ORFE 525: Statistical Learning and Nonparametric Estimation
Project Literature Review

Zachary Hervieux-Moore

Tuesday 18th April, 2017

1 Introduction

The problem that this project aims to address is that of classifying handwritten mathematical equations to its correct *LaTeX* formulation. In fact, this is very similar to a request for research put out by Open AI [1]. A team from Harvard has submitted their solution on Open AI and has graciously provided their 100k image dataset [2]. The techniques we wish to use to tackle this problem are modern neural network architectures such as the sequence-to-sequence model, a type of recurrent neural network (RNN), and generative adversarial networks (GAN). The proceeding sections aim to introduce these architectures and how they may be applied to this problem.

2 Similar Research

Alluded to above, one team has tackled a similar problem. The team had two goals, the first was to take images of compiled *LaTeX* formulas and generate the original *LaTeX*. The other was to take web pages and generate the HTML that rendered it. Both problems are similar as they involved decompiling an end result into an original markup language. The architecture they used was a combination of a convolutional neural network (CNN) to extract features from an image and then to use an sequence-to-sequence model architecture to allow for varying input and output sizes. Both these architectures are explored further in section 3. With this structure, they are able to achieve an exact match (correctly outputting *LaTeX* that generates the image) rate of 75% [2]. A result which is extremely relevant to the problem at hand as the only change to the problem is the input from compiled *LaTeX* images to handwritten equations. While a similar architecture as above may be used, differences in the convolutional layer may be beneficial to orient the network to extract features from handwriting as opposed to typeset formulas.

3 Architectures

Convolutional Neural Networks

CNN's were covered in the class by Junwei and so not too much time will be spent here. What will be said is that CNN's have been used for a very long time for optical character recognition as exemplified by LeCun *et al.* in [3]. Since then, there has been the introduction of max pooling, rectified linear units (ReLU), fully connected layers, among other variants. One merely has to look at any of the most successful neural networks in image recognition [4], [5], text comprehension [6], or video classification [7] to see an example of a modern day convolution neural network.

Recurrent Neural Networks

Again, Junwei went over the basics of RNN's in class and so we skip the conversation to the state of the art architectures relevant to the application in mind. One clear problem that our neural network will face is turning images into arbitrary length pieces of *LaTeX* code. Originally, RNN's only allowed for fixed length outputs. However, Cho *et al.* introduced the sequence-to-sequence model in [8] which allows for arbitrary input and output lengths. To understand how it works, Figure 1 displays the general architecture.

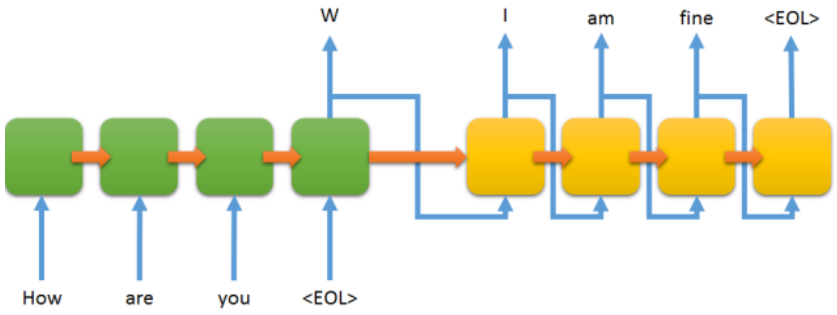

Figure 1: General Layout of a Sequence-to-Sequence Model [9]

First, there are two components of the network, the encoder (green) and decoder (yellow). How the network works is that the encoder is fed an arbitrary long input sequence until the sequence ends (denoted by `<EOL>`) above. Once the `<EOL>` is fed into the encoder, it produces an output. Now, the output is

fed into the decoder and this is used to generate the output sequence ("I" being the first output). This output then becomes the input and is fed until the decoder produces an `<EOL>` of its own. A prime example of this architecture is used in translation [10].

One problem that arises with this architecture is that the encoder tends to not be able to learn long term behaviour. This is rectified by using Long Short Term Memory Cells (LSTM) which are the blocks in Figure 1. The details will be spared here, but the main idea is that you keep a state of the network along with the outputs and force the state to lose information ("forget") about the previous state if it is not heavily weighted and replace this with information from the current input. For an in depth review of LSTM's and a comparison of its variants see [11].

Although the above sequence-to-sequence model works quite well, there is one potential deficiency. That is, in the step from the encoder to the decoder, the output of the encoder must be of a fixed length. Considering that the input is of variable length, this begs the question of whether or not the fixed length at the end diminishes the effect of learning from an arbitrary sequence. This is the exact problem that Bahdanau *et al.* attempt to solve in [12]. Their proposed solution is a so-called attention mechanism. This essentially makes the network much more complicated by feeding all of the outputs from the encoder to be available at each input for the decoder. However, for the added complexity, one generally achieves better results and using the attention mechanism allows one to analyze how the network is weighting different inputs to get a better sense of how the network is working as described in the results section of [12].

Generative Adversarial Networks

The last architecture we will mention is the generative adversarial network (GAN). The GAN is a very recent result from Goodfellow *et al.* [13]. In this paper, they create a model that allows for a trained network to generate "fake" data that resembles the training set given to the network. This is achieved by creating two agents, a discriminator, and a generator whose job it is to fool the other. That is, the discriminator is trained to detect fake input given by generator and the generator is trained to try and fool the discriminator. Mathematically, the objective is given in [13] as

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{x\sim p_z(z)}[\log(1 - D(G(z)))]$$

To see how GAN's might be useful for my application, consider the work in [14]. In this paper, the researchers create a GAN that is able to generate images of birds and flowers based on a descriptive text. While this is not particularly relevant, the part that is promising is that they are able to generate images from text descriptions that the network has never seen before. Consider flipping their network, that is, images to text descriptions. In my case, it would be *LaTeX* equations, and training a GAN that outputs *LaTeX* this way. The problem with this architecture is that it requires many examples of a specific class. Whereas, in my problem, we are given one equation and one label. Thus, we turn our attention to applications of GAN's for classification purposes.

One example of work done in the area of classification using GAN's is [15]. In this paper, they use a GAN as a feature extractor and then use these features in a linear model (SVM in this case). With this, they are able to achieve results better than other modern unsupervised learning techniques but worse than some unsupervised CNN architectures. However, this is accomplished using half as many feature units which suggests that there is room for improvement with more computational power. In the same paper, the team accomplishes state of the art performance in another task. This demonstrates the promise of GAN's in classification. Again, the issue with these results with respect to handwriting to *LaTeX* is that they depend on a small number of classes. By design, this problem has many different *LaTeX* outputs and so these techniques cannot be applied directly.

4 Conclusion

To achieve the desired goal, the use of CNN's and RNN's seem necessary after reviewing the literature. However, there is potential for GAN's to be used in a novel way for classification. As this was a literature review, discussion of the implementation specifics was omitted but still important for the overall success of the project. These include data acquisition, data processing, training scheme, etc. All things considered, this is a very approachable problem and state of the art performance could be possible.

References

[1] "Requests for research." https://openai.com/requests-for-research/#im2latex. Accessed: 2017-04-07.

[2] Y. Deng, A. Kanervisto, and A. M. Rush, "What you get is what you see: A visual markup decompiler," 2016.

[3] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, p. 2278–2324, 1998.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25* (F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds.), pp. 1097–1105, Curran Associates, Inc., 2012.

[5] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection," in *Advances in Neural Information Processing Systems*, pp. 2553–2561, 2013.

[6] T. Wang, D. J. Wu, A. Coates, and A. Y. Ng, "End-to-end text recognition with convolutional neural networks," in *Pattern Recognition (ICPR), 2012 21st International Conference on*, pp. 3304–3308, IEEE, 2012.

[7] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1725–1732, 2014.

[8] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," 2014.

[9] F. Rahman, "Seq2seq." https://github.com/farizrahman4u/seq2seq. Accessed: 2017-04-08.

[10] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," 2014.

[11] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," 2015.

[12] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014.

[13] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014.

[14] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," in *Proceedings of The 33rd International Conference on Machine Learning*, vol. 3, 2016.

[15] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2015.