Recognition of Kannada Characters in Scene Images using Neural Networks

Shahzia Siddiqua
School of Computing & Information Technology
Reva University
Bengaluru 560064, India
sshaz79@gmail.com

Naveena C
Department of Computer Science & Engineering
SJBIT
Bengaluru 560060, India
naveena.cse@gmail.com

Sunilkumar S Manvi
School of Computing & Information Technology
Reva University
Bengaluru 560064, India
ssmanvi@reva.edu.in

***Abstract*— In recent years, there has been significant progress in recognition of text from scene images. However, very less work is done for Kannada language. The focus of this paper is to identify Kannada character set. Researchers have purported a number of deep learning procedures for natural scene text detection and recognition. In this work, basic transfer learning neural network classification models are trained with 1700 scene Kannada character images extracted from Chars74K standard dataset, and labelled individual folders as per character names. Stochastic gradient descent solver with different epoch, learning rate and momentum is applied. Also, the performance of Alex net model is improved by inducing a batch normalization layer in the hidden layers. The results show that for characteristic features of Kannada text, the combination of Vgg19 network model used with higher split ratio and stated training options and values gave 100% accuracy. Also, the modified Alex net model performed better with accuracy up to 96 percent and lower error rate of 0.0134.**

***Keywords*— Convolutional Neural Network, Kannada Character Recognition, Deep Learning Classification, Natural Scene Text Recognition**

I. Introduction

Researching on recognition of optically processed characters to the highest level of accuracy has been a challenging task for long decades. The instant ease of work related to retrieving, storing, transforming and recognizing text in images gives rise to vivid practical applications of OCR. To name a few, any text in a scene can be captured as an image and read, old ancient books can be translated and made audible. These type of works demand high accuracy and speed. Also, the characters to be recognized usually have multiple complications like blur, inclined, curved, reflected …etc.

The era of machine learning and deep neural networks have replaced the traditional methods in the OCR domain. The research work by Meenu *et al.*[1] bought a two phased ensemble technique in which input to the neural network classifier and SVM classifier were SURF and curvature features of Malayalam text. For the first phase characters were used giving 89.2% accuracy while sentences were used for the second phase which gave 81.1% accuracy. Also, Chris *et al.* [2] presented a patch build up classification framework for Arabic font using ResNet pre-trained neural architectures with state of art performances. Later, Shah *et al.* [3] used Alex net convolutional neural network architectures to demonstrate triplet network and Siamese network deep metric learning for Urdu character recognition. Tan *et al.*[4] proposed pre-trained deep neural network's Inception V3 model with 2 fully connected layers giving 90% accuracy and 21.5% reduced error rate for broken and faded English characters. Yet, Asghar *et al.*[5] worked on cursive text recognition with CNN, in which different mixed filter sizes is tested instead of unique filter sizes for convolutional layers and stride values. Leena *et al.* [6] worked on Arabic text classification using deep neural networks where in among the models LSTM gave best accuracy. There is enormous work done for recognition of handwritten, digital and scene text for languages like English, Arabic, Korean, Urdu etc.

Looking at related work for kannada specifically, a novel work by Murthy *et al.* [7] is done for kannada speech detection and evaluation through a smart online app. Also, for the first time spoken kannada word recognition for lip reading is done by Nandini *et al.* [8]. Kumar and Ramakrishnan [9] worked on recognition techniques for degraded and split printed multi-lingual characters in documents. Similarly, Rajput and Ummapure [10] came up with multi-lingual handwritten script recognition using LBP operator and SVM classifiers. Asha and Krishnappa [11] proposed a technique using CNN for handwritten kannada character recognition achieving considerable results. These recent efforts are not witnessed for Kannada scene text recognition. This paper is dedicated to find the best suitable CNN models for Kannada character recognition of Chars74K dataset with the maximum accuracy and minimum error rate. Also, to test the performance of Alex net on inserting batch normalization layer to it. This paper is organized as follows. Section II describes the methodology engaged while Section III demonstrates the experimental results. In Section IV, Conclusion with prospective areas of improving the method is conferred.

II. Methodology

A. Chars74K CNN Classification

The working model is inspired from Tan Chiang Wei's OCR model [4] for English text to lessen training period and

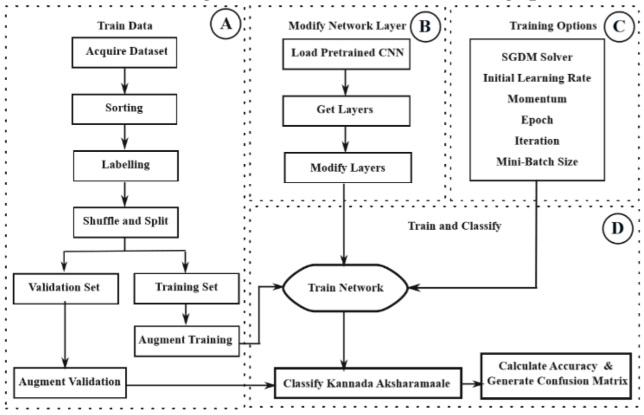


Fig. 1. The proposed Model

progress recognition accuracy. With the same purpose the idea of transfer learning for Kannada text classification is implemented. The proposed OCR system is developed as per the modules shown in the flow diagram in Fig. 1 below. It consists of four stages, namely train data, modify network layers, exercise algorithm options and train and classify.

1) Train Data: Here the data is getting ready to be trained. This phase is further simplfied as follows:

a) *Input Data Acquisition:* The work is aimed to identify Kannada characters in scene images. For this the standard dataset Chars74K [12] is used. Characters belonging to Kannada Aksharamaale only is extracted, ignoring Kaagunitaaksharas and Ottaksharas from the dataset.

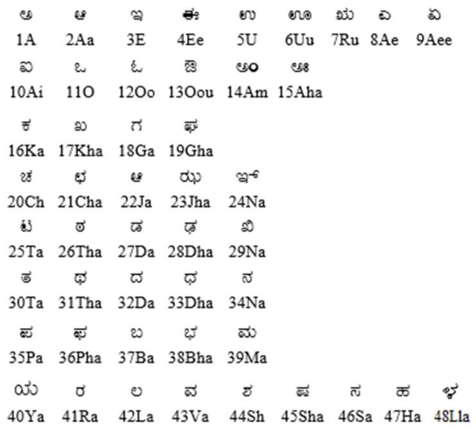
b) *Sort and Label*: The sequential labelling of the folder names to character names is done to ease the coding. Otherwise individual character names should be labelled in the programming. The Kannada Aksharamaale has 51 characters out of which few are used sparingly. However, 48 number of classes is created from the extracted images, labelled with their character names comprising of 1700 files for the classification. The detailed labelling of individual character is listed in Fig. 2.

c) *Shuffle and Split the Data Store:* An image data store object is created to import and process the collection. The dataset is shuffled before randomly splitting for training and testing purpose. Different split combinations like 60-40%, 70-30%, 80-20% 95-05% and 97-03% are employed as training-validation dataset. Their results are analysed in Table I and II.

d) *Data Store Augment:* The individual input images are of variable sizes and extremely small to be fed to the layers of deep network. The smallest being 1KB (15x15) and the largest 233KB (354x326). A pre-processing for resizing of images to the size of input layer of the model is done by an augmented image data store scaling transform. Fig. 3. Shows some random samples from the training dataset. Now the data is ready to be trained and validated.

2) Modify Network Layers

The basic building blocks of convolutional neural networks contains convolutional layers, pooling layers and fully connected layers. Each CNN model varies in depth of layers, size and number of filters in convolutions, number of convolutions, type and style of pooling applied, activation function used, sequential or non-sequential networks, etc.



ಅ	ಆ	ಇ	ಈ	ಉ	ಊ	ಋ	ಎ	ಏ
1A	2Aa	3E	4Ee	5U	6Uu	7Ru	8Ae	9Aee
ಐ	ಒ	ಓ	ಔ	ಅಂ	ಅಃ			
10Ai	11O	12Oo	13Oou	14Am	15Aha			
ಕ	ಖ	ಗ	ಘ					
16Ka	17Kha	18Ga	19Gha					
ಚ	ಛ	ಜ	ಝ	ಞ				
20Ch	21Cha	22Ja	23Jha	24Na				
ಟ	ಠ	ಡ	ಢ	ಣ				
25Ta	26Tha	27Da	28Dha	29Na				
ತ	ಥ	ದ	ಧ	ನ				
30Ta	31Tha	32Da	33Dha	34Na				
ಪ	ಫ	ಬ	ಭ	ಮ				
35Pa	36Pha	37Ba	38Bha	39Ma				
ಯ	ರ	ಲ	ವ	ಶ	ಷ	ಸ	ಹ	ಳ
40Ya	41Ra	42La	43Va	44Sh	45Sha	46Sa	47Ha	48Lla

Fig. 2. Labelled folder names for the used Kannada characters

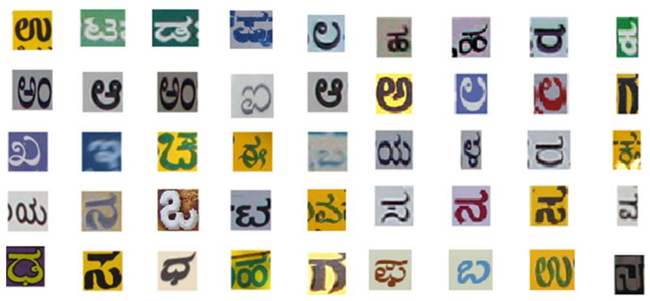


Fig. 3. Random samples from the training dataset

The models used in this work are briefed below:

a) *Alex Net*[13]*:* This neural network, named after Alex Krizhevsky, is a sequential network of 11x11, 5x5, 3x3, convolutions, max pooling, dropout, ReLU activations and SGD with momentum. It use ReLU activation function, which showed improved training performance over tanh and sigmoid. It has 60 million parameters and 650,000 neurons.

b) *Google Net*[14]*:* It used batch normalization, image distortions and RMSprop. This module is based on several very small convolutions in order to drastically reduce the number of parameters. Their architecture consisted of a 22 layer deep CNN but reduced the number of parameters from 60 million (Alex Net) to 4 million.

c) *InceptionV3*[15]*:* It is the extended version of Google net with 48 layers deep networks with high efficiency.

d) *ResNet50*[16]*:*Residual Neural Network (ResNet) is an effective neural network. It introduced skip connections to jump over layers, allowing the flow of data from initial layers to last layers. Skipping avoids the problem of vanishing gradients, by reusing activations from a previous layer until the adjacent layer learns its weights.

e) *Squeeze Net*[17]: It is 18 layers deep and can classify up to 1000 classes. It has replaced 3x3 filters with 1x1 filters which reduces the parameters while preserving accuracy.

f) *VGG-19*[18]*:* VGG19 consists of 19 weighted convolutional layers and is very pleasing because of its uniform simple design. Similar to AlexNet, only 3x3 convolutions, but lots of filters. Its downside is it consists of 138 million parameters to train.

The image input size of each of these models is known and changes to the data image size is scaled accordingly. The last fully connected layer is replaced to the number of classes, which is 48 in this case.

3) Exercise Algorithm Options

A set of training options is demonstrated, using stochastic gradient descent with momentum as a solver that updates the weights and biases parameters of the network. These parameters lessen the loss function by taking tiny steps at each iteration in the direction of the negative gradient of the loss given by,

$$\theta_{\ell+1}=\theta_{\ell}-\alpha\nabla E\,(\theta_{\ell})+\gamma\,(\theta_{\ell}-\theta_{\ell-1}) \qquad (1)$$

where ℓ is the iteration number, $\alpha > 0$ is the learning rate, θ is the parameter vector, $E\ (\theta)$ is the loss function and *γ* determines the contribution of the previous gradient step to the current iteration. The gradient of the loss function, $\nabla E(\theta)$, is evaluated using the entire training set, and the SGD algorithm uses the entire data set at once.

At each iteration the algorithm evaluates the gradient and updates the parameters using a different subset of the training data called a mini-batch. Here the mini-batch is assigned to 128. The full pass of the training algorithm over the entire training set using mini-batches is one *epoch*. The maximum number of epoch for each training is committed to 30. The momentum reduces the oscillation along the path of steepest descent towards the optimum. Momentum with 0.90 and 0.95 is tested. In both the cases the results are nearly same. Since 0.90 of momentum gave highest accuracy, the same is retained. Next, the learning rate for transfer learning with default 0.01 and 0.001 is also tested and analyzed that 0.001 gave better accuracy. These cases are reflected in Table I and II.

4) Train and Classify

Finally the network is trained with augmented training images, the modified network layers and the above mentioned training options. The resultant fine-tuned network is applied to classify the validation images. The classification accuracy on the validation set is calculated. It is the fraction of labels that the network predicts correctly. Details pertaining to accuracy performances are discussed in the experimental results section.

B. Batch Normalization with Alex net

Inspired by the elucidation by Brownlee [19] we made an attempt to insert batch normalization layer in the alexnet model. The suggestions in the examples of [20] to add batch normalization after the ReLU activation function at the 15th layer of the model is implemented. This establishes stable comportment of the gradients thus fastening the learning.

III. Experimental Results

From the available standard dataset Chars74k, data for kannada character level recognition is extracted and trained. The classification is executed on a single i7-CPU, 16GB RAM, 64bit OS with MatLabR2019a and Deep Learning Toolbox. The data consisting of 1700 images as 48 classes is trained on six neural network models, listed in column one of Table I. The parameters maintained with constant value are: initial learning rate as 0.001, momentum as 0.9 and mini-batch to 128. The analysis is done based on the size of data trained and weight and bias learning rate factor. The ratio of data was split to 70, 80, 90, 95 and 97 for training, and weight and bias learning rate factor for 10 and 20. Table I gives in-depth analysis for few combinations. It is observed that Vgg19 gives absolute accuracy taking ample amount of time while Alex net is the fastest but average with result.

The results for the modified Alex net model is depicted in Table II. Profound trials were demonstrated keeping the same training options and values, just as the previous section analysis mentioned above. It is observed that highest recognition rate is 96.4%, when the training and testing images are randomly distributed to 95% to 5% respectively. Also, this performance gives 100% mini batch accuracy and error rate with a minimum of 0.0134. It is also observed that the accuracy rate and iterations column is increased exponentially on decrease in testing input count which means that higher the training ratio better the results. Also, whenever momentum is increased the time taken for classification automatically increases.

During the training the plots visualized the progress metrics at each iteration. The Fig. 4 and 5 plots the *training accuracy/loss* on each individual mini-batch, *smoothed training accuracy/loss* which is applied on the training accuracy making the spot trends boldly visible and *validate accuracy/loss* on the entire validation set. The final validation metrics are labelled *Final* in the plots. The predictions are visualized with the help of a confusion matrix. A sample confusion matrix chart tested against Kannada vowels dataset only is shown below in Fig. 6. In the figure the rows indicate the true class i.e. actual case while the columns indicate predicted class or recognized-as case. The diagonal cells in the chart corresponds to correctly classified cases while off-diagonal cells represent incorrect observations. The row and column summaries displays the number of accurate and inaccurate classifications for each true and predicted classes accordingly as percentages of the number of observations of the corresponding true and predicted classes respectively. In Fig. 7, random sample validation images with their predicted labels are shown. It is observed that the predicted labels are accurate to their character names.

A comparison table showing accuracy for other existing recognition methods for the standard dataset chars74k is listed in Table III. The charts clearly say that the proposed method yields better results.

TABLE I. Performances 0n CNN models with chars74k

CNN models	Ratio of training-validation dataset					
	70-30		95-5		97-3	
	A	Time	A	Time	A	Time
Alex net	53.66	1.5m	50.00	3m	57.14	**3m**
Google net	65.85	8m	71.43	15m	85.71	15m
Inceptionv3	39.02	33m	78.57	59m	85.71	65m
Resnet50	51.22	24m	71.43	487m	57.14	43m
Squeezenet	50.00	4m	64.29	8m	71.43	7m
Vgg19	58.54	36m	78.57	73m	**100.00**	75m

TABLE II. Performances 0n Modified Alexnet model

split ratio	**Momentum**	**mini-batch accuracy**	**mini-batch loss**	**Validation Accuracy**
60-40	0.90	99	0.0443	75.5
60-40	0.95	100	0.0195	74.8
80-20	0.90	98	0.0308	81.4
80-20	0.95	100	0.0109	84.0
95-05	0.90	100	0.0134	96.4
95-05	0.95	99	0.0297	91.6

TABLE III. Comparison for chars74k

Method	**Accuracy %**
De Campos *et al.* [21]	54.3
Yao *et al.*[22]	75.9
Matias *et al.*[23]	99.0
Michael *et al.*[24]	94.4
Proposed method	98.4

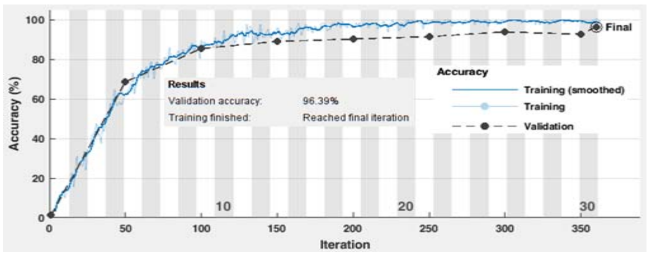


Fig. 4. Training progress for Accuracy

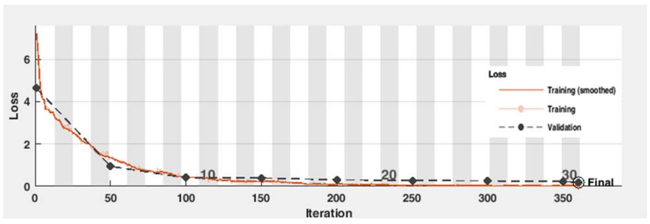


Fig. 5. Training progress for Error rate

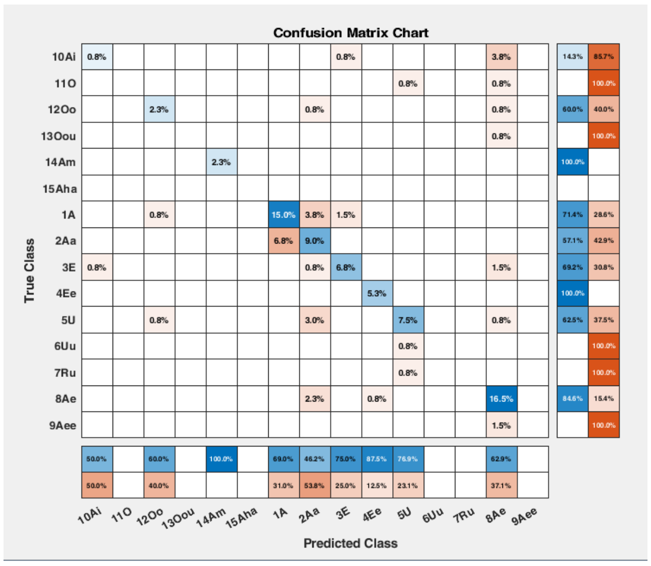


Fig. 6. Sample Confusion Matrix Chart for Kannada Vowels

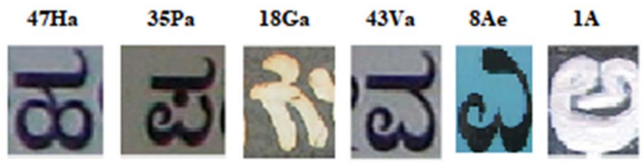


Fig. 7. Random validation samples with their predicted labels

IV. Conclusion

This paper propositioned on an OCR system for the classification of scene Kannada characters. It is achieved by building a transfer learning based OCR exercising a pre-trained deep neural network. Alex net is modified by inserting a batch normalization layer in between ReLU and Maxpooling layers. The proposed network is trained and tested using Kannada scene characters (vowels and consonants only) extracted and labelled from Chars74K Standard Dataset. The results show that the system achieved significant recognition accuracy of 96% and reduction in error rate with 0.0134. Vgg19 gave 100% accuracy for Chars74K. The method works well for all font styles with complications in image clarity and quality. The work can be extended to Kaagunitaakshara and Ottakshara categories of Kannada character set. The sample images in the standard dataset needs to be increased and all the folders should have even count of samples to justify the accurate performances for recognition. Also, enhancing the training with acute testing options will improve the results.

References

[1] M. Alex and S. Das, “An Approach towards Malayalam Handwriting Recognition Using Dissimilar Classifiers,” *Procedia Technol.*, vol. 25, no. Raerest, 2016, pp. 224–231.

[2] C. Tensmeyer, D. Saunders, and T. Martinez, “Convolutional Neural Networks for Font Classification,” 2017.

[3] S. Nawaz, A. Calefati, N. Ahmed, and I. Gallo, “Hand Written Characters Recognition via Deep Metric Learning Hand Written Characters Recognition via Deep Metric Learning,” no. April, 2018.

[4] T. C. Wei, U. U. Sheikh and A. A. A. Rahman, "Improved optical character recognition with deep neural network," *2018 IEEE 14th International Colloquium on Signal Processing & Its Applications (CSPA)*, Batu Feringghi, 2018,pp. 245-249. doi: 10.1109/CSPA.2018.8368720

[5] A. Ali, M. Pickering and K. Shafi, "Urdu Natural Scene Character Recognition using Convolutional Neural Networks," *2018 IEEE 2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR)*, London, 2018, pp. 29-34. doi: 10.1109/ASAR.2018.8480202

[6] L. Lulu and A. Elnagar, “Automatic Arabic Dialect Classification Using Deep Learning Models,” *Procedia Comput. Sci.*, vol. 142, 2018, pp. 262–269.

[7] S. Murthy *et al.*, "Kannada Kali: A Smartphone Application for Evaluating Spoken Kannada Words and Detecting Mispronunciations Using Self Organizing Maps," *2018 IEEE Tenth International Conference on Technology for Education (T4E)*, Chennai, 2018, pp. 1-7. doi: 10.1109/T4E.2018.00009

[8] M. S. Nandini, T. C. Nagavi and N. U. Bhajantri, "Deep Weighted Feature Descriptors for Lip Reading of Kannada Language," *2019 6th International Conference on Signal Processing and Integrated Networks (SPIN)*, Noida, India, 2019, pp. 978-982. doi: 10.1109/SPIN.2019.8711730

[9] H. R. S. Kumar and A. G. Ramakrishnan, "Gamma Enhanced Binarization - An Adaptive Nonlinear Enhancement of Degraded Word Images for Improved Recognition of Split Characters," *2019 National Conference on Communications (NCC)*, Bangalore, India, 2019, pp. 1-6. doi: 10.1109/NCC.2019.8732254

[10] G. G. Rajput and S. B. Ummapure, "Script Identification from Handwritten document Images Using LBP Technique at Block level," *2019 International Conference on Data Science and Communication (IconDSC)*, Bangalore, India, 2019, pp. 1-6. doi: 10.1109/IconDSC.2019.8816944

[11] K. Asha and H. K. Krishnappa, “Kannada Handwritten Document Recognition using Convolutional Neural Network,” *2018 3rd Int. Conf. Comput. Syst. Inf. Technol. Sustain. Solut.*, 2018, pp. 299–301.

[12] [Online]. Available: http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/

[13] A. Krizhevsky and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” *2012 Neural Inf. Process. Syst.* pp. 1–9. doi: 10.1145/3065386.

[14] C. Szegedy *et al.*, "Going deeper with convolutions," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, 2015, pp. 1-9. doi: 10.1109/CVPR.2015.7298594

[15] C. Szegedy, V. Vanhoucke, and J. Shlens, “Rethinking the Inception Architecture for Computer Vision,” 2016 *Conference on Computer Vision and Pattern Recognition*. doi: 10.1109/CVPR.2016.308.

[16] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, 2016, pp. 770-778. doi: 10.1109/CVPR.2016.90

[17] F. N. Iandola, S. Han, W. J. Dally, and K. Keutzer, “SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size,” 2016 February, pp. 0–13.

[18] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” 2015, pp. 1–14.

[19] J. Brownlee, “How to Accelerate Learning of Deep Neural Networks With Batch Normalization.” [Online]. Available: https://machinelearningmastery.com/how-to-accelerate-learning-of-deep-neural-networks-with-batch-normalization/.

[20] “Batch normalization and activation function.” [Online]. Available:https://github.com/ducha-aiki/caffenet benchmark/blob/master/batchnorm.md.

[21] T. E. De Campos, B. R. Babu and M. Varma, “Character recognition in natural images,” 2009 *VISAPP (2)*.

[22] C. Yao, X. Bai and W. Liu, "A Unified Framework for Multioriented Text Detection and Recognition," in *IEEE Transactions on Image Processing*, vol. 23, no. 11, Nov. 2014, pp. 4737-4749. doi: 10.1109/TIP.2014.2353813

[23] M. Valdenegro-Toro, P. Plöger, S. Eickeler and I. Konya, “Histograms of Stroke Widths for Multi-script Text Detection and Verification in Road Scenes,” 2016 *IFAC-PapersOnLine*, *49*(15), pp. 100–107. [Online]. Available: https://doi.org/10.1016/j.ifacol.2016.07.716

[24] M. R. Phangtriastu, J. Harefa, and D. F. Tanoto, “Comparison between Neural Netwok and Support Vector Machine in Optical Character Recognition,” *Procedia Comput. Sci.*, vol. 116, 2017, pp. 351–357.