Generative Mathematical Problems - Computational Creativity Project

Dimitrios Mylonas-180326363

School of Electronic Engineering and Computer Science
Queen Mary University of London, UK
ec21882@qmul.ac.uk

Abstract. In this paper, a system for Mathematical Word Problem generation, for several different mathematical topics at around school-level, is proposed. This is achieved through a LSTM Recurrent Neural Network-based model for text generation, utilising DeepMind's mathematics and Microsoft's Dolphin18K datasets. The system produces encouraging results, at times generating solvable and original (which do not appear in the original datasets) problems. Finally, possible applications of such a tool are suggested, which allow for the exploration of higher level computational creativity issues regarding collaboration of humans and AI in education and research.

1 Introduction

Despite the indisputable creative aspect of mathematical and scientific discovery, research in these domains is significantly less represented in the Computational Creativity (CC) community [12]. It could be argued that a greater focus on these could "bring the reputation of CC away from a purely aesthetic domain" [10] and into real-world problem solving.

A fundamental area of this type of mathematical creativity is that of Mathematical Word Problems (MWP). Generating and solving MWP has been an active research area for many years, primarily through the use of hand-crafted rules and semantic parsing techniques. However, MWP and in general automated reasoning has regained attention in the past few years with the advancement of powerful AI techniques for natural language understanding.

Advancement in solving and generating MWP can have major pedagogical benefits [11] [15], such as tools for curriculum design and analysis, and automatic content generation. In this paper, a system for generating MWP through Recurrent Neural Network text generating techniques is proposed. With the use of two different MWP datasets, encouraging results are achieved for a wide variety of different school level topics. This setting also allows exploration of the higher level issue of the collaboration between AI and human educators.

2 Background

MWP generation and systems that can also solve them are of great interest to researchers. Most resources are being focused on the solving aspect of the problem, but

the two are intrinsically related, as a human that understands linear algebra would be able to formulate a question on it. Both therefore face the same challenge of the "semantic gap" between human-understandable words and machine-understandable logic [16].

Classically, attempts in MWP generation were mainly based on templates. In [13] the authors consider a number of different techniques such as graph construction and [9] attempt to produce MWP by revising existing questions into a new topic. With the rise of deep learning, a lot of papers are attempting to narrow the semantic gap. Neural Network methods for question and story generation such as [6], [7], and [18] produce great results, but cannot be used for MWP as they lack the intrinsic logic to produce solvable questions.

Some papers attempting to resolve this issue include [17] where the authors propose a RNN based encoder-decoder architecture that takes into account both the topic and underlying equation. Another more recent paper uses the GTP-3 [1] transformer language model combined with program synthesis to produce state of the art results in both generating and solving university math problems [5].

3 Datasets

The first dataset used in this paper is DeepMind's mathematics dataset [14]. It consists of .txt files on many different mathematical topics such as linear algebra, probability, calculus etc. and it's laid out in question-answer pairs through the file. To make files usable for generation purposes, the lines that contain the solution to the problems were omitted, retaining only the questions.

The second dataset used is a subset of Microsoft's Dolphin18K dataset [8] containing 1,878 MWP from Yahoo! Answers in .json format. The files were parsed using a python script to convert them into a .txt file containing only the questions.

As the text-generation system is character-based, the questions are pre-processed so that they are represented with unique ID's for each letter by use of a dictionary. As this is essentially a next-character prediction task, the dataset is split into chunks of input data of a given number of characters and target data, which is a chunk starting from the second character of the input data and ending one character further. These are then batched in numbers of our choice and shuffled.

4 System Description

The system is based on a Recurrent Neural Network (RNN) structure, more specifically utilising Long Short Term Memory (LSTM) units.

RNNs are a special kind of neural network (NN) architecture that can, unlike feed-forward NNs, retain information with the use of an additional hidden state, which gives it a type of 'memory'. This is particularly useful when, like in this case, sequences of data are fed into the RNN, as it helps retain information about past. In the problem of text generation, keeping track of what has been said and the ability to work with any length of sequence data are convenient features.

Simple RNNs however have their downfalls. As RNN weights are updated through backpropagation so as to improve the model's performance, this gives rise to the vanishing gradient problem. This occurs because to carry out backpropagation, the error gradients need to be calculated for each weight. This is an issue for the hidden state gradient, because for each time step, this gradient is dependent on the whole history of the network. This leads to information that is too 'far back' from the current time step being lost as gradients get infinitesimally small.

In this paper, a model using LSTMs is created instead (although vanilla RNNs are also explored). LSTM units allow the network to have greater control of the information retention by utilising a cell state and 3 gates. The first, forget gate, allows the LSTM to discard information directly. The second, update gate, allows the addition of new information. Finally, the output gate, produces the final output which is used for the next timestep as a hidden state.

The model used for the system is composed of an embedding layer, which creates embeddings for each character through training, LSTM layers and dropout layers and a final dense layer. The exact architecture can be seen in Fig.1. The final system is a tool that gives the user an option to choose which topic they would like to practise and how many questions they would want generated. This tool loads the pre-trained weights of models, with the difference that the networks now have a final dense layer of only 1 character to produce text.

5 Experiments and Results

The first investigation is that of a simple RNN model based on a a 256 unit embedding layer, a RNN layer with 1024 units and Dense layer mapping to each of the unique characters in the DeepMind dataset. It was found that due to the regularity of the data (questions being similar for each topic) and the large amount of data for each topic, a small amount of epochs is required for the training to reach a minimum loss, beyond where the model does not improve further.

This structure was used to train multiple systems of MWP generators, each for a different topic. Some examples of questions produced for different topics are seen in Table 1.

Some of the systems, especially for calculus and unit conversion produce good results with questions that do not appear in the original dataset. However, a clear pattern emerges: questions that consist of more than one sentence become unsolvable. This is because the system fails to 'remember' what variables it has stated in the previous part of the question. Changing hyper-parameters such as learning rate, batch number, and the amount of words in each training sequence window were experimented with but had no noticeable effect on producing better text, only affecting the time taken for the algorithm to converge.

To resolve this issue, the RNN based structure of the system was converted into a LSTM based one as described in the system description. This was with the hope that the 'memory' functionality of LSTM would be useful in retaining information during a long question. Although this change had a positive effect on the minimum loss achieved, it did not fix the issue of disjointed math problems. Nonetheless, the LSTM structure was

Topic	Examples of Generated Questions
Calculus	Find the second derivative of 633*i**3 + 117947*i + 6152*i**6 wrt i.
	Differentiate -1827 + 1388*r**2 - 548 - 186369 - 31216*r - 1524*r + 41801 wrt r.
	Let a(l) be the first derivative of -969*l**4/4 + 2455*l + 1470. Differentiate x(b) with respect to b.
Linear Algebra	Let j be (12 - 253) + 2 + -3. Solve c*d + 6 = 29*d for d.
	Let x be ((-8)/5)/(-1*(-48)/(-44). Suppose 0 = c*v + 45 - 48. Solve -v = -l - 5*c for c.
	Solve 17191 = 1842*v - 424*v + 383*v - 1505 for v.
Conversion of units	How many months are there in 0.5404472 millennia?
	How many millilitres are there in seventy-nine fifths of a litre?
	How many nanoseconds are there in 8250.019ns?
Greatest Common Divisor	Suppose 2*t - 24 = -p, 5*l + 2*m + 67 = 3*f. What is the greatest common factor of m and q?
	Suppose 5*u - 2021 = 2*m, 2*u - 2*f = -2*u = -175 + 226. What is the greatest common divisor of m and 49?
	Let m be 1 - 0 - (142/(-54)*-90)/((-2)/(-165))). Calculate the highest common factor of h and 20.
Probability	What is prob of picking 2 x when two letters picked without replacement from {z: 1, c: 1, x: 2}.
	Calculate prob of picking 3 d and 1 x when four letters picked without replacement from egcgyf
	Three letters picked without replacement from {i: 1, q: 1, z: 2, v: 1, z: 1, g: 1}.

Table 1: Examples of generated questions for different math topics using a RNN system.

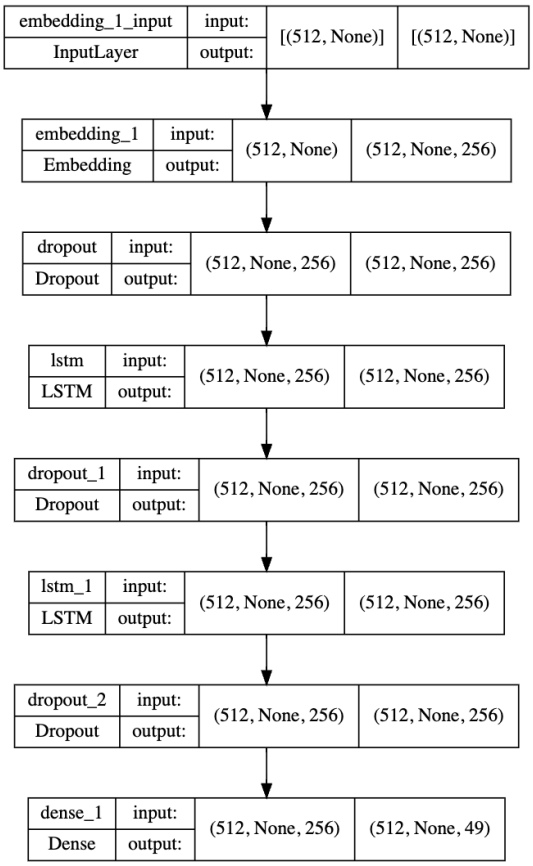

Fig. 1: Architecture of the generative system.

kept for the following part, which was to train a system to generate a mix of 2 different topics. The system was successful in generating a mix of unit conversion and linear algebra questions, however occasionally fused the topics together in a single questions. Example of a sequence of 5 questions produced is seen in Table 2.

It can be assumed that this functionality can extend to include more topics but can expect a greater unwanted fusion of said topics.

Finally, the LSTM model was trained with a subset of the Dolphin18K dataset. This dataset contains MWP from various different topics, with a lot less ”standardised” structure in the questions, as they are obtained from Yahoo! answers. A learning rate of 0.005, with batch size 32 and 50 epochs were used to arrive to the final generative system. Running the system for more than 50 epochs resulted in the loss increasing, and so early stopping was applied. Results produced from this system are shown in Table 3.

This system has produced a surprisingly valid Question 1. Not only does the question not appear in the original and makes sense grammatically, but it is also solvable, which is not always the case with questions of this type when plugging in random num-

	Questions produced
1	Solve 36987 = -4161*x - 1694 + 3462 - 359 for x.
2	How many months are there in 48/5 of a millennium?
3	What is 7/6 of a kilometer in + 3677 = 6885 for x.
4	Convert 431175.9ns to minutes.
5	Solve 3046 = 250*m + 27 - 657 for m.
6	Solve 0 = 131*d + 126*d - 132*d - 6060 for d.

Table 2: Examples of generated questions for 2 different math topics combined using a LSTM system.

	Questions produced
1	The sum of two numbers is 32. The difference of the two numbers is 12. find the numbers
2	find two consecutive even integers such that four times the smaller is 62
3	The product of two consecutive odd integers is -286. Find the two integers
4	Find three consecutive odd integers such that the sum of the largest and second is five less than twice the largest. Find the integers
5	What number is 6. Find the number

Table 3: Examples of generated questions using the Dolphin18K dataset and a LSTM model.

bers. Questions 2 and 3 are examples of grammatically correct questions but which do not make sense mathematically. This dataset also runs into the same issues as previously, that of long questions that 'forget', as can be seen in Question 4. Finally, occasionally it produces some nonsensical, and slightly entertaining questions, such as Question 5.

6 Discussion and Higher Level Issue

Overall, the results are encouraging. Given that the system only uses a character based method, completely uninformed to mathematical logic, is impressive. Some of the questions produced are surprisingly comprehensible given no pre-trained embeddings from a large corpus were used. The curation coefficient [3] of the system can reach a number of 1 when it comes to shorter questions, but depending on the topic, long questions composed of multiple sentences are generally less presentable.

The system also encounters a number of issues. Mainly, the LSTM was unsuccessful in retaining this very much needed information from previous parts of the question. Perhaps such a simple model cannot be improved further without introducing more sophisticated methods, like transformers or guidance from a logic system.

Nonetheless, the results produced show that such a creative system could potentially find its place in some segment of education, especially in some kind of partnership between human educators and AI. This could prove beneficial for students, as MWP systems can produce tailored and unique questions for them, perhaps even testing their

knowledge on a subject in creative and novel ways, which humans have missed. In the future, AI could hope to write its own examination papers, or even its own textbooks, with fresh examples and explanations, and self-made exercises in the back of the book.

This potentiality however can be controversial. For some, AI may seem amusing and harmless when it comes to producing beautiful images and funny texts, but a line is drawn when that creativeness is used in a way that could decide the trajectory of a young generation, especially when it comes to such a serious matter as education. Automatically generated content can seem untrustworthy in this case and it might take a long time before the public can entrust AI with anything more than menial tasks.

A possible solution for this is some version of framing [4]by the generative system. A clarification on the reasoning behind why a certain question was generated, and what it aims to teach that specific student can relieve some worries and provide that much needed reassurance to the audience.

According to Colton's tripod view of creative ability [2], skill, appreciation, and imagination are needed by an AI to be creative. Although the system made here possessed skill to be able to produce sensible MWP, it lacked the other two prongs. Appreciation was absent as no technique like framing was implemented, and the system was prone to pastiche due to the similar looking questions in the dataset, arguably making the system not very imaginative.

7 Conclusions and Future Work

This paper was successful in demonstrating a text-based technique for MWP generation, producing satisfactory results for a number of different topics. It failed however, at producing sensible outputs for more complex, longer questions.

This problem could be addressed in further work in a number of ways. A reasonable approach would be to introduce attention mechanisms with transformers to the text generation task. This could result in increasing performance when it comes to longer MWP, as the system can now potentially "pay" attention to specific variable names further back in the text.

Another approach is to introduce some mathematical knowledge to the system. This could guide it away from generating non-solvable problems, as it would be able to "check" its own work before displaying the resulting MWP, helping the system attain the appreciation prong of the Colton's tripod. The remaining prong of imagination can be addressed by training the system with a much more varied dataset, to avoid pastiche.

This paper also explored the high level issue of human-AI collaboration in education and established that it will be challenging to convince the public that use of AI is appropriate. Further work can approach this problem by introducing framing, where the system will be able to comment on its decisions and reassure the audience, in hope of attaining a favourable attitude towards it.

References

1. T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan,

R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," 2020. [Online]. Available: https://arxiv.org/abs/2005.14165

2. S. Colton, "Creativity versus the perception of creativity in computational systems," in *AAAI Spring Symposium: Creative Intelligent Systems*, 2008.
3. S. Colton and G. Wiggins, "Computational creativity: The final frontier?" in *Proceedings of the 20th European Conference on Artificial Intelligence*, ser. Frontiers in Artificial Intelligence and Applications, 2012.
4. M. Cook, S. Colton, A. Pease, and M. T. Llano, "Framing in computational creativity - a survey and taxonomy," in *ICCC*, 2019.
5. I. Drori, S. Zhang, R. Shuttleworth, L. Tang, A. Lu, E. Ke, K. Liu, L. Chen, S. Tran, N. Cheng, R. Wang, N. Singh, T. L. Patti, J. Lynch, A. Shporer, N. Verma, E. Wu, and G. Strang, "A neural network solves, explains, and generates university math problems by program synthesis and few-shot learning at human level," 2021. [Online]. Available: https://arxiv.org/abs/2112.15594
6. X. Du, J. Shao, and C. Cardie, "Learning to ask: Neural question generation for reading comprehension," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 1342–1352. [Online]. Available: https://aclanthology.org/P17-1123
7. A. Fan, M. Lewis, and Y. Dauphin, "Hierarchical neural story generation," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 889–898. [Online]. Available: https://aclanthology.org/P18-1082
8. D. Huang, S. Shi, C.-Y. Lin, J. Yin, and W.-Y. Ma, "How well do computers solve math word problems? large-scale dataset construction and evaluation," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 887–896. [Online]. Available: https://aclanthology.org/P16-1084
9. R. Koncel-Kedziorski, I. Konstas, L. Zettlemoyer, and H. Hajishirzi, "A theme-rewriting approach for generating algebra word problems," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, Nov. 2016, pp. 1617–1628. [Online]. Available: https://aclanthology.org/D16-1168
10. R. Loughran and M. neill, "Application domains considered in computational creativity," 06 2017.
11. K. Nandhini and S. R. Balasundaram, "Math word question generation for training the students with learning difficulties," in *Proceedings of the International Conference amp; Workshop on Emerging Trends in Technology*, ser. ICWET '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 206–211. [Online]. Available: https://doi.org/10.1145/1980022.1980069
12. A. Pease, S. Colton, C. Warburton, A. Nathanail, I. Preda, D. Arnold, D. Winterstein, and M. Cook, "The importance of applying computational creativity to scientific and mathematical domains," in *Proceedings of the 10th International Conference on Computational Creativity, ICCC 2019*, ser. Proceedings of the 10th International Conference on Computational Creativity, ICCC 2019, K. Grace, M. Cook, D. Ventura, and M. Maher, Eds. Association for Computational Creativity, Jun. 2019, pp. 250–257.
13. O. Polozov, E. O'Rourke, A. M. Smith, L. Zettlemoyer, S. Gulwani, and Z. Popović, "Personalized mathematical word problem generation," in *IJCAI 2015*, May 2015.

14. D. Saxton, E. Grefenstette, F. Hill, and P. Kohli, “Analysing mathematical reasoning abilities of neural models,” 2019. [Online]. Available: https://arxiv.org/abs/1904.01557
15. N. A. Smith and M. Heilman, “Automatic factual question generation from text,” 2011.
16. D. Zhang, L. Wang, L. Zhang, B. T. Dai, and H. T. Shen, “The gap of semantic parsing: A survey on automatic math word problem solvers,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 9, pp. 2287–2305, 2020.
17. Q. Zhou and D. Huang, “Towards generating math word problems from equations and topics,” in *Proceedings of the 12th International Conference on Natural Language Generation*. Tokyo, Japan: Association for Computational Linguistics, Oct.–Nov. 2019, pp. 494–503. [Online]. Available: https://aclanthology.org/W19-8661
18. Q. Zhou, N. Yang, F. Wei, C. Tan, H. Bao, and M. Zhou, “Neural question generation from text: A preliminary study,” 2017. [Online]. Available: https://arxiv.org/abs/1704.01792