2018 International Conference on Computational Science and Computational Intelligence (CSCI)

Sequence-to-Sequence learning using Deep Learning for Optical Character Recognition (OCR)

Vishal Mishra
Department of EECS
The University of Toledo
Toledo,Ohio,USA
Email: Vishal.Mishra9602@gmail.com

Dr. Devinder Kaur
Department of EECS
The University of Toledo
Toledo,Ohio,USA
Email: Devinder.Kaur@utoledo.edu

***Abstract*— In this paper, Convolution Neural Network (CNN) and a special variant of Recurrent Neural Network (RNN) named Long Short-Term Memory Model (LSTM) with peep hole connection is developed for optical character recognition (OCR). Data-set of mathematical equations known as *Image to Latex 100K* is retrieved from OPEN-AI and used for testing the model. First, the mathematical equations from the images are converted to Latex texts. Then this Latex text is used to render the mathematical equations. The proposed method uses the tokenized data, which is sequentially given to the deep learning network.**

The sequential process helps the algorithms to keep track of the processed data and yield high accuracy. A new variant of LSTM called "LSTM with peephole connections" and Stochastic "Hard" Attention model was used. The performance of the proposed deep learning neural network is compared with INFTY (which uses no RNN) and WYGIWYS (which uses RNN). The proposed algorithm gives a better accuracy of 76% as compared to 74% achieved by WYGIWYS.

Keywords— **Convolutional Neural Network, Recurrent Neural Network, Long Short-Term Memory (LSTM) with peephole connections. IMAGE2LATEX 100K.**

I. INTRODUCTION

In modern times, printed paper data records, consisting of passport documents, invoices, bank statements, printouts of static-data, or any appropriate documentation are being stored in the form of digital copies. It is a common practice to digitize printed texts so that it can be electronically edited, searched and stored, and can be used for text mining. Optical Character Recognition (ORC) can be used to convert printed texts into a digital representation. In the 1900s, an early form of optical character recognition (OCR) was used in the technologies such as telegraphy and reading device for blind people. In 1914, Emanuel Goldberg invented a device that could read characters and translate them into standard telegraphic code [1]. In general, OCR is used to identify and read a natural language from an image and convert it into standard representation. In 1967, the research work of Anderson R.H, there has been a surge in interest for extracting patterns from images for representing them in markup form, which is a correct semantic representation of the images [2].

In the early 2000s, Andrew Kae and Erick Miller addressed the OCR problem in an efficient w ay w ith the computational power that existed during that time [3]. However, with the advancement in computational power both in hardware and software, a great deal of research interest has emerged in OCR. The availability of graphical processing units (GPUs) in hardware and the development of pattern recognition algorithms based on deep learning have given a thrust to the new algorithms of OCR based on convolutional neural networks (CNNs) and recurrent neural networks (RNNs) etc. [4].

In 2003, Fukuda and Tamari invent a system that takes in handwritten mathematical expression and converts it into TeX format [5]. However, the focus of this paper is to use an optical character recognition mechanism for an image of mathematical formulas, including Greek symbols, superscript and subscript conversion in markup form [5]. The effectiveness of this system can be measured by the combination of segmented characters with grammars of the underlying mathematical layout language.

In this paper, we have used a data-set obtained from OPENAI website, which contains an image of mathematical formulas. In this experiment, the deep learning techniques named CNN and LSTM with peephole connections have been used to convert the mathematical formulas into Latex representation. In this paper, a new variant of LSTM unit called LSTM with peephole connections and a Stochastic Hard Attention mechanism based encoder- decoder model for Image-to-Latex 100K data set [5] [6]. At present, this is the best machine translation system we have. This model comprises of multi-layered convolution neural network to obtain the features of an image with the attention-based recurrent neural network. In our case, we introduce one more layer of the multi-row recurrent neural network called LSTM with peephole connection in front of attention model, so that it should addresses the OCR problem. Image2Latex 100K Data-set.

II. IMAGE2LATEX 100K DATA-SET

Image-to-latex-100k data set contains 127,652 different mathematical equations along with their rendered pictures in PNG format. The mathematical formulas have been extracted from the Latex sources of papers available on the arXiv website (https://arxiv.org/). These latex sources of papers were parsed through the regular expressions in python to obtain the mathematical formulas. In this research, the size of the

978-1-7281-1360-9/18/$31.00 ©2018 IEEE
DOI 10.1109/CSCI.2018.00069

formulas is restricted in between 35 to 1024 characters. The regular expression generated 963,890 different latex formulas from the latex sources. Amongst these 900K, formulas only 300K formulas were chosen to pass through the KaTeX API to render the PDF files and only 100K formulas were used to compare the proposed model with the existing models like WYGIWYS. These PDFs were converted into PNG format and the size of each rendered image was 1654 2339 pixels. To improve the results, rendered images were cropped to 360 60 pixels. Once these images were cropped, they were divided into tokens to train the model and all the large size images with more than 175 tokens have been discarded. The Training batch size is set to 35 tokens because of the size limit of GPU memory.

III. Convolutional Neural Network

Convolutional Neural Networks (CNNs) fall under the purview of deep learning. They are specifically used for high dimensional data processing such as colored images, videos etc. CNNs are multilayer feedforward networks. Each neuron in the convolution layers performs a dot product of image pixels with a filter. Each convolution layer is followed by Rectified Linear Unit (ReLu) layer and pooling layer. ReLu is a nonlinear activation function, which is used to perform a transformation on the images. The dimensionality of the image is reduced as the computation moves forward in successive layers and it is achieved by pooling layer. The output of the pooling layer becomes the input for the next convolution layer. Figure 1 shows an illustration of Convolutional Neural Network [26].

For example, in Figure 1 an image of 32*32 size with 20 filters each of size 5*5 are used to extract features, which will produce 20 activation feature maps and it is forwarded to pooling layer. Then with a filter size of 5*5 in pooling layer and a stride of 1 pixel, the image reduces to 28*28. This reduced image is then forwarded to the convolution layer and pooling which will reduce the image to 14*14 and so on, until the image is reduced to dimension 1*1. CNN architectures are completely relied on four hyper-parameters such as **Filters**, **Pooling**, **Stride**, and **Padding** to give the optimal results.

A. Layers in a CNN

1) Convolutional Layer: A convolutional layer consists of set of filters. Filters are small spatially; however, it covers the depth of an input volume (for example, if filter size is 5*5*3 i.e. 5-pixel width and height and 3 is the depth of the image because of color channel). Filters are used to extract features from an image. In feature extracting process, filters are moved across the image with given strides and perform dot products with the entries of the filter and the input at any position. As the filter is moved across the input volume, it produces a 2-dimensional activation feature map for that filter. For instance, if there are 20 filters of size 3*3, then there will be 20 activation feature maps for each filter and each feature map shows the responses of the respective filters at every spatial position. So, the input to the next layer would be these activation feature maps (for example, in Figure 2 the size of the activation feature map is 4*4 and if there are 20 such activation feature maps, then the input to the next layer would be 4*4*20). In Figure 2, the image size is 5*5, filter size is 2*2, and stride of 1 pixel and the activation feature map is 4*4. The activation feature can be calculated with a formula i.e. (W-F)/S+1 where W is the size of the image, F is the filter size and S is the stride size. Calculation for a resulting activation feature map in Figure 2 is given by:

$$= (W - F)/S + 1$$
$$= (5 - 2)/1 + 1$$
$$= (3)/1 + 1$$
$$= 3 + 1$$
$$= 4$$

Now, we repeat this process for every location on the input volume. Every unique location on the input volume produces a number. After sliding the filter over all the locations, we are left with a two-dimensional array, which is called an activation map, or feature map.

Calculation of first convolution is computed by moving a filter with given stride across the image pixels and perform the dot product to get the activation feature map (for example, first pixel of image is multiplied to the first pixel of the filter (0*1)). Calculation of first convolution as follows [14].

$$= (0 * 1) + (1 * -1) + (0 * 1) + (1 * 1)$$
$$= 0$$

2) Rectified Linear Unit (ReLU): ReLU is a non-linear activation function, which is used to apply elementwise non-linearity. ReLU layer applies an activation function to each element, such as the max (0, x) thresholding to zero. ReLU is by far the most successful activation function in deep neural network. Figure 3 shows the behavior of ReLU function. ReLU finds the negative values out of input and threshold it to zero.

3) Pooling Layer: CNN uses pooling layers for down-sampling. Pooling layers are interleaved in-between successive convolutional layers. It is used to reduce the size of the image so that the number of parameters get reduced and helps to control overfitting. The Pooling layer works on every activation feature maps independently and resizes it spatially, using MAX operation. For example, in Figure 4 a pooling layer with filter size 2*2 and stride 2 will reduce the image of size 4* 4 to 2*2 i.e. 50% less than the previous size. The max operation is used to find the largest number amongst the numbers that fall into a given filters window [14].
For example, in Figure 4 if the filter size is, 2*2 then it will cover the first rows and two columns and it will apply max operation as shown below. In Figure 4, the final reduced size of an output is shown, i.e. 2*2.

$$= max(2, 1, 0, 3)$$

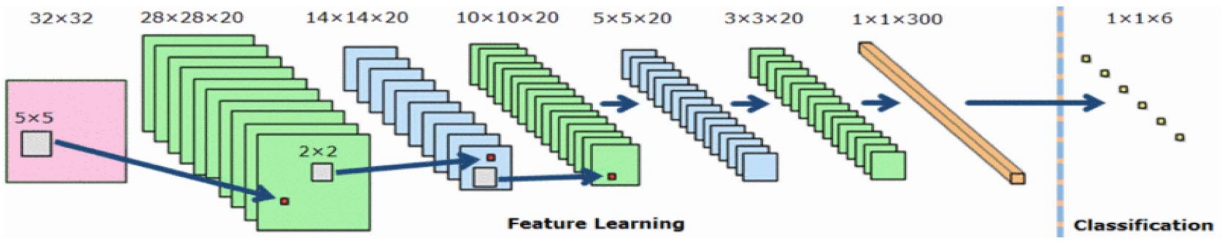

Fig. 1. Convolutional Neural Network[26].

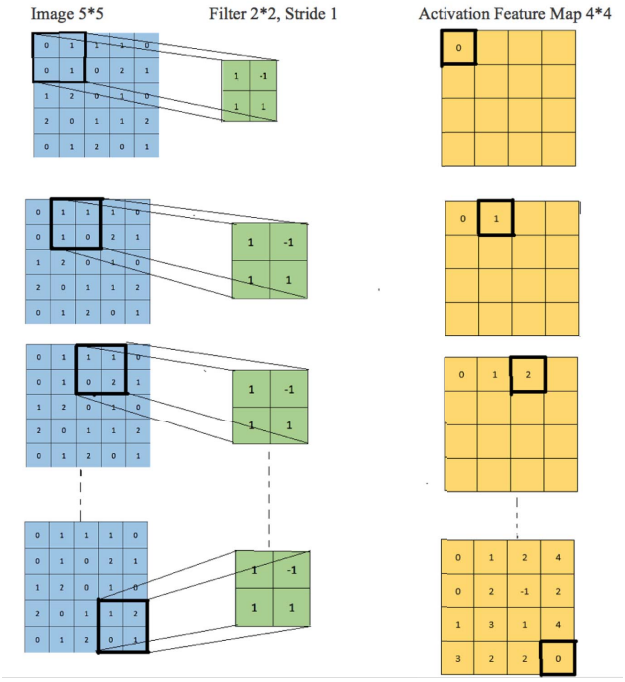

Fig. 2. The process of convolution on 5*5 image with 2*2 filter produce 4*4 Activation feature map[14].

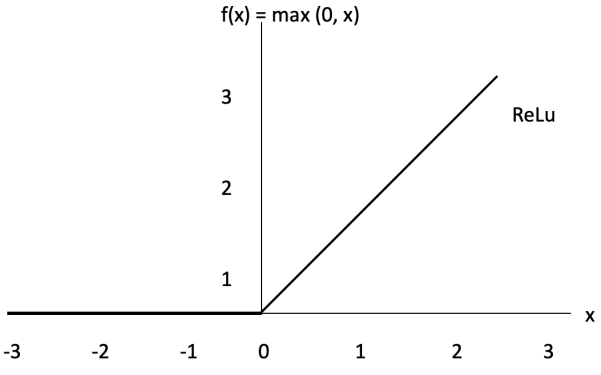

Fig. 3. Rectified Linear Unit.

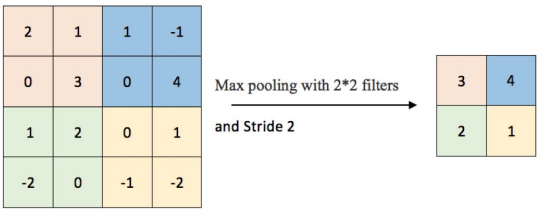

Fig. 4. Pooling Layer.

$$= 3$$

4) Fully Connected Layer: Fully connected layer is the last layer of the CNNs. This layer takes an input from preceding layer (i.e. convolutional layer or pooling layer or relu layer) and outputs an N dimensional vector, where N is the number of classes that algorithm must choose from. For example, in digit classification problem, N would be 10 because there are 10 digits (from 0-9) in our number system. Each number in this N dimensional vector specifies the probability of a certain class. For example, if the outcome of a digit classification problem is [0 .05 .05 .65 .1 .1 0 0 .05 0] vector, which means that there is probability of digit 1 is 0%, digit 2 is 5%, digit 3 is 5%, digit 4 is 65%, digit 5 is 10%, digit 6 is 10%, digit 7 is 0%, digit 8 is 0%, digit 9 is 10% and digit 10 is 0%. Therefore, this vector represents that the given image is 4 because of high probability of the corresponding number in the vector. FC layer performs a dot product with the output of the previous layer and the filters and produce the N-dimensional vector, which contains the probabilities for the different classes.

IV. Recurrent Neural Network

Figure 5 shows the RNN architecture, where each vertical rectangular box is a hidden layer and each layer contains several neurons. RNN comprises of the input layers (X_{t-1}, X_t, X_{t+1}), hidden layers (h_{t-1}, h_t, h_{t+1}), output layers (y_{t-1}, y_t, y_{t+1}) and weight matrices (W, U, V). RNN takes one input at each time step (for example, at time t, the first input X_t is given to the network) and then it is passed to the hidden layer to predict the output. The hidden layers are the important part of the RNN, because they keep the track

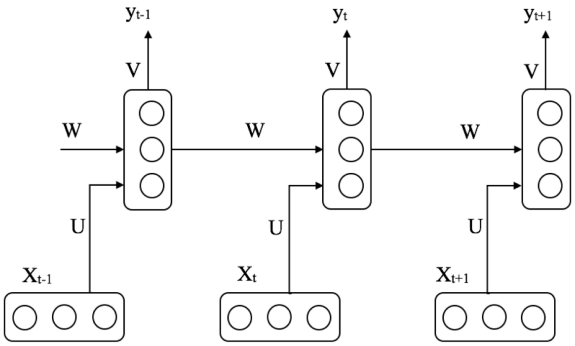

Fig. 5. RNN Architecture.

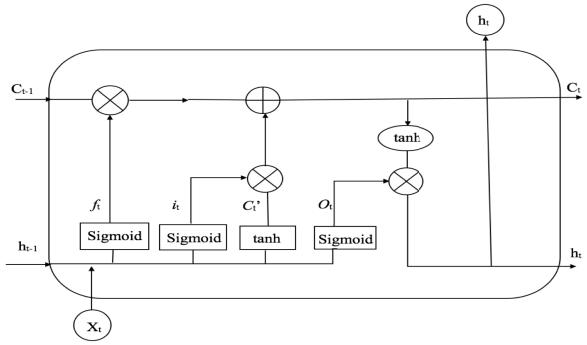

Fig. 6. Block diagram of LSTM.

of the previous work. The hidden layers take inputs from its input unit and from its previous hidden layer to predict its output. The weight matrix of the hidden layer (i.e., W as shown in Figure 5) has to be squared, because it helps to keep the same number of inputs, as there are outputs. The input layer matrix (i.e. U) and output layer matrix (i.e V) need not be squared, because it can connect to any random number of inputs to any random number of hidden units. In the beginning, all the weight matrices (i.e W, U, V) are randomly initialized.

The first hidden layer is initialized by the dot product of its current input X_{t-1} at time $t-1$ and weight matrix U. This dot product is passed through the activation function (sigmoid function) to generate the values for the first hidden layer. In general, to process the data, RNN takes an input X_t at time t, multiply it with weight matrix U and pass it to hidden layer h_t, an output of the previous hidden layer h_{t-1} parameterized with weight matrix W is given to the current hidden layer ht to predict the output y_t. The output y_t is obtained by taking a dot product of the present hidden layer h_t and weight matrix V. This process keeps on going until it covers all the layers and predict the final output.

A. Long Short-Term Memory (LSTM)

In this paper a special case of RNN called Long-Short-Term Memory (LSTM) with a peep hole connection is used.The main unit of an LSTM network [12] is the memory unit. This memory unit comprises of a cell state and a pair of gate layers as shown in Figure 6.An LSTM unit consists of three main states called cell state (C_{t-1}, C_t,), input state ($X_t and h_{t-1}$) and output state (h_{t-1}) and have four gates called forget gate (f_t), input gate (i_t), new memory gate (C_t), and output gate (O_t) to perform the internal operation [6]. The cell state (C_{t-1}, C_t,) is a crucial part of the LSTM (also called a memory unit) that runs through all the LSTM units in the network to transfer the information. This information is modified with the help of gate layers (a systematic work-flow of the four gates is shown in Figure 6). These gates are used to regulate the information and help LSTM to decide what information must be removed and what must be retained. Each LSTM unit has four gates that protect and control the flow of the information of cell state C_{t-1}. These gates are a way to allow correct information flow through one LSTM unit to another. Each LSTM unit takes three inputs X_t, h_{t-1}, and C_{t-1} and generates one output ht and a new cell state C_t. The input X_t given to LSTM can be a character, a word, or a speech, input h_{t-1} which is an input from the previous unit helps to control the flow of the information. If the current unit is the first unit of the LSTM then there is no previous input. In that case, a randomly generated value for h_{t-1} is given to the first unit to compute the functional blocks of sigmoid and tanh. Once these inputs are processed through the internal gates, then they are used to update the cell state C_{t-1} to C_t and help to predict the output h_t of the current LSTM unit.LSTM unit consists of four neural network layers and three of them are using sigmoid activation function and one is using tanh activation function as shown in Figure 6.

Figure 7 shows the peephole model of LSTM model. The proposed method uses a Convolutional Neural Network, a new variant of LSTM called LSTM with peephole connection .Peephole LSTM uses a weighted peephole connections from the cell state unit (C_{t-1}) to all the gates in the same memory unit as shown in Figure 7 [9]. The peephole connections allow every gate to assess the current cell state even though the output gate is closed and this peephole connections helped the proposed model to surpass the accuracy of the model called Update patterns in peephole LSTM along with the Stochastic "Hard Attention Model". Each cell state component must be updated based on the most current activation's of peep connection. The peephole connections need two-phase update scheme. In the first phase, when the recurrent connections are made with the gates, the following gates will be activated.

1) Input gate
2) Forget gate
3) Cell state

In Second phase, the output gate and the output of the LSTM unit will be activated [9].

V. Proposed Method

A. Convolutional Neural Network (CNN) Features extraction.

The proposed model takes a raw image and generates a Latex representation $y$encoded as a sequence of 1-of-K

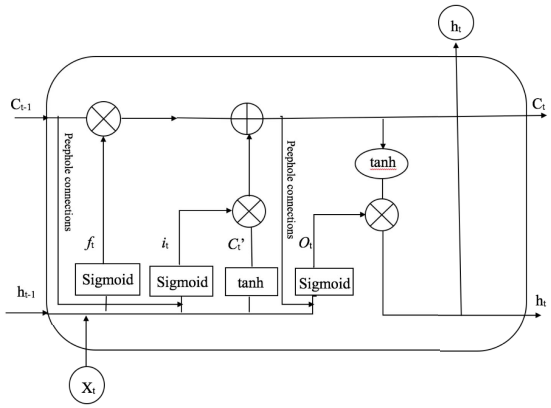

Fig. 7. LSTM with "peephole connection".

encoded latex word [25].

$$y = y_1, ..., y_c, y_i R^K$$

Where K is the size of the Latex vocabulary and C is the length of the Latex representation. In the proposed model, a convolutional neural network is used to extract a set of feature vectors, which is referred as an annotation vectors. The length of the generated annotation vector is L, each of which represents a D-dimensionality corresponding to a part of the image.

$$a = a_1, ..., a_L, a_i R^D$$

In this research, the features were obtained from the lower level of the CNN to get the close correspondence between the feature vectors and portions of the 2-D image, which allows the decoder (LSTM) to effectively emphasis on certain parts of an image by selecting a subset of all the feature vectors. In basic terms, at time t, the relevant part of an image is dynamically represented by context vector z_t'. The mechanism Φ that calculate z_t' from the annotation vectors $a_i, i = 1, ..., L$ represents the features that were extracts from different image locations. At each location i the attention mechanism calculates a positive weight α_i which can be recognized as the probability of that location i which then can be used by attention mechanism to focus for generating the next word in the sequence. This process is called hard but stochastic attention mechanism or a relative recognition can be given to i^{th} location in the a_is together. The positive weight α_i of every annotation vector a_i is generated by a hard attention model f_{att} and to compute that, a multilayer perceptron is conditioned on the hidden state h_{t-1}. The soft attention mechanism was introduced by Bahdanau et al. (2014).

In general, the hidden states of the RNN network changes as the output advances to the next level in the sequence however the next move of the network will rely on the previously generated words in the sequence.

$$e_{ti} = f_{(att)}(a_i, h_{(t-1)})$$

$$\alpha_{ti} = \frac{e^{((e_{ti}))}}{(\sum_{(k=1)}^{L} e^{((e_{ti}))})}$$

Once the weights (which sum to one) are computed, the context vector z_t' is computed by,

$$z_t' = \Phi(a_i, \alpha_i)$$

Once the annotation vector and positive weights are generated, the Φ function returns a single vector to calculate the output. We has used the hard attention mechanism to generate the output [25].

The entire proposed model is shown in Figure 8. It shows the six layer of CNN and LSTM part along with the attention model. A latex representation can be seen at the output of the model.

VI. Results

The experimental results are summarized in Table 1. The proposed method is compared with the previous two methods called INFTY and WYGIWYS on the bases of BLEU (Bilingual evaluation understudy) metric and Exact Match [10]. BLEU is a metric to evaluate the quality for the predicted Latex markup representation of the image. Exact Match is the metric which represents the percentage of the images classified correctly. The accuracy of the proposed model is 75.87% which is the highest in this research area. Previously, the highest result was around 73% achieved by WYGIWYS (What You Get Is What You See) model [21]. A Figure 9 shows an original image given to the model, Figure 10 shows the latex representation of the original image as an output, and Figure 11 shows the rendered image that is used to check how relevant is the latex representation to the original image.

TABLE I
Experiment Results on Image-to-Latex data-set.

Model	Preprocessing	BLEU	Exact Match
INFTY	–	51.20	15.60
WYGIWYS	Tokenize	73.71	74.46
PROPOSED MODEL	Tokenize	75.08	75.87

VII. CONCLUSIONS

In this research, a new variant of LSTM called LSTM with peephole connections and Stochastic hard Attention model is used to address the problem of OCR. In this experiment, the dataset called Image2latex-100K is used. However, I also have generated 200K Images of the mathematical equations to train and test my model. In this research, it is show that a new variant of the LSTM has outperformed the previous work based on traditional LSTM. This work will encourage other researchers to try the new variant of LSTM for ORC or sequence to sequence related work. For possible future work, this research can be scaled from printed mathematical formulas images to the hand written mathematical formulas

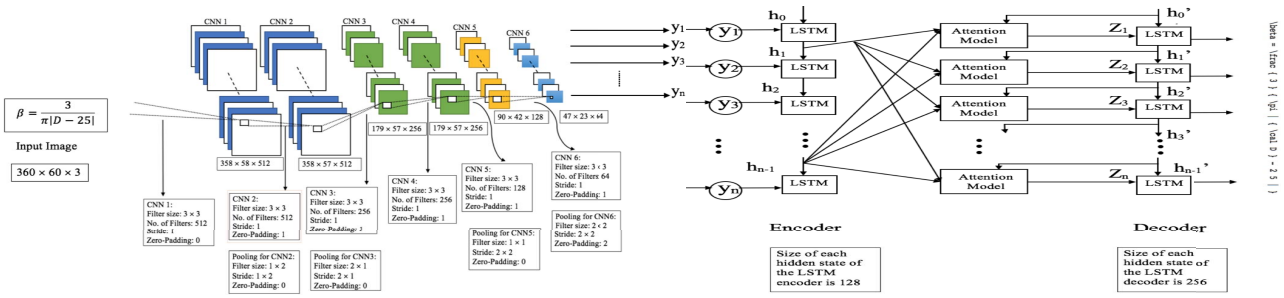

Fig. 8. The proposed model with CNN, LSTM, and Hard Attention.

$$\Theta = \log \frac{4N}{\mathcal{M}L}$$

Fig. 9. Original image given to CNN.

```
\Theta = \operatorname { l o g } \frac { 4 N } { { \cal M } L }
```

Fig. 10. Final Output of the model with latex representation .

$$\Theta = \mathrm{l\,o\,g}\, \frac{4N}{\mathcal{M}L}$$

Fig. 11. Rendered Image to check the relevance with the original image.

images. To recognize the hand written mathematical formulas, one can implement the bidirectional LSTM with CNN [18]. It can also be used to generate an API for latex code.

REFERENCES

[1] Anderson, R.H, Syntax-directed recognition of hand printed mathematics,Symposium,CA,1967.

[2] Bengio, Dzmitry Bahdanau, KyungHyun Cho Yoshua,Neural Machine Translation By Jointly Learning To Align And Translate, ICLR.

[3] Kyunghyun Cho Bart van Merrienboer Caglar Gulcehre Universite de Montreal, Fethi Bougares Holger Schwenk Universite du Maine, Dzmitry Bahdanau,Jacobs University, Yoshua Bengio,Learning Phrase Representations Using RNN Encoder-Decoder For Statistical Machine Translation, 2014.

[4] Bishop, Christopher M., Pattern Recognition and Machine Learning.

[5] Cho, Kyunghyun; Courville, Aaron; Bengio, Yoshua,Describing Multimedia Content Using Attention-Based Encoder-Decoder Networks, IEEE, 2015.

[6] Colah, Understanding LSTM Networks, http://colah.github.io/,http://colah.github.io/posts/2015-08-Understanding-LSTMs/ , 2015.

[7] Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio, Neural Machine Translation by Jointly Learning to Align and Translate,Accepted at ICLR 2015 as oral presentation.

[8] Ellis, Colin Raffel and Daniel P. W.FEED-FORWARD NETWORKS WITH ATTENTION CAN SOLVE SOME LONG-TERM MEMORY PROBLEMS,LABROSA, Columbia Univeristy, ICLR,2016.

[9] Gers, Felix A; Schraudolph, Nicol N; Schmidhuber, Jurgen, Learning Precise Timing with LSTM Recurrent Networks,Journal of Machine Learning Reserach,2002.

[10] H. Okamura, T. Kanahori, M. Suzuki, R. Rukuda W. Cong F. Tamari,Handwriting Interface for Computer Algebra System,Kyusha, China.

[11] heuritech, la chane,heuritech Le Blog,blog.heuritech.com,

[12] Hochreiter, S. and Schmidhuber, J.Long Short-Term Memory. Neural Computation,1997.

[13] Karpathy, and Fei-Fei, L,Image captioning.2015.

[14] Karpathy, Andrew,CS231n: Convolutional Neural Networks for Visual Recognition,Stanford.edu,http://cs231n.stanford.edu/.

[15] Le, Ilya Sutskever Oriol Vinyals Quoc V, Sequence to Sequence Learning with Neural Network.

[16] Learned-Miller, Andrew Kae and Erik, Learning on the Fly: Font-Free Approaches to Difficult OCR Problems, 2000.

[17] Lopresti, Daniel, Optical Character Recognition Errors and Their Effects on Natural Language Processing, International Journal on Document Analysis and Recognition,2008.

[18] M.Rush, Yuntian Deng Anssi Kanervisto Jeffrey Ling Alexander, Image-to-Markup Generation with Coarse-to-Fine Attention, 2017

[19] Max Jaderberg, Karen Simonyan Andrea Vedaldi, Andrew Zisserman, Reading Text in the wild with Convolutional Neural Networks, Int J Comput Vis (2016) 116:120 DOI 10.1007/s11263-015-0823-z Springer Science+Business Media New York 2015.

[20] Milad Mohammadi, Rohit Mundra, Richard Socher, Deep Learning for NLP1, Stanford, 2015.

[21] Yuntian Deng, Anssi Kanervisto, Alexander M. Rush, What You Get Is What You See: A visual Markup Decompiler.

[22] OPENAI, Requests for Research,

[23] Schantz, Herbert F.The history of OCR, optical character recognition, 1983.

[24] WILDML,http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/

[25] Xu, Kelvin; , Jimmy lei Ba; Cho, Kyunghyun ; Courville, Aaron; Salakhutdinov, Ruslan ; Zemel, Richard S; Bengio, Yoshua,Show, Attend and Tell: Neural Image Caption Generation with Visual Attention.

[26] Pavol Bezk, Yury Rafailovich Nikitin,and Pavol Boek, Robotic Grasping System Using Convolutional Neural Networks American Journal of Mechanical Engineering, 2014 2 (7), pp 216-218.DOI: 10.12691-ajme-2-7-9.