
CS224D Final Report: Deep Recurrent Attention Networks for \LaTeX to Source

Keegan Go

Department of Computer Science
Stanford University
Stanford, CA 94305
keegango@stanford.edu

Kenji Hata

Department of Computer Science
Stanford University
Stanford, CA 94305
khata@stanford.edu

Abstract

For our project, we wanted to explore the problem of recognizing \LaTeX expressions and translating them to source using a deep neural network. Previous work involving attention models have improved sequence to sequence mappings and greatly helped in digit recognition. Inspired by these previous work, we implemented an attention model to recognize simple \LaTeX expressions and also tested it on a small subset of CROHME, a dataset of handwritten mathematical expressions. We thoroughly explain our network, training procedure, hyperparameter tuning, and results. We achieve a classification accuracy of 63.4% on an automatically generated dataset and an accuracy of XXXXXX on CROHME.

1 Introduction

\LaTeX documents pervade the computer science community (and many others). In this project, we want to tackle the ability to transcribe \LaTeX expressions to source code. The applications of a highly robust transcription images of \LaTeX to source code are numerous: for example, taking a picture of any paper and having its source to compile and edit would be extremely useful for the research community.

The expressions found in \LaTeX equations naturally have a lot of structure compared to regular text. For instance, the expression

$$\frac{1}{1 + \frac{1}{1+\frac{1}{4}}} + \frac{1}{1 + \frac{1}{1+\frac{1}{4}}} + \frac{1}{1 + \frac{1}{1+\frac{1}{4}}}$$

has a recursive structure, which a deep convolutional recurrent network could learn and generalize it to other expressions.

Convolutional neural networks have recently seen great success and subsequently gained immense popularity in numerous classification and recognition tasks [10]. Additionally, recurrent networks have proven to learn sequence to sequence mappings. A combination of these two type of networks has become extremely effective recently, especially with the addition of attention mechanisms to further boost the efficacy of how deep networks "see" images [1, 4, 15].

In this project, we implement a deep recurrent, convolutional network with an attention model in order to identify \LaTeX tokens from simple expressions in images. We discuss our training process, hyperparameter tuning, and the results our network produces. Overall, we find that this network achieves an accuracy of 63.4% on 41 tokens.

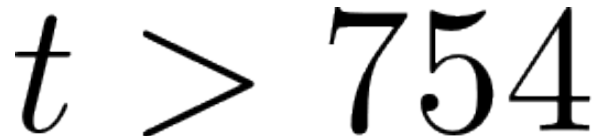
The image displays a mathematical expression in a serif font: the variable 't' followed by a greater-than sign '>' and the number '754'. The characters are black and set against a plain white background.

Figure 1: An example automatically generated \LaTeX image.

2 Related Work

2.1 Digit, Handwriting, and Mathematical Expression Recognition

Digit recognition is a simpler classification problem of mathematical expression recognition, but still proves to be useful. For example, the MNIST dataset [11] of handwritten digits has been used as a benchmark for a wide variety of machine learning algorithms. A more unstructured application of digit recognition is Google’s translation of streetview house numbers [4].

Two predominant forms of handwriting recognition: online, where the writing is recorded in real-time [6], and offline, where the data is stored statically, such as on paper [7]. The Competition on Recognition of Online Handwritten Mathematical Expressions (CROHME) dataset [13] provides a mapping from handwriting strokes to mathematical expressions. Work submitted to the CROHME competition often include online recognition using stroke to shape context features with AdaBoost [9] and a variety of foundational machine learning techniques (support vector machines, random forests, etc.) for offline recognition [2].

Previous approaches for \LaTeX expression recognition in apps like Photomath usually involve segmenting each individual token and then running each segmented token through a classifier. As far as we know, there has been no sufficient work on recognizing \LaTeX expressions using a deep neural network. Thus, we believe this project is a nice extension and application of previous deep learning techniques.

2.2 Attention Models

Many deep network models take inspiration from the way humans perform visual recognition tasks, specifically focusing on relevant areas as they progress through sequences [3]. Attention models for deep neural networks focus on different parts of images or sentences to help handle the recurrent or sequential nature inherent to many tasks [5, 8]. These attention models have proven to work on a wide variety of real-world problems, such as image captioning [15], multiple digit recognition [1], and image generation [8]

3 Data and Model

3.1 Data

We will initially simplify our problem as much as possible by generating \LaTeX documents and taking images of their outputted PDFs. By doing so, we can artificially generate as much data as we want to better train our neural networks. This idea can be further trained to produce results on images of long expressions from books or papers that one would generally not want to re-typeset.

For our dataset, we generated ten-thousand images of three to ten character expressions from a set of 41 tokens (a to z, 0 to 9, and a few operators). To generate each expression, first choose the length uniformly at random from the above range, and then choose characters uniformly at random from the character set. We did not enforce any semantic constraints at this point, since we wanted our model to learn how to map arbitrary sequences back into source. Figure 1 shows an example image in our automatically generated dataset.

While the compiled output generally varied in height and width, we standardized our data by first scaling all images to a height of 32 pixels and then padding the width to that of the widest image. While we initially hoped to avoid the final padding, we found it difficult to process the data in batches without matching the image sizes. Additionally, there exists the CROHME dataset [13],

$$y = r \sin \theta$$

Figure 2: An example CROHME image.

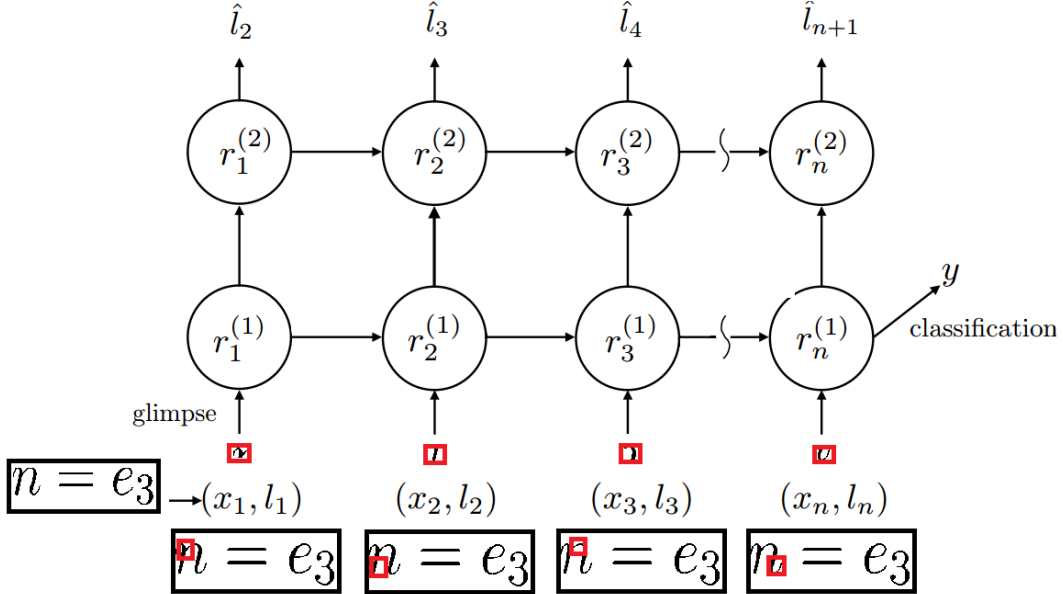


Figure 3: The RAM network we used to predict \LaTeX tokens. Multiple RAM networks can be stacked together to predict multiple tokens.

which allows us to extend to the handwritten domain as well. Figure 2 shows an example image in the CROHME dataset.

3.2 Model

Noticing that mathematical expression recognition, in a very simplified sense, can be represented as a sequence of digit or character recognition, we drew inspiration from Recurrent Attention Model (RAM) networks [12][1]. RAM networks identify salient parts of the image to look at and focus on these parts to improve their discrimination ability.

At a high level, our model does the following:

Glimpse Network: At time step t , given a location l_t , we generate a "glimpse" x_t of an image. This glimpse mirrors the foveal vision in humans, in that the area that the glimpse focuses on is at the highest resolution. As pixels become more distant from the glimpse, they progressively have less resolution. We send each glimpse x_t through three convolutional layers and a fully connected layer to generate $g_t^{(x)}$. Furthermore, we send each location l_t through a fully connected layer to generate $g_t^{(l)}$.

Recurrent Network: We have two recurrent networks interwoven:

$$r_t^{(1)} = \text{LSTM}(g_t^{(x)} \odot g_t^{(l)}, r_{t-1}^{(1)})$$

$$r_t^{(2)} = \text{LSTM}(r_t^{(1)}, r_{t-1}^{(2)})$$

Long-Short-Term-Memory is used for the non-linearity because of their ability to learn stable, long-range dependencies.

Classification Network: By sending the final $r_n^{(1)}$ through a fully connected layer and into a softmax, we compute the probabilities of the next LATEX token. This token will be our predicted value.

Emission Network: Using a fully connected layer with $r_t^{(2)}$ as input, we can compute the next location the network can focus its attention l_{t+1} .

To be explicit, we give the sizes and compositions of each component in the network.

- The Glimpse Network is composed of two sub-networks. It first maps the location (a 2-vector) and the associated glimpse (a 8×8 image) into two separate 128-vector hidden states using ReLUs. It then combines these two state vectors together into a 256-vector hidden state using ReLU again.
- We used LSTMs with hidden layers of size 256 for our Recurrent Networks.
- Our Classification Network is a simple fully connected layer fed into a softmax loss that takes the 256 size hidden state and maps it to predictions over the 41 character classes.
- The Emission Network consists of an affine layer fed into a tanh non-linearity to bound the locations to between the square given by ± 1 along the axes. We then scale these coordinates by the image height and width in order to allow the algorithm to gaze at any part of the image.

Our model was implemented in Torch (and partially in Tensorflow) on a NVIDIA GTX 970.

3.3 Training

While the model described above is powerful and has led to high performance results in a number of areas, it introduces additional complexities in training the model. In particular, the step of indexing the image to create the next glimpse is a non-differentiable function. This means that standard backpropagation techniques are not sufficient to train the model.

The solution is to use reinforcement learning to train the model to select glimpse locations that lead to good classification results. As the other attention models did, we used the REINFORCE algorithm [14]. At a high level, this technique works by sampling around the predicted location in the forward pass, and then treats the image as a fixed input in the backwards pass. To propagate error through the network, the Emission Network generates its own “gradient” by using a reward function based on the success of the classification.

4 Experiment and Results

4.1 Hyperparameter Tuning

After implementing the RAM network, we wanted to experiment on what parameters were the most influential on the network’s ability to learn. Upon a few quick tests, we found that learning rate, the reward scale for REINFORCE, and the standard deviation for the Monte Carlo sampling were among the most influential factors. Therefore, we did a standard grid search on these hyperparameters to find a combination of them that worked well. Figure 4 shows the results of our tuning. We found that a learning rate of 0.05, reward scale of 10, and standard deviation of 0.5 worked well.

However, upon completing this grid search, we wanted to further test the effect of the standard deviation of the Monte Carlo sampling, as we saw that it was somewhat sensitive. We believe this to be the case because, in order for the network to read each token at the correct position, the glimpse windows need to be able to quickly adjust towards the correct locations during training. Therefore, we completed additional tuning on the standard deviation, keeping the previously stated learning rate and reward scale. As seen in Figure 5, we found the standard deviation for Monte Carlo sampling did not play a major role. However, we noticed if the values were too low or too high, it would sometimes not learn at all. We believe this effect stems from the reinforcement learning ending up not improving the emission network.

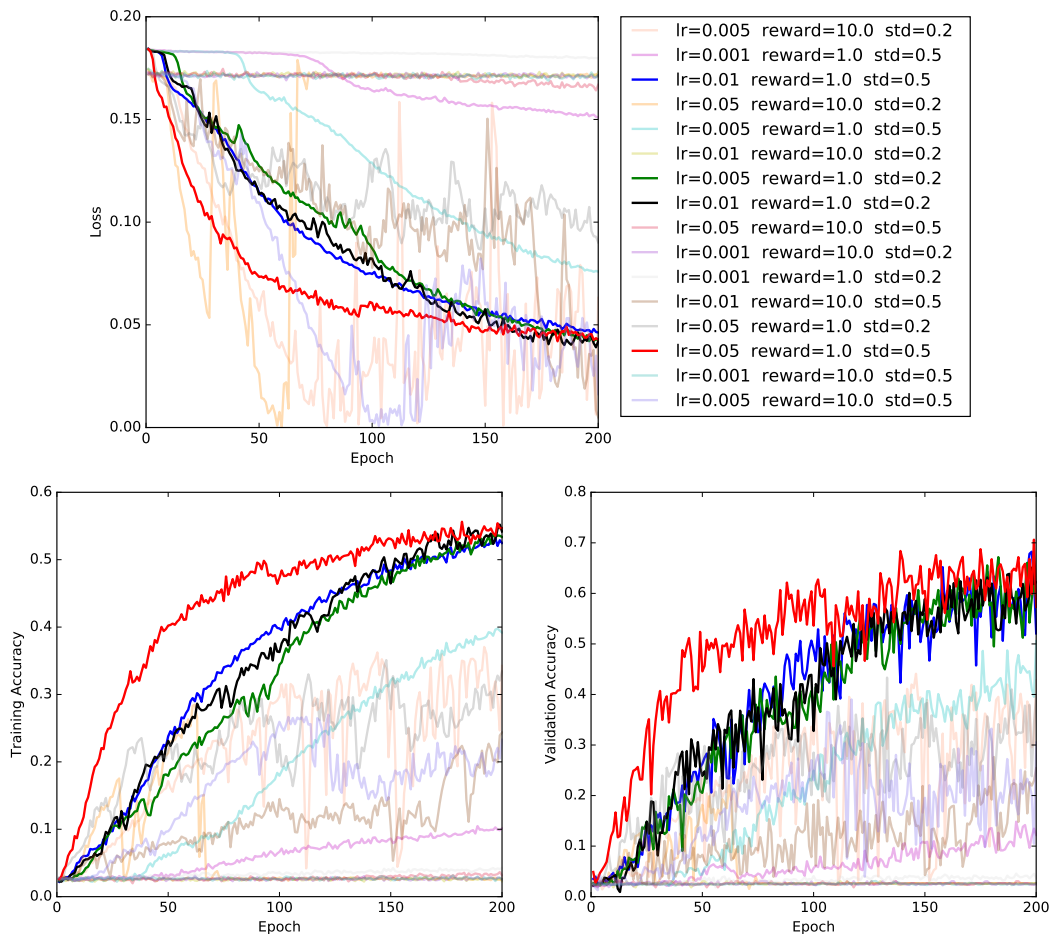


Figure 4: Hyperparameter tuning on learning rate, reward for REINFORCE, and standard deviation for the Monte Carlo sampling using a grid search. The best tuned network (red line) seems to follow a quick dropoff in loss and a corresponding sharp increase in accuracy.

4.2 Results

Our best model achieves a 58.6% accuracy on the training set, 66.5% accuracy on the validation set, and 63.4% accuracy on the test set. Although these numbers seem low compared to the 90% accuracy achieved by similar networks on the MNIST dataset, we believe that our data is more confusing for the network to learn because of the additional tokens within the image. These additional tokens make it confusing for the emission network to predict a new location.

4.3 Further Experiments on CROHME

As mentioned previously, we wanted to test our network on the CROHME dataset. Although the majority of the CROHME dataset uses very complex expressions, we chose a subset of around 500 images of easier expressions and tried to classify only 11 tokens (a few common characters and numbers). Ultimately, using our best model, we were able to achieve a training set accuracy of 45.1%, a validation set accuracy of 32.3%, and a test set accuracy of 28.7%. This is still much better than random, but there exists a lot of room for improvement.

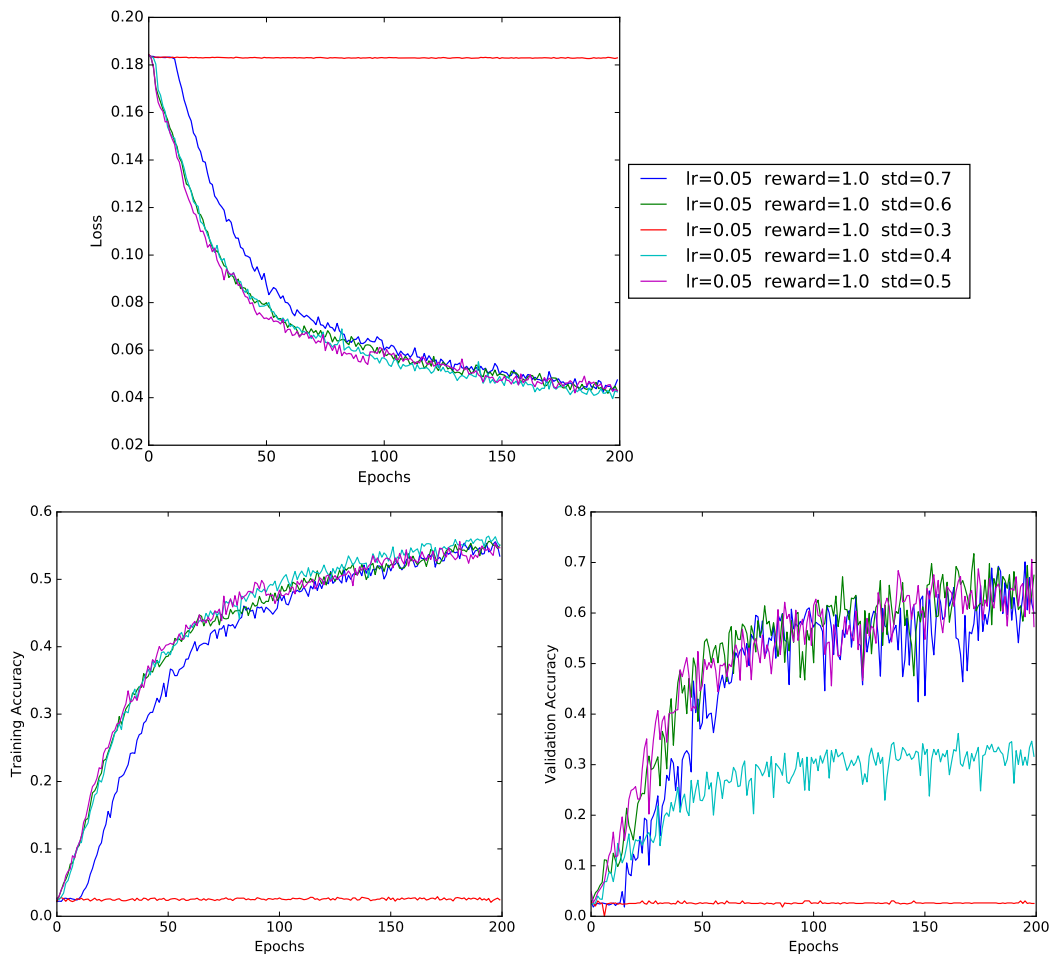


Figure 5: Hyperparameter tuning specifically on the standard deviation on the Monte Carlo sampling. We found that when the values were low or high, the network more frequently does not learn quickly.

5 Conclusion

We have implemented an attention based model for classifying \LaTeX expressions by character. We trained the network while varying hyperparameters and found a configuration that trains quickly and achieves significant results on our test set. However, as mentioned, training this type of model comes with the complication of introducing a number of new parameters associated with the reinforcement learning that need to be tuned in conjunction with the usual parameters. In addition to the larger parameter space, the parallel updates for the model from reward and error terms leads to more complicated behavior and so it is difficult to tell if the best results are being achieved.

For future work, we'd like to extend our model to making accurate predictions for arbitrary numbers of digits and for structured expressions. The second of these two is especially interesting, but will likely require a more recursive structure, perhaps outputting multiple subsequent glimpse locations and then tracing each of these separately.

References

- [1] J. Ba, V. Mnih, and K. Kavukcuoglu. Multiple object recognition with visual attention. *arXiv preprint arXiv:1412.7755*, 2014.

- [2] K. Davila, S. Ludi, and R. Zanibbi. Using off-line features and synthetic data for on-line handwritten math symbol recognition. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 323–328. IEEE, 2014.
- [3] R. Desimone and J. Duncan. Neural mechanisms of selective visual attention. *Annual review of neuroscience*, 18(1):193–222, 1995.
- [4] I. J. Goodfellow, Y. Bulatov, J. Ibarz, S. Arnoud, and V. Shet. Multi-digit number recognition from street view imagery using deep convolutional neural networks. *arXiv preprint arXiv:1312.6082*, 2013.
- [5] A. Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.
- [6] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(5):855–868, 2009.
- [7] A. Graves and J. Schmidhuber. Offline handwriting recognition with multidimensional recurrent neural networks. In *Advances in neural information processing systems*, pages 545–552, 2009.
- [8] K. Gregor, I. Danihelka, A. Graves, and D. Wierstra. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.
- [9] L. Hu and R. Zanibbi. Segmenting handwritten math symbols using adaboost and multi-scale shape context features. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 1180–1184. IEEE, 2013.
- [10] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [11] Y. LeCun, C. Cortes, and C. J. Burges. The mnist database of handwritten digits, 1998.
- [12] V. Mnih, N. Heess, A. Graves, et al. Recurrent models of visual attention. In *Advances in Neural Information Processing Systems*, pages 2204–2212, 2014.
- [13] H. Mouchere, C. Viard-Gaudin, R. Zanibbi, and U. Garain. Icfhr 2014 competition on recognition of on-line handwritten mathematical expressions (crohme 2014). In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 791–796. IEEE, 2014.
- [14] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.
- [15] K. Xu, J. Ba, R. Kiros, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*, 2015.