DocDescribor: Digits + Alphabets + Math Symbols - A Complete OCR for Handwritten Documents

Ridhi Aggarwal[1], Hiteshi Jain[1], Gaurav Harit[1(✉)], and Anil Kumar Tiwari[2]

[1] Department of Computer Science and Engineering, Indian Institute of Technology Jodhpur, Jodhpur, Rajasthan, India
{pg201384012,jain.4,gharit,akt}@iitj.ac.in
[2] Department of Electrical Engineering, Indian Institute of Technology Jodhpur, Jodhpur, Rajasthan, India

Abstract. This paper presents an Optical Character Recognition (OCR) system for documents with English text and mathematical expressions. Neural network architectures using CNN layers and/or dense layers achieve high level accuracy in character recognition task. However, these models require large amount of data to train the network, with balanced number of samples for each class. Recognition of mathematical symbols poses challenges of the imbalance and paucity of training data available. To address this issue, we pose the character recognition problem as a Distance Metric Learning problem. We propose a Siamese-CNN Network that learns discriminative features to identify if the two images in a pair contain similar or dissimilar characters. The network is then used to recognize different characters by character matching where test images are compared to sample images of any target class which may or may not be included during training. Thus our model can scale to new symbols easily. The proposed approach is invariant to author's handwriting. Our model has been tested over images extracted from a dataset of scanned answer scripts collected by us. It is seen that our approach achieves comparable performance to other architectures using convolutional layers or dense layers while using lesser training data.

Keywords: Distance metric learning · Ensemble methods · Siamese network · Mathematical symbols · Characters · Digit · Handwritten documents

1 Introduction

Optical Character Recognition (OCR) is widely used in many applications such as signature verification, cheque processing, form processing, converting scanned documents to machine-text form, etc. Handwritten documents can consist of character lines, digits, mathematical expressions and figures. In this work we

R. V. Babu et al. (Eds.): NCVPRIPG 2019, CCIS 1249, pp. 292–301, 2020.
https://doi.org/10.1007/978-981-15-8697-2_27

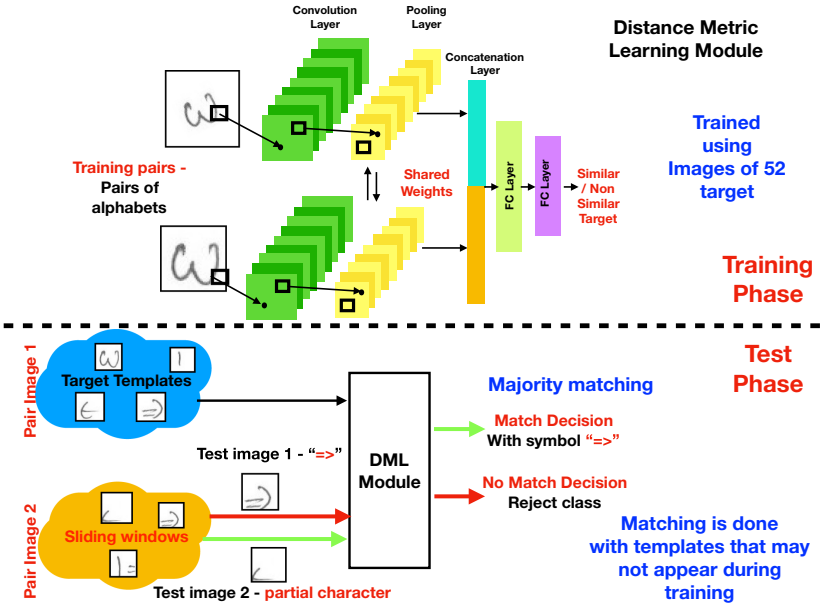

Fig. 1. Alpha-symbol recognition framework

focus towards recognizing characters, digits and symbols from a handwritten document.

The integrated OCR for printed [15] and online handwritten mathematical documents [11] are already well developed. In contrast, the task of offline handwritten mathematical symbol recognition is relatively unexplored. Offline formula recognition and structure parsing of isolated expressions is recently addressed in the Competition on Recognition of Online Handwritten Mathematical Expressions (CROHME) 2019 [8]. However, the task of recognizing digits and math symbols in a complete handwritten document image has not been addressed in the past.

The design of an OCR for documents involves three steps: 1) segmentation step that involves isolating the characters from each other, 2) designing a classification system for possible target classes and a decision function that allows assigning a character image to predefined class 3) finally, the verification strategy that uses the global decision module and allows the rejection or acceptance of the processed image.

Handwritten documents pose various challenges like touching characters, non-uniform sized words and slanted/unaligned writing patterns unlike printed documents that have non-touching individual characters mostly uniform in size. Thus segmentation of a handwritten document is a difficult task and a sliding window mostly used for segmentation can result into partial characters, twin characters or whole characters, as shown in Fig. 2 for one of our document

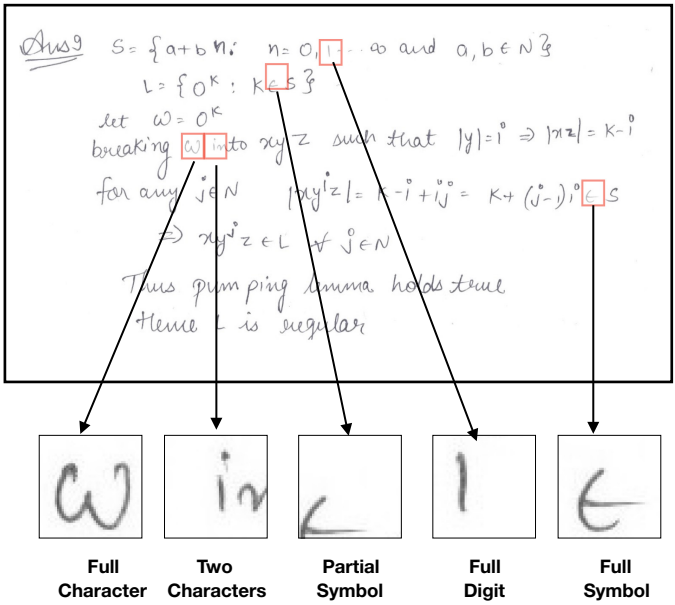

Fig. 2. Sliding windows of a document image highlighting various possible different ways of characters present in a window

images. In this work we address the classification steps and the decision making step.

Neural network architectures using convolutional layers (CNN)/dense layers have emerged as a replacement of traditional techniques of handwritten character recognition that used hand-crafted features [4,5,9,12,14]. These networks can extract visual features from images using different filters such that the learned features are invariant to shift, size and noise.

Training a good model requires a large training dataset with equal number of samples for all target classes. Further, prediction of these models is limited to classes used during training and learning a new category requires retraining the models.

Building a character recognition model to identify mathematical symbols like $\leq, \in, \sum$ in presence of alphabets and digits faces a problem of *Dataset Imbalance*. Math symbols such as $\leq, \sum, \in$ or greek alphabets like α, β rarely occur in documents. Thus there are too few samples of such characters in the training data as compared to alphabets and digits. As an example, Kaggle dataset contains 2692 samples of $\sum$ and 11197 samples for letter a. This dataset imbalance leads to a alphabet/digit biased training.

To solve the problem of dataset imbalance, image transformation techniques can be used that can help increase the samples of rarely occurring characters in the dataset. Rotation, translation, scaling and shearing are few of such transformations used in past to increase the dataset size [1,3,6,7].

In the decision step, a test image that does not belong to any of the trained target class is classified as a novel class/reject-class/anomaly based on a probability threshold i.e. if for a test image the probability of recognition is less than a threshold the image is classified as a reject class [2,10]. The difficulty here remains - how to choose the right threshold?

Few questions arise: Can we design a classification framework that can leverage only few target class labels and use this to identify new unseen (not seen during training) but known target character classes? Further, can a framework be designed to make a decision of the target class without using a threshold?

A recent paper [13] has addressed this problem and proposed a Siamese-CNN model that uses the contrastive loss to maximize Euclidean distance between the representations of dissimilar pairs and minimize the distance between similar ones. In contrast, we pose the similarity learning task with match/mismatch targets. This allows the CNN layers to learn representations of inputs such that the fully connected layer can learn to map similar/dissimilar inputs to match/mismatch targets. Unlike [13], our network is able to learn the distance metric without the use of Euclidean distance between learned representations.

Our Alpha-Symbol Recognition framework (Fig. 1) works in two phases: 1) Distance Metric Learning between two characters using Siamese-CNN network. This module learns to identify if two images in a pair are similar or dissimilar. This module is trained using image pairs taken from alphabets only. 2) Decision module that evaluates a match/non-match decision using the learned distance metric learning (DML) module where one image in the pair is the sliding window for which the target label needs to be evaluated and the other image is a candidate template of a candidate target class. These target classes may or may not be included in the training phase.

Our model is seen to overcome the problem of dataset imbalance as it has only been trained with the balanced Kaggle alphabet dataset and does not require re-training of the network for new symbols. Further, the decision of rejection is based on a match-mismatch decision of DML module and does not depend on any threshold. The framework has been tested for sliding windows extracted from examination answer scripts that have been collected by us. It is seen that our approach gives comparable or better performance compared to alternative architectures trained to do classification using dense, convolutional or Gated-CNN layers, with softmax classification layer.

2 Alpha-Symbol Recognition Framework

To recognize alphabets and mathematical symbols, we pose the recognition problem as Distance Metric Learning problem. As a first step, the Siamese-CNN Network is designed to identify if the pair of images containing characters are similar or dissimilar. Once the network learns to identify the similarity and dissimilarity between two images through distance metric learning, we use this network to compare the test sliding windows with images of known symbols to predict the label for the image. We explain the two steps in detail here:

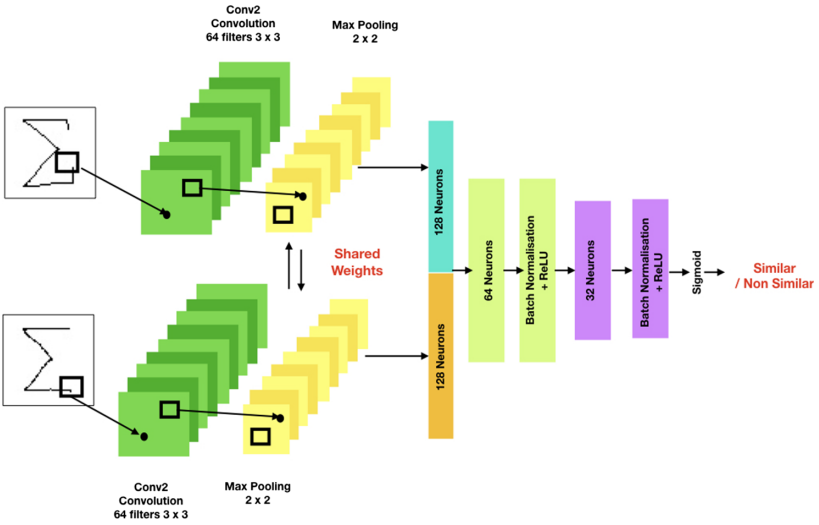

Fig. 3. Siamese network for similarity metric learning and identify input pair as match/mismatch

2.1 Distance Metric Learning Module

The goal of this module is to learn if two images are similar or dissimilar. For this we propose a Siamese-CNN Network that takes symbol image pairs as input and learns similarity between them. The network is as shown in Fig. 3. The input images are individually passed to separate but similar networks comprising a single layer CNN with 64 3×3 filters and a 128 unit dense layer. The weights of the two individual branches are shared. Thus the two CNN networks take an image pair as input and produce two output vectors O_p and $O_q \in R_N$ (here $N = 128$). The outputs are concatenated and fed to a Multi Layer Perceptron with a final sigmoid layer to give 0/1 output label, i.e. match/non-match label.

2.2 Label Prediction for Sliding Windows

Till now, the Similarity-CNN Network has been able to mark the input pairs as similar and dissimilar. The learned metric of image similarity is used to find the matching labelled images. The label of the matched image is adopted as the label of the test image. As an input to the learned Similarity-CNN network, we give the test image as one input to the network and a template with a known label as the other input. One by one we compute the similarity of the test image with the template images of the target classes. The class for which there are maximum number of matched templates with the sliding window is assigned as the label to the window. If there is no match found from the template images, the sliding window is said to belong to a reject-class.

This technique of Distance Metric Learning for handwritten character recognition provides multiple advantages:

1. If a test image is a partial character or a touching character, i.e. belongs to the reject class, the Similarity-CNN Network predicts the reject class label without any threshold requirements.
2. Such a network provides an advantage that it can be trained with a few classes and during test time it can be used to find similarity with even the templates belonging to new labels i.e. symbols not seen during training. The reason for this being that the network has learned to discriminate if a pair is similar or dissimilar. Thus, irrespective of the type of input images provided at the test time it can identify if the input images match or not and thus provide an appropriate label. The network now is not required to be retrained every time for a new character type. Further, it can be trained with a few symbol classes only, like the Kaggle alphabet dataset. Use of datasets like Kaggle or MNIST for training the Siamese makes the system learn features which are invariant to handwriting.

In the next section we discuss the experimental settings and results where we show the capabilities of our model and compare it with a few character recognition methods.

3 Experiments

Our task is to identify the different characters, digits, and mathematical symbols that are there in the examination scripts collected by us. Character hypotheses are generated with a sliding window approach. The sliding window approach comes with the inherent drawback of the windows containing partial as well as touching characters which need to be classified by the network to a reject class.

The documents that we consider contains 106 character types including 10 digits, 26 alphabets (uppercase and lowercase) and mathematical symbols like $\alpha, \beta, \sum$, etc. Some of the sample sliding windows are as shown in Fig. 4.

We consider the following baselines for comparison.

1. Dense Neural Network: In the dense neural networks (Fig. 5), the input images are flattened and passed to a single dense layer (128 neuron) network followed by a dropout layer with drop probability of 0.2. The final layer is the softmax layer used for multi-class classification.

Fig. 4. Sliding windows of the documents with sample partial and full characters

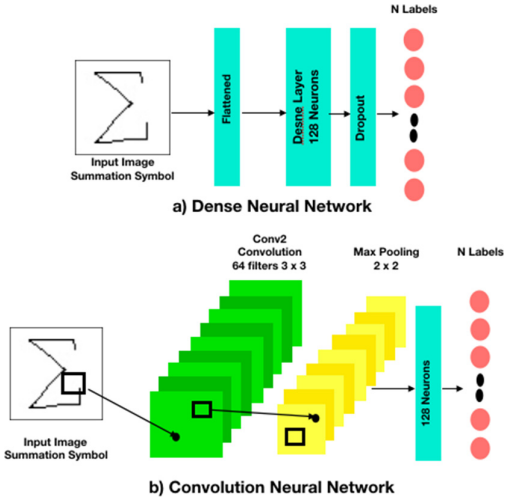

Fig. 5. Dense Neural Network and Convolution Neural Network (CNN) architectures used for character recognition

2. Convolution Neural Network: A popular technique to use deep learning to classify images is to build a convolutional neural network (CNN). A convolution multiplies a matrix of pixels with a filter matrix or kernel and sums up the multiplication values. Then the convolution slides over to the next pixel and repeats the same process until all the image pixels have been covered. We use 64 kernels of size 3×3 followed by Max-Pooling with filters of size 2×2 to reduce the image output size. Finally the images are flattened and passed as an input to the 128 dimension dense layer. The output of the layer is passed to the softmax layer for the final multi-class classification (Fig. 5).
3. Gated Convolution Neural Network: In the gated convolution network, 2-dimensional convolution layer is used. We use 32 kernels of size 3×3 followed by gated convolution block. Then Max-Pooling with filters of size 2×2 is used followed by dropout layer with drop probability 0.25. It is flattened and passed to a single dense layer (128 neurons) network followed again by dropout layer with drop probability 0.5 and single dense layer of 106 neurons. The final layer is softmax layer used for multi-class classification.

For training the 3 architectures, we used the Kaggle dataset of alphabets, digits, and mathematical symbols. Due to the imbalance in the dataset, the networks were seen to under-perform. So we used the SMOTE technique for dataset augmentation to increase the samples of characters that were rarely occurring in the dataset.

For our framework we used Kaggle character dataset only for training. The positive and negative pairs were generated based on similar/dissimilar characters in the images. Table 1 gives the details of the count of samples used for training and testing. The network has been tested for 52 document images that have been

Table 1. Number of Samples used for training

Comparison techniques	Number of training samples	
	Before augmentation	After augmentation
CNN, DenseNN, GCNN	Variable samples Class with min sample = 21 Class with max samples = 33997 Total Samples = 639113	2500 samples for each class (Alphabets, Symbols, Digits) Total samples = 265000 (Balanced dataset)
Siamese-CNN Model	2500 samples of digits and alphabets. (Balanced dataset, but symbols were not used for training)	

Table 2. Accuracy for isolated instances of Symbols, Characters and Digits. Note that the last row shows results when symbols were not used for training

Model	Train	Test Accuracy (%)					
		Alphabets		**Symbols**		**Digits**	
	Train data	Before augment	After augment	Before augment	After augment	Before augment	After augment
DenseNN	Digits + Alphabets + Symbols	78.9	89.6	66	81	81.8	78
CNN	Digits + Alphabets + Symbols	81.2	84.3	67.55	84.9	81.8	80.8
GCNN	Digits + Alphabets + Symbols	88.7	88.9	68.55	84.9	81.8	81.8
[13]	Digits	54.72		29.77		82.6	
Proposed	Digits	56.8		32.2		90	
	Alphabets	76		45.8		78.9	
	Digits + Alphabets	86.9		61.7		88.6	

collected by us from answer scripts of students. A sample image of the document is shown in Fig. 2. The accuracy of these models on isolated digits, symbols and alphabets recognition are as shown in Table 2.

The recognition accuracy, averaged over all classes, of the different architectures for classifying sliding windows in test documents are as shown in Table 3. A candidate window was classified to the reject class if the highest probability attributed to a class by the softmax classification layer was below a threshold. For the proposed technique, a candidate window was classified to the reject class if it could not match with enough (majority) number of templates of any of the classes.

From both Table 2 and Table 3, we can see that though our model does not outperform the models like CNN, Dense-NN and GCNN, however, with limited training data and without the use of instances from symbol classes for training it performs close to these models.

Table 3. Recognition accuracy (%) of different models over test dataset of handwritten documents

	Before data augmentation	After data augmentation
Dense-NN	75.6	82.82
CNN	76.85	83.33
Gated CNN	79.66	85.2
Proposed	79.06	

4 Conclusion and Discussion

In this paper we have introduced an OCR system to recognize math symbols, along with alphabets and digits, for a handwritten document image. We use a distance metric learning approach to learn discriminative features such that two similar characters are drawn nearer to each other in the representation space and the dissimilar ones are moved apart. Such a network can be trained on fewer classes and tested for unseen character recognition using match or no-match decision. The network has been trained for digits and characters and still can be used to identify symbols. This helps in handling the class imbalance problem that occurs because of few training samples available for rarely used symbols. The network is seen to perform comparable to the traditional neural network architectures using dense layers, convolutional layers, or gated convolutional layers, with a softmax classification layer. The classification stage takes time because a candidate window needs to be compared for match/no-match with a number of templates belonging to all classes. However, retraining of the model to classify a new symbol gets avoided. A drawback of sliding windows technique is that some sliding windows containing a partial character or two touching characters do not get classified into the reject class. Common problems, e.g., are, a window with a partial alphabet 'B' matches to digit '3', partial alphabets 'D', 'L', 'M', 'N', 'P' often match to digit '1', partial symbols 'α', 'X', 'K', match to 'a', 'Y', '$<$', respectively. Future work would consider refining the possible misclassifications by using language models.

References

1. Brown, W.M., Gedeon, T.D., Groves, D.I.: Use of noise to augment training data: a neural network method of mineral-potential mapping in regions of limited known deposit examples. Nat. Resour. Res. **12**(2), 141–152 (2003). https://doi.org/10.1023/A:1024218913435
2. Chalapathy, R., Menon, A.K., Chawla, S.: Anomaly detection using one-class neural networks. arXiv preprint arXiv:1802.06360 (2018)
3. Cireşan, D.C., Meier, U., Masci, J., Gambardella, L.M., Schmidhuber, J.: High-performance neural networks for visual object classification. arXiv preprint arXiv:1102.0183 (2011)

4. Gunawan, T.S., Noor, A.F.R.M., Kartiwi, M.: Development of English handwritten recognition using deep neural network. Indonesian J. Electric. Eng. Comput. Sci. **10**(2), 562–568 (2018)
5. He, W., et al: Context-aware mathematical expression recognition: an end-to-end framework and a benchmark. In: 2016 23rd International Conference on Pattern Recognition (ICPR), pp. 3246–3251. IEEE (2016)
6. LeCun, Y.: The mnist database of handwritten digits. http://yann.lecun.com/exdb/mnist/ (1998)
7. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., et al.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
8. Mahdavi, M., Zanibbi, R., Mouchere, H., Garain, U.: ICDAR 2019 CROHME+ TFD: competition on recognition of handwritten mathematical expressions and typeset formula detection. In: Proceedings ICDAR (2019)
9. Maitra, D.S., Bhattacharya, U., Parui, S.K.: CNN based common approach to handwritten character recognition of multiple scripts. In: 2015 13th International Conference on Document Analysis and Recognition (ICDAR), pp. 1021–1025. IEEE (2015)
10. Marchi, E., Vesperini, F., Eyben, F., Squartini, S., Schuller, B.: A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional LSTM neural networks. In: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1996–2000. IEEE (2015)
11. Mouchère, H., Viard-Gaudin, C., Zanibbi, R., Garain, U.: ICFHR 2016 CROHME: competition on recognition of online handwritten mathematical expressions. In: 2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR), pp. 607–612. IEEE (2016)
12. Simard, P.Y., Steinkraus, D., Platt, J.C., et al.: Best practices for convolutional neural networks applied to visual document analysis. In: ICDAR, vol. 3 (2003)
13. Sokar, G., Hemayed, E.E., Rehan, M.: A generic OCR using deep siamese convolution neural networks. In: 2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), pp. 1238–1244. IEEE (2018)
14. Suryani, D., Doetsch, P., Ney, H.: On the benefits of convolutional neural network combinations in offline handwriting recognition. In: 2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR), pp. 193–198. IEEE (2016)
15. Suzuki, M., Tamari, F., Fukuda, R., Uchida, S., Kanahori, T.: INFTY: an integrated OCR system for mathematical documents. In: Proceedings of the 2003 ACM Symposium on Document Engineering, pp. 95–104. ACM (2003)