remove noise using the following steps Fig. 11:
 - o **Greyscale conversion**–The colored inscription image is converted to Greyscale using the formulae “ $Y = 0.2125 R + 0.7154 G + 0.0721 B$ ” (where Y is the grey value of a pixel using the weighted sum of red (R), green (G) and blue (B) pixels).
 - o **Gaussian blurring and Thresholding**–Blurring is used to smoothen an image by convolving with a Gaussian Kernel. Here we have used a kernel of size (3, 3). This is followed by thresholding to remove random noise in the image. Adaptive thresholding [30] has been used, taking a block size of 50 pixels as mentioned in Eq. (1) for a pixel $p(x, y)$.

$$p(x, y) = \begin{cases} 255 & \text{if } p(x, y) > \text{Threshold} \\ 0 & \text{if } p(x, y) \leq \text{Threshold} \end{cases} \quad (1)$$

where ‘Threshold’ is the mean of 50×50 pixel block in neighborhood of pixel $p(x, y)$.

- Feature extraction–Scale-Invariant Feature Transform (SIFT) [31] is used to extract the key-points descriptor of both the images. SIFT algorithm identifies the key points of the image, which are invariant to rotation and scaling. Each key point is represented by a vector of size 128. As these features are scale-invariant, it is useful when matching similar character images having different scale.
- Feature Matching–Brute Force matching [32] is used to match each feature descriptor of the RD1 image and the feature descriptor for the WD1 image. To match or identify similarity between the two feature descriptors, Euclidean Distance is evaluated between the two feature descriptors. Brute force matching returns the best matches of the descriptor in the WD1 image and the RD1 image. Considering a key point feature descriptor of size 128 ($n = 128$), the Euclidean distance is calculated as mentioned in Eq. (2) below.

$$D_e = \left(\sum_{i=1}^n (p_i - q_i)^2 \right)^{\frac{1}{2}} \quad (2)$$

The lesser the distance between the descriptor vectors, the higher the similarity between the images. The Brute force matcher is executed such that each reference image descriptor is matched with all the key point descriptors of the downloaded images.

3.3.2 Stage 2–Multi-scale Template Matching

Stage-1 discards most non-relevant images that are not related to any inscription or any script character set. The resultant image dataset (WD2) contains images identified as an inscription or printed character-set images. Now the reference dataset for stage-2 is very different from stage-1. At this stage, Template Matching is used to identify relevant images. Therefore, RD2 includes images of each individual Brahmi character rather than a group of characters used in stage-1. Further, as the WD2 dataset may include images of various sizes, so the RD2 dataset should include reference images of various scales. Each of the WD2 images is then compared iteratively with multi-scale template of each of the Brahmi input characters. As shown in Fig. 2 of the Brahmi character-set, 30 characters out of 43 are included in RD2, excluding the characters with a similar shape. Each of these 30 characters is scaled to create a multi-scale template. 25 scaled images for each of the character generate the RD2 dataset containing 750 template images. The workflow for stage-2 is shown in Fig. 12.

- Pre-processing–The pre-processing steps for the dataset images (WD2) are same as those mentioned in stage-1. For the reference dataset images (RD2) since these are printed images so denoising is not required. Rest of the steps are same as mentioned in stage-1 pre-processing.
- Multi-Scale Template matching–The RD2 dataset images form a multi-scale template for comparison with each WD2 dataset image. Comparison is made using template matching techniques, which involves sliding the reference image RD2 over the test image WD2 and calculating the match result of the two images over the area of overlap. At each location (x, y) on the WD2 image, Normalized Correlation coefficient is calculated as mentioned in Eq. (3). This gives the normalized correlation between the two images. The result is normalized to make the comparison invariant to brightness changes across the image. As the template matching is

still sensitive to scale variation, so to handle this variation, multiple scaled images of the input character set are used for comparison.

$$M(x, y) = \frac{\sum_{x',y'} (RD2'(x', y') \cdot WD2(x + x', y + y'))}{\sqrt{\sum_{x',y'} RD2'(x', y')^2 \cdot \sum_{x',y'} WD2'(x + x', y + y')^2}} \quad (3)$$

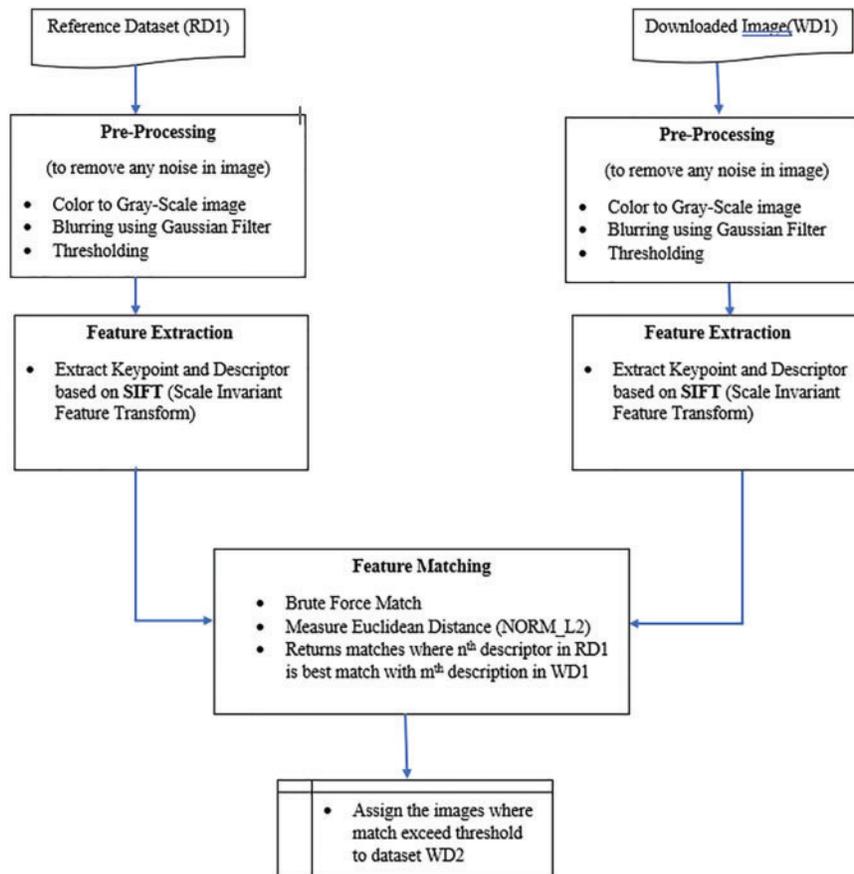


Figure 10: Flow diagram for image feature matching in stage-1

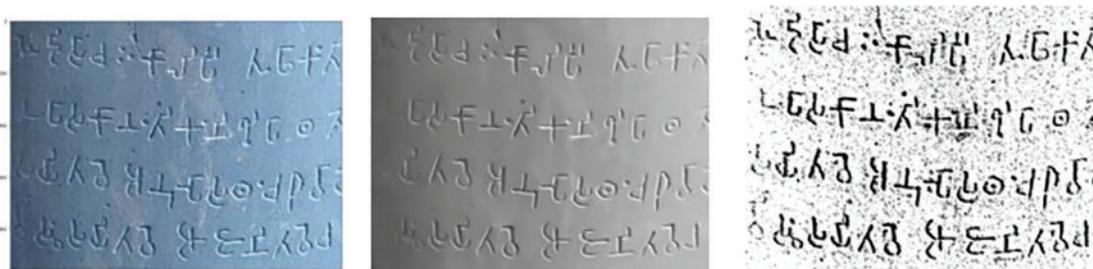


Figure 11: Pre-processing of the camera image to a gray-scale image and threshold image

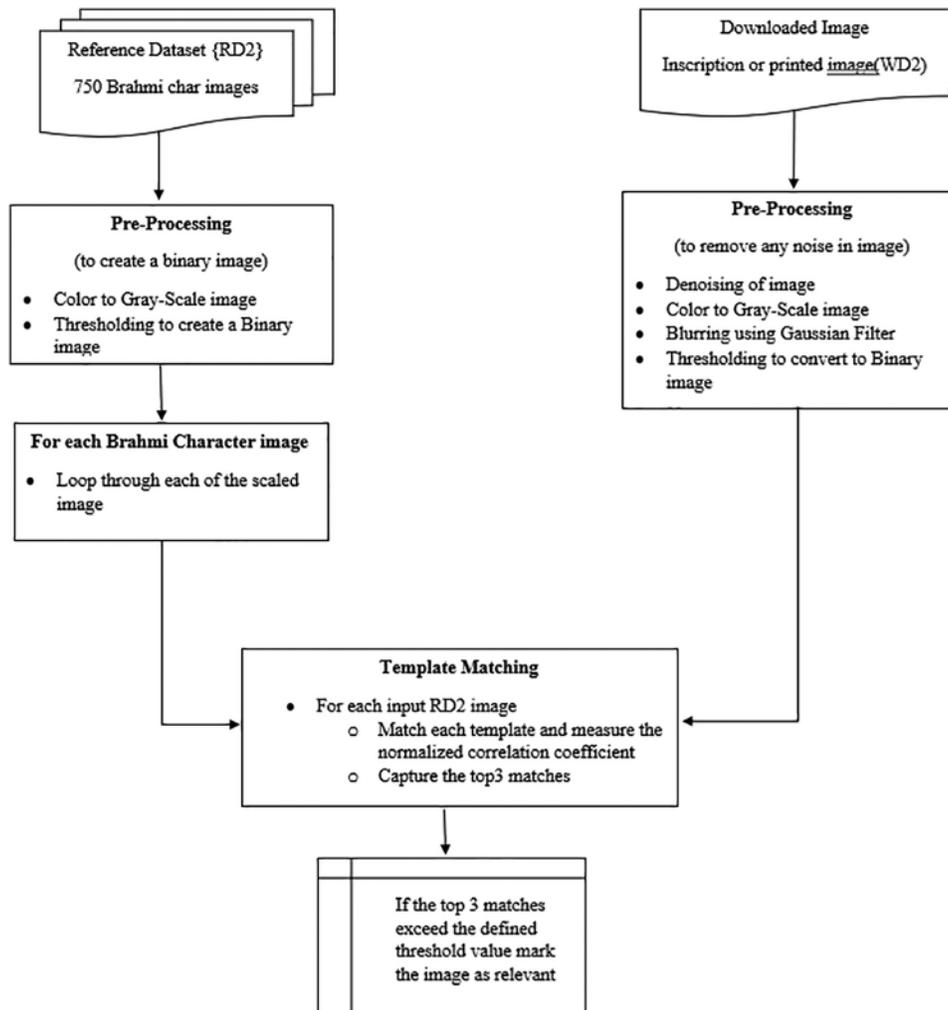


Figure 12: Flow diagram for image similarity measure stage-2

4 Experimentation and Results

Anaconda based Spyder development environment using Python 3.7 [33] has been used for the proposed system. The steps are pre-processing, key-point and descriptor generation, descriptor matching, template rescaling, and template matching. In stage-1 of image comparison, the SIFT features descriptors are extracted for each of the downloaded images, and these are compared with the SIFT feature descriptor of the reference images RD1. A matcher object is created to match the features. Figs. 13.1–13.3 are some of the feature matching results for a different sets of images considering 65% match based on Euclidean distance between the features descriptors. The images below the threshold are discarded.

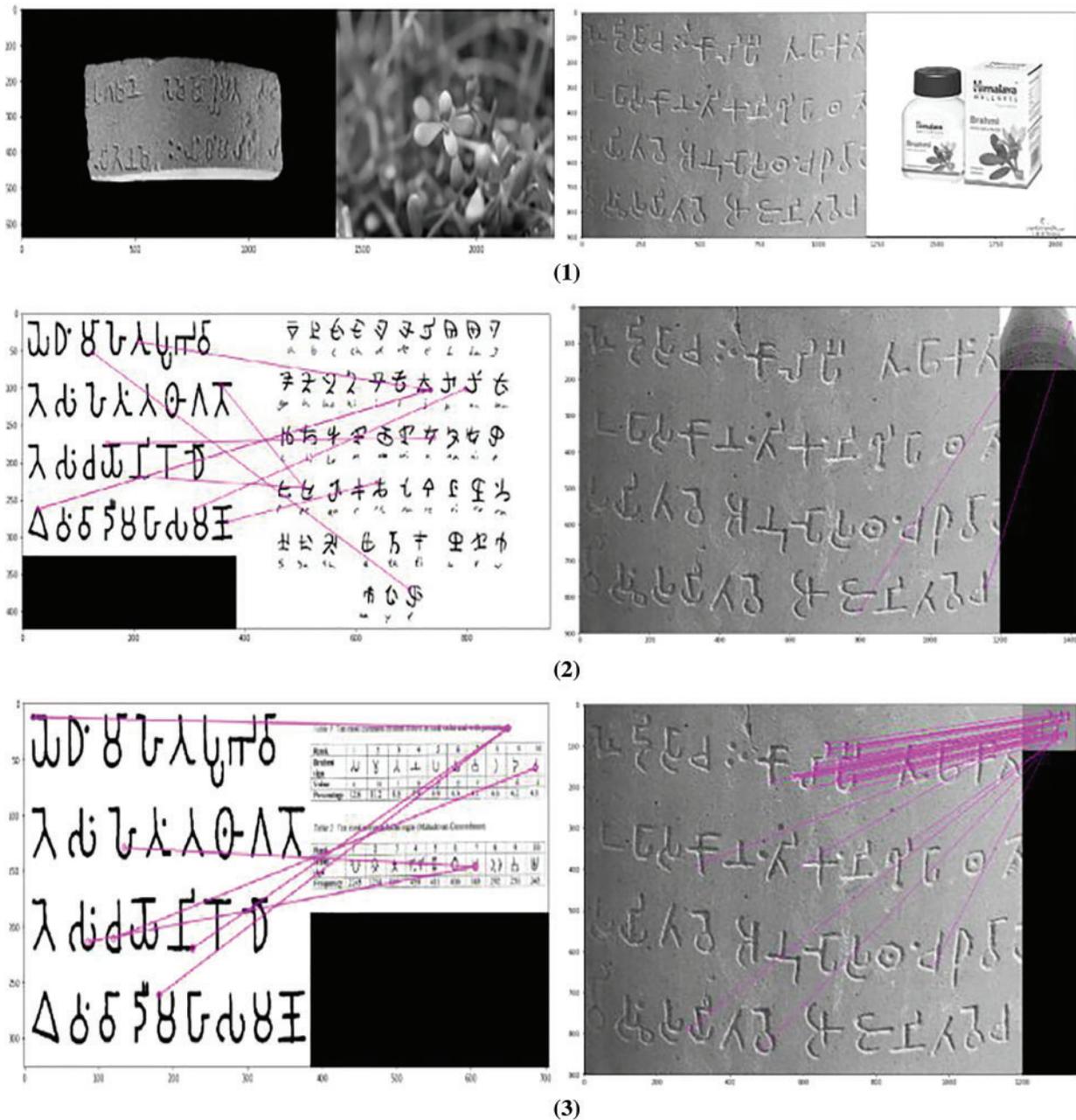


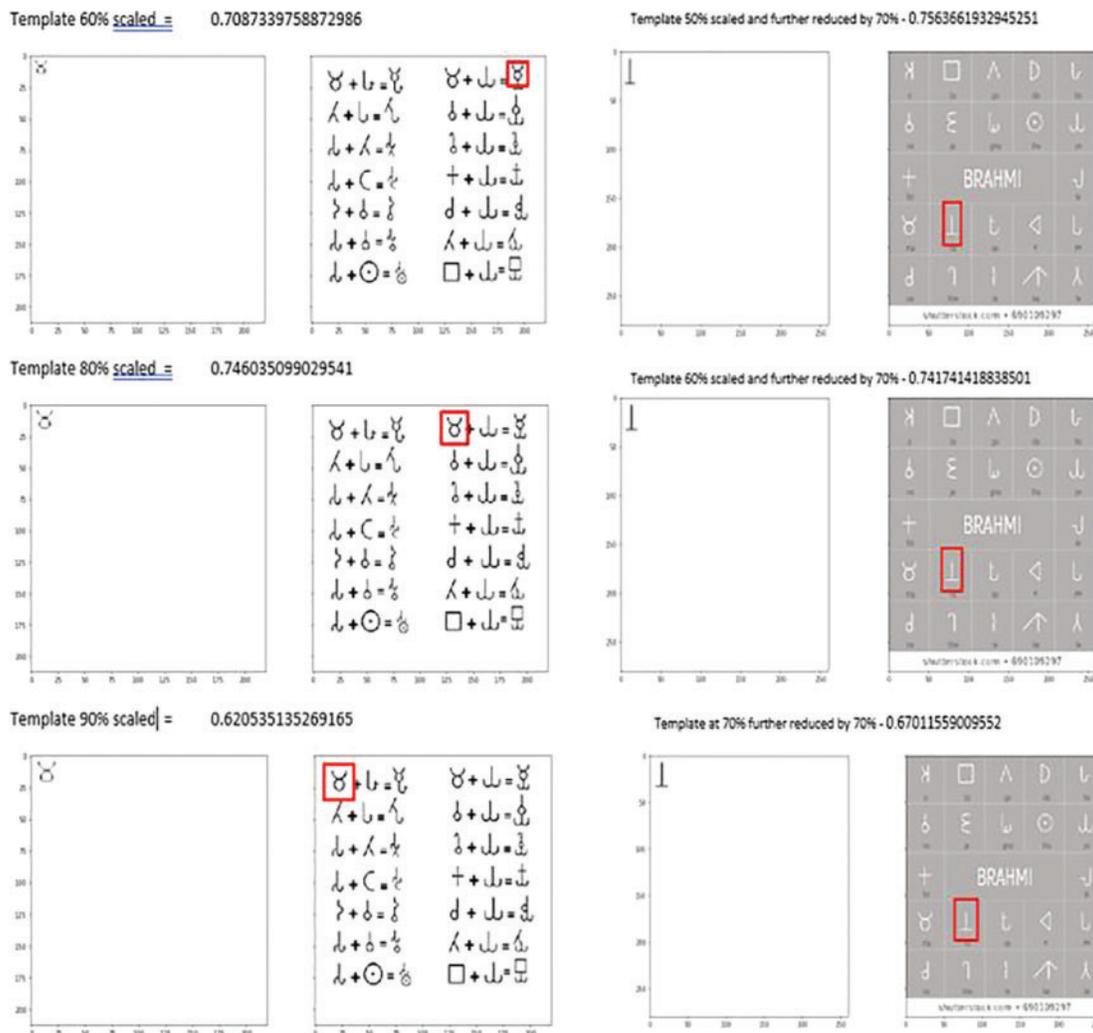
Figure 13: (1) Non-relevant images like plants or medicine are discarded during SIFT match (2) Relevant images showing match with RD1 images (3) Inscription images and printed character set images that are matched by SIFT

During the experiment, 200 downloaded images were processed through stage-1, and the stage-1 flagged 89 images as related to inscriptions and printed characters. These include some images that are of scripts other than Brahmi. These were matched by SIFT due to similarity in the shape of characters. Next stage-2 of the image comparison is executed. The images extracted and marked relevant at stage-1 are now compared with each of the multi-scale Brahmi characters using template matching. The

Normalized Correlation Coefficient (NCC) is calculated for each character image, and the results are sorted in descending order. The match is decided based on the below steps:

- Sort the results in descending order based on the NCC match value.
- Select the top three matches.
- Find the average NCC for the three matches.
- The image is marked as a Brahmi script image if it exceeds the threshold value.

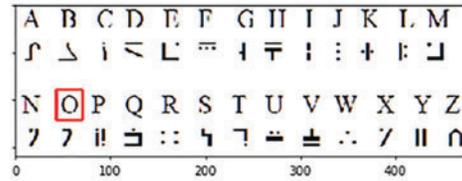
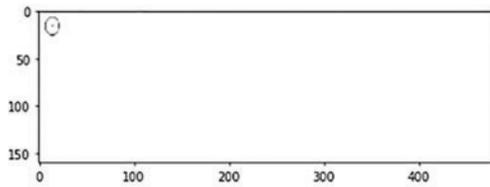
The threshold value used for the printed test image and the inscription test image would be different as the template dataset (RD2) consists of only the printed character images. Fig. 14.1 shows the comparison match results for printed character images using threshold of ‘0.7’. Fig. 14.2 shows match results for non-Brahmi printed images. Fig. 14.3 shows the comparison match results for printed character images using threshold of 0.5.



(1)

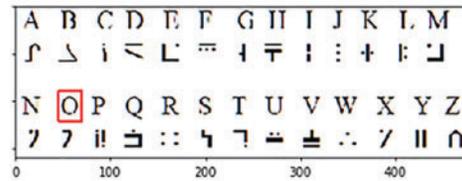
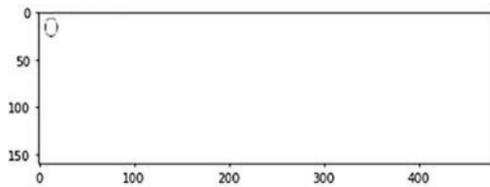
Figure 14: (Continued)

Template size scaled by 90% = 0.6557683944702148

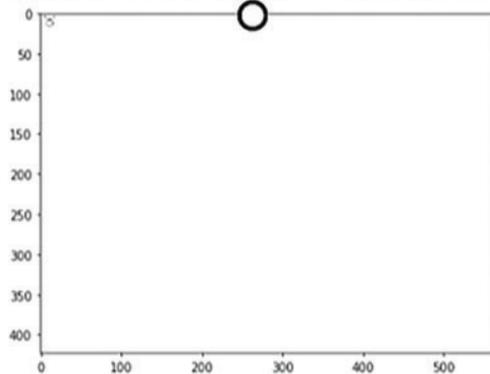


Template size scaled by 80% -> 1.jpg =

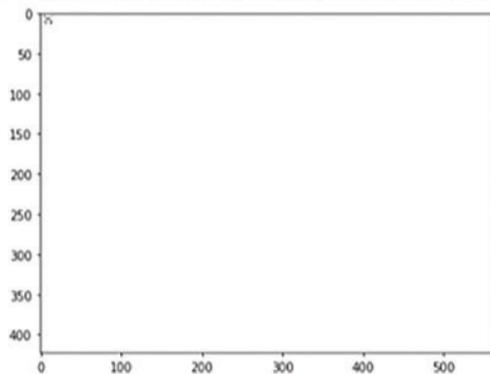
0.6192525625228882



Template size scaled by 70% = 0.5640946626663208



Template size scaled by 60% = 0.5307260751724243



(2)

Figure 14: (Continued)

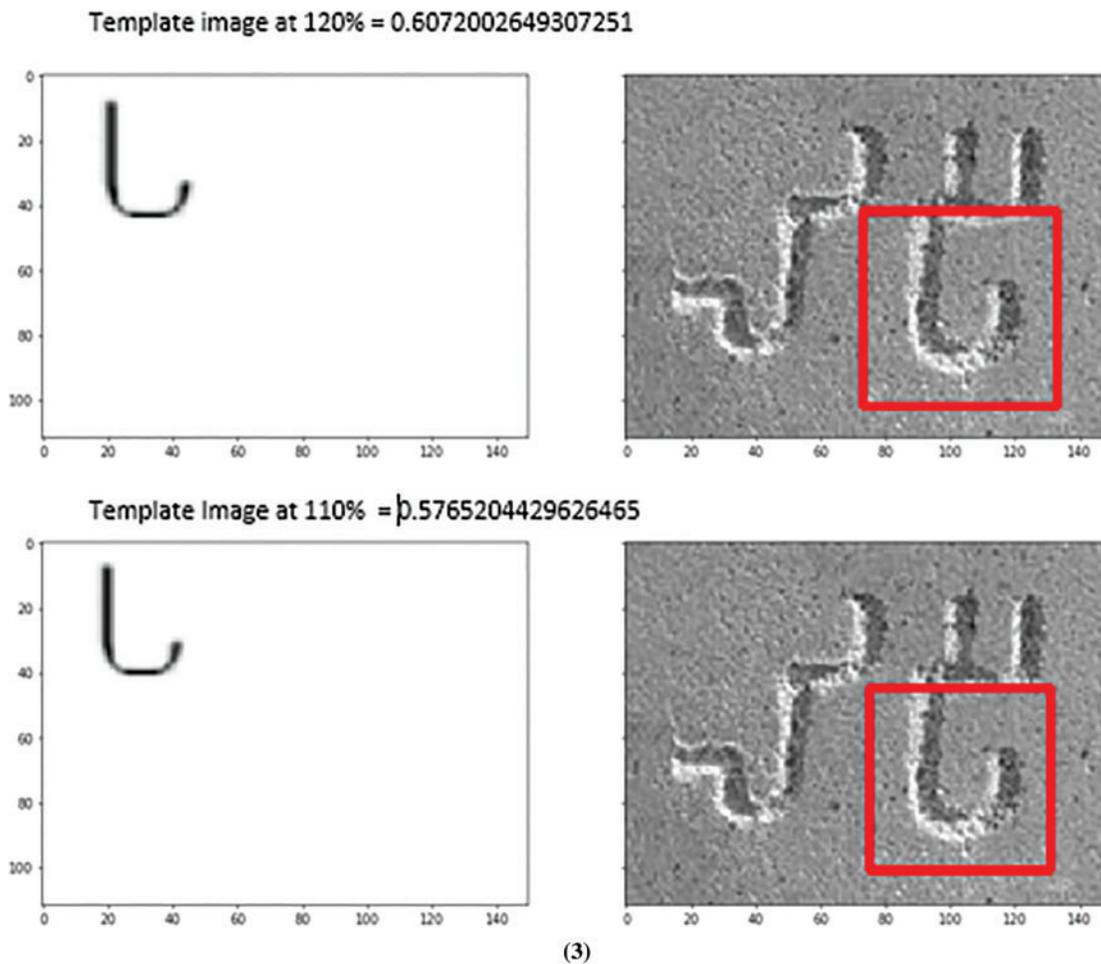


Figure 14: (1) Matching result for printed Brahmi character set image (shown on the right) with template (on the left). The template scale level and the NCC result are shown above each matching image (2) Matching result for printed non-Brahmi character set image (shown on the right) with template (on the left). The template scale level and the NCC result are shown above each matching image (3) Matching result for Brahmi inscription image (shown on the right) with template (on the left). The template scale level and the NCC result are shown above each matching image

From Fig. 14.2, we can observe that the characters are very similar to Brahmi characters and are matched with the template, but the match correlation value is lesser than the documents, including Brahmi characters.

The two-stage image similarity measure greatly improves the accuracy. If we apply the multi-scale template images directly to the images downloaded from the web, a lot of false positives get captured. This is because common shapes of Brahmi characters, including arcs, lines, etc. get easily matched with physical objects in the image. Stage-1 filters out such images and improves the template matching accuracy and hence the overall accuracy of the system. Out of the 89 images identified by stage1, stage-2 flagged 58 images as Brahmi images. Out of these, 36 were correctly Brahmi images, and 22 were for other scripts. Total of 118 images were correctly identified as non-Brahmi script images, and 24 non-Brahmi images were wrongly classified as Brahmi.

Based on the test data set of 200 images with 140 non-Brahmi and 60 Brahmi images, the confusion matrix obtained is shown in Fig. 15 below. The multistage stage system displayed an F1 score, as mentioned in Eq. (4), of 61% in classifying the Brahmi and non-Brahmi images.

$$F1 = 2 * (Precision * Recall) / (Recall + Precision) \quad (4)$$

$$\text{where } Precision = \frac{TP}{TP + FP} \text{ and } Recall = \frac{TP}{TP + FN}$$

		predicted	
		brahmi	non-brahmi
actual	brahmi	TP(36)	FN(24)
	non-brahmi	FP(22)	TN(118)

Figure 15: Confusion matrix for the proposed system

The terms used in the precision and recall are –

- TP (True Positive)–Brahmi images that have been identified by the system correctly as Brahmi.
- FP (False Positive)–Non-Brahmi images identified as Brahmi.
- TN (True Negative)–Non-Brahmi images identified as non-Brahmi.
- FN (False Negative)–Brahmi image identified as non-Brahmi.

F1 score is used as a matrix in this case as the number of objects in both classes is unbalanced. The web downloaded image set has many non-Brahmi images as compared to a few Brahmi images.

5 Conclusion

The system for identifying Brahmi script images from a larger set of web downloaded images has been proposed with the help hybrid pipeline method of SIFT based feature and template-based matching. Web search, whether based on web scraping tool or text query, returns large number of results that contain both relevant and unrelated images. Image-based queries available in most web browsers incorporate image color histogram and pattern matching, which again can return similar looking but non-relevant images. The system proposed can be used to classify the downloaded image set. Combining two of the most common image similarity measure techniques, the proposed system manages to further narrow down the search result and improve the overall accuracy of identifying only the relevant script image. Future work will include improving the accuracy of the system by handling variations due to skewness, and noisy image as many inscription images get impacted due to curvature of pillars and cracks on the surface. Moreover, we plan to incorporate deep learning-based ensemble systems by regularly improving the reference data as new images are identified and added to the local dataset. The proposed method can be extended to other ancient scripts and languages to create a local dataset of ancient scripts.

Acknowledgement: The authors would like to thank the editors of CMC and anonymous reviewers for their time and reviewing this manuscript.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

- [1] R. Mokashi and P. V. Samel, "Brahmi inscriptions from kondane caves," *Ancient Asia*, vol. 8, pp. 3–20, 2017.
- [2] A. K. Datta, "A generalized formal approach for description and analysis of major Indian scripts," *IETE Journal of Research*, vol. 30, no. 6, pp. 155–161, 1984.
- [3] C. Violatti, "Brahmi script," in *World History Encyclopedia*, Surrey, UK: World History Publishing, pp. 4456–4475, 2016.
- [4] C. Anilkumar, A. Karrothu and G. Aishwaryalakshmi, "Framework for statistical machine recognition and translation of ancient tamil script," in *Proc. Fifth Int. Conf. on Smart Computing and Informatics (SCI 2021)*, Hyderabad, India, pp. 513–522, 2022.
- [5] R. Chirimilla and V. Vardhan, "A survey of optical character recognition techniques on indic script," *ECS Transactions*, vol. 107, no. 1, pp. 6507–6514, 2022.
- [6] S. Kaur and B. B. Sagar, "Brahmi character recognition based on SVM (support vector machine) classifier using image gradient features," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 22, no. 8, pp. 1365–1381, 2019.
- [7] A. Alaei, U. Pal and P. Nagabhushan, "Dataset and ground truth for handwritten text in four different scripts," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 26, no. 4, pp. 1253001–1253020, 2012.
- [8] W. Sun, G. C. Zhang, X. R. Zhang, X. Zhang and N. N. Ge, "Fine-grained vehicle type classification using lightweight convolutional neural network with feature optimization and joint learning strategy," *Multimedia Tools and Applications*, vol. 80, pp. 30803–30816, 2021.
- [9] N. Jayanthi and I. Sreedevi, "Inscription image retrieval using bag of visual words," in *Proc. IOP Conf. Series: Materials Science and Engineering*, Hyderabad, India, vol. 225, no. 3, pp. 012211–012228, 2017.
- [10] I. Sreedevi, A. Tomar, A. Raj and S. Chaudhury, "Enhancement and retrieval of historic inscription images," in *Proc. Asian Conf. on Computer Vision*, Singapore, pp. 529–541, 2014.
- [11] N. Haddaway, "The use of web-scraping software in searching for grey literature," *Grey Journal*, vol. 11, no. 2, pp. 186–190, 2015.
- [12] R. Mitchell, "Advanced HTML parsing," in *Web Scraping with Python: Collecting Data from the Modern Web*, 2nd ed., Sebastopol, CA, USA: O'Reilly Media, pp. 150–197, 2018.
- [13] A. Latif, A. Rasheed, U. Sajid and A. J. Jameel, "Content-based image retrieval and feature extraction: A comprehensive review," *Mathematical Problems in Engineering*, vol. 23, no. 2, pp. 1–21, 2019.
- [14] L. Piras and G. Giacinto, "Information fusion in content-based image retrieval: A comprehensive overview," *Information Fusion*, vol. 37, pp. 50–60, 2017.
- [15] W. Sun, X. Chen, X. R. Zhang, G. Z. Dai, P. S. Chang *et al.*, "A multi-feature learning model with enhanced local attention for vehicle re-identification," *Computers, Materials & Continua*, vol. 69, no. 3, pp. 3549–3561, 2021.
- [16] B. Bagasi and L. A. Elrefaei, "Arabic manuscript content based image retrieval: A comparison between SURF and BRISK local features," *International Journal of Computing and Digital Systems*, vol. 7, no. 6, pp. 355–364, 2018.
- [17] U. D. Dixit and M. S. Shirdhonkar, "Language-based document image retrieval for trilingual system," *International Journal of Information Technology*, vol. 12, no. 1, pp. 1217–1226, 2020.
- [18] S. Zhalehpour, E. Arabnejad, C. Wellmon, A. Piper and M. Cheriet, "Visual information retrieval from historical document images," *Journal of Cultural Heritage*, vol. 40, no. 2, pp. 99–112, 2019.
- [19] P. Chhabra, N. K. Garg and M. Kumar, "Content-based image retrieval system using ORB and SIFT features," *Neural Computing and Applications*, vol. 32, no. 7, pp. 2725–2733, 2020.

- [20] E. Karami, M. Shehata and A. Smith, "Image identification using SIFT algorithm: Performance analysis against different image deformations," in *Proc. Newfoundland Electrical and Computer Engineering Conf.*, St. Johns, Canada, pp. 779–798, 2015.
- [21] M. Kashif, T. M. Deserno, D. Haak and S. Jonas, "Feature description with SIFT, SURF, BRIEF, BRISK, or FREAK? A general question answered for bone age assessment," *Computers in Biology and Medicine*, vol. 68, no. 1, pp. 67–75, 2016.
- [22] J. Dou, Q. Qin and Z. Tu, "Robust image matching based on the information of SIFT," *Optik*, vol. 171, no. 2, pp. 850–861, 2018.
- [23] N. S. Hashemi, R. B. Aghdam, A. S. B. Ghiasi and P. Fatemi, "Template matching advances and applications in image analysis," *American Academic Scientific Research Journal for Engineering, Technology, and Sciences*, vol. 26, no. 3, pp. 91–108, 2016.
- [24] A. Singh and A. Kushwaha, "Analysis of segmentation methods for Brahmi script," *DESIDOC Journal of Library & Information Technology*, vol. 39, no. 1, pp. 109–124, 2019.
- [25] P. D. Devi and V. Sathiyapriya, "Brahmi script recognition system using deep learning techniques," in *Proc. Third Int. Conf. on Inventive Research in Computing Applications (ICIRCA)*, Coimbatore, India, pp. 1346–1349, 2021.
- [26] G. Siromoney, R. Chandrasekaran and M. Chandrasekaran, "Machine recognition of Brahmi script," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 13, no. 4, pp. 648–654, 1983.
- [27] E. Hultsch, "Texts and translation," in *Corpus Inscriptionum Indicarum*, 1st ed., vol.1. New Delhi, India: Archaeological Survey of India, pp. 119–137, 1991.
- [28] T. Manigandan, V. Vidhya, V. Dhanalakshmi and B. M. Nirmala, "Tamil character recognition from ancient epigraphical inscription using OCR and NLP," in *Proc. Int. Conf. on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, Chennai, India, pp. 1008–1011, 2017.
- [29] S. Kohli, S. Kaur and G. Singh, "A website content analysis approach based on keyword similarity analysis," in *Proc. IEEE/WIC/ACM Int. Conferences on Web Intelligence and Intelligent Agent Technology*, Macau, China, pp. 254–257, 2012.
- [30] H. Devi, "Thresholding: A pixel-level image processing methodology preprocessing technique for an OCR system for the Brahmi script," *Ancient Asia*, vol. 1, no. 1, pp. 161–165, 2006.
- [31] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. IEEE Int. Conf. on Computer Vision*, Corfu, Greece, vol. 2, no. 2, pp. 1150–1157, 1999.
- [32] A. Jakubović and J. Velagić, "Image feature matching and object detection using brute-force matchers," in *Proc. Int. Symp. (ELMAR)*, Zadar, Croatia, pp. 83–86, 2018.
- [33] A. F. Villan, "Handling file and images," in *Mastering Open CV 4 with Python: A Practical Guide Covering Topics from Image Processing, Augmented Reality to Deep Learning with OpenCV 4 and Python 3.7*, 1st ed., Birmingham, UK: Packt Publishing Ltd., pp. 75–100, 2019.