*International Journal of Engineering & Technology*, 7 (4.41) (2018) 145-162

SPC

International Journal of Engineering & Technology

*Website: www.sciencepubco.com/index.php/IJET*

***Research paper***

# Handwritten Hindi Numeral Recognition Using Clustering Techniques

**Sukriti Paul [1],Nisha P. Shetty[2]*,**

[1] *Student, Department of Information and Communication Technology, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal-576104 INDIA*
[2]*Assistant Professor, Department of Information and Communication Technology, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal 576104 INDIA*
**Corresponding author E-mail: nisha.pshetty@manipal.edu*

**Abstract**

The problem of automated Hindi numeral recognition is a challenging task owing to the complexity of the script which is characterized by concavities, holes and curvatures. In case of handwritten numerals, the varying writing styles of individuals have to be considered. Our paper focuses at tackling the Hindi numeral recognition problem via various clustering techniques and evaluating them. Subsequently, we work on modifying the framework in Joint Unsupervised Learning (JULE) of Deep Representations and Image Clusters, with different convolutional neural network (CNN) architectures, to obtain normalized mutual information (NMI) results which are better than the state of the art results. Additionally, clustering results obtained on applying different de-noising and contrast adjustment techniques have been presented.

***Keywords:*** *Indian Script Recognition; Clustering; Joint Clustering; Devanagari numerals; K-means Clustering; Hierarchical Agglomerative Clustering; BIRCH Clustering*

## 1. Introduction

Devanagari script finds its origin in the Brahmi script and is widely incorporated in different native languages of India and Nepal; Hindi is an Indian language that makes use of such an abugida. It is the official language of India with more than 400 million first language speakers (based on the 2001 census) [1]. Recognition of native script characters can be extended to a plethora of recognition applications like those of vehicle number plates, shop and municipal sign boards, milestones, government tax forms and handwritten postal addresses. Characterized by rounded shapes, the variability of handwritten Hindi numerals poses to be a challenge as each writer has a different writing style. In addition to this, voluminous samples tend to reduce the recognition accuracy due to textual variation. Hence there is a transition from traditional prototype matching techniques to classification methods like artificial neural network (ANNs) and support vector machines (SVMs).

## 2. Background Theory

Machine recognition of hand-printed Devanagari script dates back to 1977 during which I.K.Sethi and B.Chatterjee designed a method [2] based on concatenation of primitive elements used in a multi-step decision making process. A detailed survey covering various techniques in both, optical character recognition (OCR) development and research work pertaining to the recognition of Indian scripts can be found in [3]. An artificial neural networks approach has been incorporated in [4], in the classification stage of the OCR system, for printed Devanagari Script. Sandhya et.al [5] utilize extracted shadow features, chain code histogram features, view based features and longest run features to compare classifiers like SVM and ANN for handwritten Devanagari character recognition. High error rates due to word to character segmentation, have been overcome by implementing a Bidirectional Long Short Term Memory (BLSTM) to recognize printed Devanagari text [6].

A considerable amount of literature pertaining to Devanagari numeral recognition has been reported. Most of the works are unique in terms of feature extraction methodologies and classification techniques. Research work in this field [7] can be found as early as the 1970s. In 2006, a two-stage classification system [8] was designed for Devanagari numeral recognition; it was based on ANN and HMM classifiers that used directional-view-based strokes of a character image as their inputs. The algorithm in [9] couples supervised and unsupervised learning via a general fuzzy hyper line segment neural network for a similar problem. Extending the scope of the problem statement to handwritten script, Reena et.al. proposes a multi-classifier connectionist architecture [10] as a solution to the same. Multistage recognition is implemented in [11] where three multilayer perceptron classifiers corresponding to three coarse-to-fine resolution levels, are applied to Devanagari, Bangla and English numerals, in a cascaded manner.

Although there are numerous schemes pertaining to the classification of different Indic scripts, only a handful utilize clustering techniques for this. Kannada numerals have contour points associated with them. Features extracted from directional chain code information of the contour points are fed into the k-means clustering algorithm in [12]. Yoshinobhu et.al.[13] proposed a concept of within-category clustering followed by between category clustering of mislabeled characters, after performing handwritten character recognition.Regional K-means clustering followed by using an SVM classifier for Hindi character recognition, has been proposed in [14]. We evaluate various clustering algorithms used for recognizing Hindi numerals.

### 2.1. Our Contribution

This paper provides an in-depth comparative study of evaluating four clustering algorithms against the Hindi numeral data-set. The K-means clustering, hierarchical agglomerative clustering, BIRCH and JULE algorithm in [15], have been employed. Clustering results obtained before and after image de-noising and contrast adjustment, have been noted. Furthermore, the CNN architecture present in the JULE framework has been modified to increase the clustering quality. Our work is the first of its kind, w.r.t clustering methodologies for Indic script recognition. Section 2 describes our approach to evaluating different clustering algorithms for the given dataset. In section 4, we tabulate the results and draw inferences from them. Our conclusions have been consolidated in section 4.

### 2.2. Dataset Description

The numeral data is obtained from the Devanagari Character Dataset [16]. It comprises 2880 samples (288 samples per class). The original images are binarized and have a spatial resolution of 36X36 pixels. The process of clustering is preceded by image enhancement.

## 3. Methodology

### 3.1. Image Pre-processing

In order to observe the effect of image de-noising and enhancement, on the cluster quality, several traditional image transformation techniques are applied. Results for each of the techniques are analysed in section 4. Gamma and logarithmic corrections are performed to adjust the contrast of the image. Gamma correction is based on the non-linear power-law expression as shown in eqn (1), where B=1 generally. This mode of correction is used to rectify the luminance and governs the image brightness. The log correction leads to a pixel level transformation of the image, as shown by eqn (2), where each input pixel is scaled in the range [0, 1].

$$V_{output} = B \times V_{input}^{\gamma} \tag{1}$$

$$Output = gain \times \log(1 + I) \tag{2}$$

The median, bilateral and Gaussian filters have been applied separately, to de-noise the images. The median filter is a de-noising filter which replaces the central pixel value with the median of neighbouring pixel entries. Consisting of 2 kernels, the bilateral filter is an edge preserving filter used for smoothing an image. It is the weighted average of pixels, that considers the closeness of pixels and its neighbours both, spatially and w.r.t radiometric similarity. The following equation gives fb: I→R, the output obtained on bilateral filtering.

$$fb(i) = \frac{\sum_{j \in \Omega} w(j)\phi(f(i-j) - f(i)) f(i-j)}{NormalizationFactor} \tag{3}$$

where

$$Normalization\ Factor = \sum_{j \in \Omega} w(j)\phi(f(i-j) - f(i)) \tag{4}$$

The range and spatial kernels are φ(s) and w(j) respectively, where s is the intensity of a given pixel. $\sigma_s$ and $\sigma_r$ are the standard deviation values of the domain and the range kernels correspondingly. Eqn.5 depicts a 2-D Gaussian distribution where x and y are the spatial coordinates while σ is the standard deviation of the distribution.

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp^{-\frac{x^2+y^2}{2\sigma^2}} \tag{5}$$

Contrast stretching is a point operation aimed at distributing the original intensity values of images to a given range or span of values, thereby 'stretching' the intensity range. Also known as non-linear normalization, it targets localized low contrast areas and increases the contrast of areas with homogeneous intensities. Histogram equalization works on the intensity histogram on an image. Pixel values are re-assigned via a monotonic and nonlinear mapping to obtain an output image having uniformly distributed intensities or a flatter histogram.

### 3.2. Clustering Techniques

Partitional and Hierarchical clustering methods have been used to assess the cluster quality on Hindi numerals. K-means clustering is a type of partitional clustering used for unsupervised categorization of n multi-dimensional data-points into k clusters or groups. The data points are $X_1 \ldots X_N$. It is an iterative process initialized by selecting k data points or instances randomly. Each data point is associated with its corresponding centroid as shown in algorithm 1.

**Algorithm 1:** The k-means algorithm

**Data:** The Devanagari Numeral dataset
**Result:** k different clusters and their respective centroids

```
initialize M_1, ..., M_k as randomly chosen cluster centres;
while The objective function continues to improve do
    C_1, ..., C_k ← φ;
    for i ← 1 to n do
        j ← argmin_j' ||X_i − M_j'||^2 add i to C_j
    for i ← 1 to k do
        M_j = (1/|C_j|) Σ_{i∈C_j} x_i
```

$C_j$ are the sets of points with cluster center as $M_j$ and $M_1, \ldots, M_k$ are the cluster centres. The objective function (OF) is given by eqn.6.

$$OF = \sum_{i \in [N]} \min_{j \in k} \|x_i - \mu_j\|^2 \tag{6}$$

Birch clustering is an incremental algorithm depending upon CF-trees (cluster feature trees). By expressing the time and space complexity clearly, it approaches the clustering problem in a way that is suitable for massive data-sets. A dense region of points is collectively viewed as a storage of a compact summarization or a cluster feature [17]. These summaries depict the natural closeness of data, thereby reducing the original problem of clustering data to clustering summaries. The distance measures taken into consideration are cluster centre, radius, diameter, centroid Euclidean distance, centroid Manhattan distance, the average inter-cluster distance, average intra-cluster distance and variance increase distance. The sub-cluster information is maintained as a triple which corresponds to a CF entry. The triples represent the number of data points in the cluster, linear sum and square sum of these data points.

A CF tree is characterized by a threshold and branching factor for leaf nodes and non-leaf nodes. Figure1 represents an overview of BIRCH.

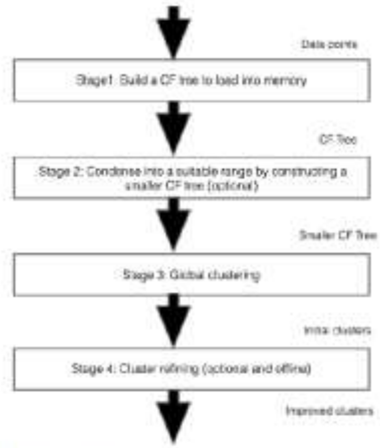


**Figure 1** The BIRCH algorithm overview

Agglomerative clustering is a 'bottom-up' clustering technique which begins with each data point behaving as a different cluster. The aim is to merge the clusters at each stage, till we obtain a single cluster comprising all the data points. Algorithm 2 demonstrates this approach. We consider 3 types of linkages to measure the distance between the clusters, namely, Ward, complete link and average link.

**Algorithm 2:** Hierarchical agglomerative clustering

**Input:** N items to be clustered, NXN similarity matrix
**Output:** A cluster containing N data points
1 Allot each item to a different cluster (N different clusters)
2 Let cluster similarity be the distance between the items they contain
3 Obtain the nearest cluster pair and merge them
4 Calculate the distance between the new cluster and each of the old clusters
5 Repeat *Steps 3 to 4*, till a single cluster is obtained

NMI values are our primary focus for assessing the clustering quality. In addition this, we have considered the Rand index and Fowlkes Mallows scores as metrics for measuring cluster quality.

# 4. Experiments

## 4.1. Performance Metrics

The following performance measures are considered to evaluate the cluster quality:
Normalized Mutual Information: To compute the NMI, the class labels of the instances are required. It can be used to compare different types of clustering algorithms with different clusters, owing to the fact that it gives us a normalized value. The NMI is given by eqn. (7).

$$NMI(A,B) = \frac{2 \times I(A;B)}{[H(A) + H(B)]}$$

where A and B represent the class labels and cluster labels respectively H(X) denotes the entropy of X and I (A;B) is the mutual information between A and B.NMI values have lie between [0, 1], where 1 corresponds to correct cluster labels for all data points.

Rand Index (RI): A pair-wise approach is followed to find the ratio of correct decisions succeeding clustering. Eqn. (8) is used to calculate the same.

$$RI = \frac{TP + TN}{TP + TN + FP + FN} \qquad (8)$$

where TP, TN, FP and FN correspond to the number of true positives, true negatives, false positives and false negatives respectively.
Fowlkes Mallows Score (FM): The similarity between two clusterings (either a clustering and a benchmark classification or two separate hierarchical clusterings) can be found via this measure. Greater values of the score imply a higher similarity between the two clusterings considered. The Fowlkes Mallows score is given by eqn. (9).

$$RI = \sqrt{\frac{TP}{TP + FP} \times \frac{TP}{TP + FN}} \qquad (9)$$

where TP,FP and FN denote the number of true positives, false positives and false negatives respectively.

## 4.2. Results and Inference

On performing k-means, BIRCH and hierarchical agglomerative clustering, we use NMI, RI and FM scores to determine the clustering quality. Table 2 illustrates that the BIRCH algorithm yields the highest NMI value for the Hindi numerical dataset.
We repeat the 3 clustering approaches mentioned in section 3.2, for contrast enhanced and denoised images, thereby recording the values of the performance metrics for each of the cases. Table 3 displays the results obtained on applying gamma and logarithmic corrections to the original dataset. Applying the BIRCH algorithm on gamma corrected images proves to be effective, in terms of NMI values. The value of gain = 1 and &$\gamma$= 2.2. Contrast stretching and histogram equalization have been performed on the dataset, independently; the values of the performance measures are shown in Table 4. Image intensities are re-scaled between the 2nd and 98th percentile of the original image intensities. Table 5 enables us to compare the effect of various de-noising techniques on the performance metrics. A 5X5 bilateral filter has been considered, with spatial parameter $\sigma_s$= 15 and range parameter $\sigma_r$ = 0.1. We also use a 3X3 median filter and a 5X5 Gaussian filter with $\sigma$ = 1. On analyzing the 3 cases, it can be inferred that the NMI values obtained after performing gamma correction, contrast stretching and bilateral filtering are better than the values presented in Table 1.

| Hyper-parameter | LR | BS | $\eta$ | $\lambda$ | $\gamma$ |
|---|---|---|---|---|---|
| Value | 0.001 | 20 | 0.9 | 1.0 | 2.0 |

**Table 1** Clustering results for the original images

Proposed by Jianwei et.al., the framework in [15] involves consecutive operations in a clustering algorithm expressed as steps in a recurrent process, assembled on top of representations output by a CNN. The algorithm has various hyper-parameters shown in Table 1. A learning rate (LR) of 0.001 and batch size (BS) of 2 has been considered. □ represents the unrolling rate for the recurrent process while $\lambda$ is the regularization parameter. We have modified the CNN architectures and varied the filter size of the convolutional layers, to obtain different NMI values depicted in Table 6. In the case of original images, the JULE algorithm yields NMI values as high as 0.87259. With a maximum NMI value of 0.87778, it can be observed that the JULE algorithm can increase the NMI values considerably, for different CNN architectures and denoised images. The highest NMI value is obtained by designing a network with a convolutional layer, batch normalization, relu and maximum pooling layer repeated thrice, in succession. $NMI_{ori}$ and $NMI_{bil}$ correspond to NMI values obtained for original images and those which are denoised by bilateral filters, respectively.

| Method | NMI | Rand Index | FM score |
|---|---|---|---|
| K−means | 0.63296 | 0.50645 | 0.55775 |
| BIRCH | **0.71393** | 0.47543 | 0.55369 |
| AC−Ward | 0.39021 | 0.20557 | 0.30239 |
| AC−Average | 0.39037 | 0.15062 | 0.32173 |
| AC−Complete | 0.37623 | 0.17155 | 0.28713 |

**Table 2**. Clustering results for the original images

**Table 3**. Clustering results on applying Gamma and Logarithm corrections to the images

| Method | Correction | NMI | Rand Index | FM Score |
|---|---|---|---|---|
| K-means | Gamma | 0.65624 | 0.53050 | 0.57918 |
| BIRCH | Gamma | **0.74723** | 0.56499 | 0.61984 |
| AC-Ward | Gamma | 0.45281 | 0.26764 | 0.35562 |
| AC-Average | Gamma | 0.48516 | 0.25190 | 0.38984 |
| AC-Complete | Gamma | 0.43702 | 0.24413 | 0.33789 |
| K-means | Log | 0.63118 | 0.47325 | 0.53435 |
| BIRCH | Log | 0.68207 | 0.46257 | 0.53504 |
| AC-Ward | Log | 0.20228 | 0.09608 | 0.20048 |
| AC-Average | Log | 0.21844 | 0.07752 | 0.23715 |
| AC-Complete | Log | 0.21065 | 0.09061 | 0.22421 |

**Table 4**. Clustering results on applying contrast stretching and histogram equalization on images

| Method | Enhancement | NMI | Rand Index | FM Score |
|---|---|---|---|---|
| K-means | Contrast stretch | 0.64088 | 0.51507 | 0.56572 |
| BIRCH | Contrast stretch | **0.73212** | 0.54609 | 0.60679 |
| AC-Ward | Contrast stretch | 0.37661 | 0.19776 | 0.29693 |
| AC-Average | Contrast stretch | 0.39047 | 0.16646 | 0.31050 |
| AC-Complete | Contrast stretch | 0.36995 | 0.15120 | 0.28173 |
| K-means | Histogram eq. | 0.61612 | 0.49568 | 0.54745 |
| BIRCH | Histogram eq. | 0.67260 | 0.47034 | 0.53611 |
| AC-Ward | Histogram eq. | 0.50185 | 0.33325 | 0.41151 |
| AC-Average | Histogram eq. | 0.47636 | 0.27846 | 0.37854 |
| AC-Complete | Histogram eq. | 0.48453 | 0.32700 | 0.40551 |

**Table 5** Clustering results for de-noised images

| Method | Filter | NMI | Rand Index | FM Score |
|---|---|---|---|---|
| K-means | Bilateral Filter | **0.74879** | 0.56638 | 0.62253 |
| BIRCH | Bilateral Filter | 0.73212 | 0.54609 | 0.60679 |
| AC-Ward | Bilateral Filter | 0.40289 | 0.22864 | 0.31876 |
| AC-Average | Bilateral Filter | 0.42366 | 0.15560 | 0.34160 |
| AC-Complete | Bilateral Filter | 0.38915 | 0.12962 | 0.30039 |
| K-means | Gaussian Filter | 0.64373 | 0.51618 | 0.56661 |
| BIRCH | Gaussian Filter | 0.73466 | 0.54046 | 0.59918 |
| AC-Ward | Gaussian Filter | 0.22503 | 0.09944 | 0.21687 |
| AC-Average | Gaussian Filter | 0.23185 | 0.07426 | 0.24957 |
| AC-Complete | Gaussian Filter | 0.22250 | 0.07836 | 0.21219 |
| K-means | Median Filter | 0.61500 | 0.48364 | 0.53803 |
| BIRCH | Median Filter | 0.68155 | 0.47575 | 0.54743 |
| AC-Ward | Median Filter | 0.37377 | 0.16944 | 0.28364 |
| AC-Average | Median Filter | 0.36818 | 0.11481 | 0.31135 |
| AC-Complete | Median Filter | 0.33373 | 0.14674 | 0.26912 |

**Table 6**. Results obtained across various CNN architectures, on using the JULE algorithm for the original images

| Architecture | A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|---|
| Convolution1 | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| Batchnorm1 | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| Relu1 | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| Pool1 | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| Convolution2 | ✔ | ✔ | ✔ | ✔ | X | ✔ | ✔ | X | ✔ |
| Batchnorm2 | ✔ | ✔ | ✔ | ✔ | X | ✔ | ✔ | X | ✔ |
| Relu2 | ✔ | ✔ | ✔ | ✔ | X | ✔ | ✔ | X | ✔ |
| Pool2 | X | X | X | ✔ | X | ✔ | ✔ | X | ✔ |
| Convolution3 | X | ✔ | ✔ | X | X | X | ✔ | X | ✔ |
| Batchnorm3 | X | ✔ | ✔ | X | X | X | ✔ | X | ✔ |
| Relu3 | X | ✔ | ✔ | X | X | X | ✔ | X | ✔ |
| Pool3 | X | X | X | X | X | X | ✔ | X | ✔ |
| Convolution4 | X | X | ✔ | X | X | X | X | X | X |
| Batchnorm4 | X | X | ✔ | X | X | X | X | X | X |
| Relu4 | X | X | ✔ | X | X | X | X | X | X |
| Pool4 | X | X | X | X | X | X | X | X | X |
| Ip1 | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| L2-Norm | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| Weight loss | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ | ✔ |
| Conv Filter Size | 5 | 5 | 5 | 5 | 5 | 3 | 3 | 3 | 5 |
| NMI[illegible] | 0.83559 | 0.78493 | 0.69029 | 0.83640 | 0.83824 | **0.87259** | 0.85586 | 0.82184 | FIND |
| NMI[illegible] | 0.80710 | 0.75379 | 0.68765 | 0.80334 | 0.81203 | 0.84311 | **0.87778** | 0.86641 | FIND |

## 5. Conclusion

Indic script recognition is an on-going research area; our approach can be extended to other native scripts like Kannada, Bangla or Oriya. Our work makes use of a flattened image vector as a feature vector for clustering. It is plausible that the NMI values can be enhanced by better feature extraction techniques, CNN architectures or image pre-processing techniques.

## References


[1] List of languages by number of native speakers in India https://en.wikipedia.org/wiki/List_of_languages_by_number_of_native_speakers_in_India.

[2] Sethi, Ishwar & Chatterjee, B. Machine recognition of constrained hand printed Devanagari. Pattern Recognition. 9. 69-75. 10.1016/0031-3203(77)90017-6, 1977.

[3] Pal, Umapada & Chaudhuri, Bidyut. Indian script character recognition-A survey. Pattern Recognition 37, 1887-1899. Pattern Recognition. 37. 1887-1899. 10.1016 /j.patcog. 2004.02.003.

[4] Singh, Raghuraj & S Yadav, C & Verma, Prabhat & Yadav, Vibhash. (0002). Optical Character Recognition (OCR) for Printed Devnagari Script Using Artificial Neural Network. International Journal of Computer Science & Communication. 1. 91-95.

[5] Sandhya Arora and Debotosh Bhattacharjee and Mita Nasipuri and Latesh G. Malik and Mohantapash Kundu and Dipak Kumar Basu. Performance Comparison of SVM and ANN for Handwritten Devnagari Character Recognition. CoRR, vol. abs/1006.5902, 2010.

[6] N. Sankaran and C. V. Jawahar. Recognition of printed Devanagari text using BLSTM Neural Network. Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012), Tsukuba, pp.322-325, 2012.

[7] I. K. Sethi and B. Chatterjee. Machine Recognition of Hand-printed Devnagri Numerals. IETE Journal of Research, volume 22, pp.532-535.

[8] U. Bhattacharya, S.K. Parui, B. Shaw, K. Bhattacharya. Neural Combination of ANN and HMM for Handwritten Devanagari Numeral Recognition. Tenth International Workshop on Frontiers in Handwriting Recognition, 2006.

[9] Patil, P. M. and Sontakke, T. R. Rotation, Scale and Translation Invariant Handwritten Devanagari Numeral Character Recognition Using General Fuzzy Neural Network. Pattern Recogntion, pp.2110–211, 2007.

[10] Bajaj, Reena and Dey, Lipika and Chaudhury, Santanu. Devnagari numeral recognition by combining decision of multiple connectionist classifiers. Sadhana, pp. 59–72, 2002.

[11] U. Bhattacharya and B. B. Chaudhuri. Handwritten Numeral Databases of Indian Scripts and Multistage Recognition of Mixed Numerals. EEE

Transactions on Pattern Analysis and Machine Intelligence, vol. 31, no. 3, pp. 444-457, March 2009.
[12] H. R. Mamatha, K. S. Murthy, A. V. Veeksha, P. S. Vokuda and M. Lakshmi. Recognition of Handwritten Kannada Numerals Using Directional Features and K-Means. 2011 International Conference on Computational Intelligence and Communication Networks, Gwalior, 2011, pp. 644-647.
[13] Yoshinobu Hotta, Satoshi Naoi and Misako Suwa Handwritten Numeral Recognition Using Personal Handwriting Characteristics Based On Clustering Method. Applications of Computer Vision, 1996.WACV '96., Proceedings 3rd IEEE Workshop on, Sarasota, FL, USA, 1996, pp. 284-289.
[14] A. Gaur and S. Yadav. Handwritten Hindi character recognition using k-means clustering and SVM. 2015 4th International Symposium on Emerging Trends and Technologies in Libraries and Information Services, Noida, 2015, pp. 65-70.
[15] Jianwei Yang and Devi Parikh and Dhruv Batra. Joint Unsupervised Learning of Deep Representations and Image Clusters. CoRR, vol.abs/1604.03628,2016.
[16] Pant, Ashok Kumar and Panday, Sanjeeb Prasad and Joshi, Shashidhar Ram. Off-line Nepali handwritten character recognition using Multilayer Perceptron and Radial Basis Function neural networks. Third Asian Himalayas International Conference on Internet, 2012.
[17] Zhang, T., Ramakrishnan, R. Livny, M. BIRCH: A New Data Clustering Algorithm and Its Applications. Data Mining and Knowledge Discovery (1997) 1: 141. https://doi.org/10.1023/A:1009783824328