Advanced language models for scientific text generation

Zineb Et-tarraf [2]
Supervised by Cyril Labbé [1]

[1] *Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG, 38000 Grenoble, France ,ORCID: 0000-0003-4855-7038*
E-mail: cyril.labbe@univ-grenoble-alpes.fr

[2] *Student at Ensimag, a school of computer science and applied mathematics , 38000 Grenoble*
E-mail: zineb.et-tarraf@grenoble-inp.org

13 May 2022

Meaningless scientific papers generated by a computer program still appear in the scientific literature several years after the problem was first seen in 2005, when a few students at MIT's Computer Science and Artificial Intelligence Lab created paper-generating software called SCIgen. Such nonsensical papers have been detected thanks to the researcher Cyril Labbé who went on to find more than 120 fake SCIgen papers published by the IEEE and by Springer . But now based on neural language models automatic text generation has achieved performance levels that make the generated text almost indistinguishable from those written by humans which make the detection task more complicated . In this paper we propose to combine a model of generating text based on a neural network and grammar-based text generator to generate text more complicated to be detected by the former detectors with the aim of finding a more robust detector.

Introduction

Using SCIgen, Labbé, who works at the University of Grenoble Alpes in France, generates 102 forged papers from a fictional author named Ike Antkare [1]. Thus, Labbé showed how easy it was to add these fake papers to the Google Scholar database. we can conclude how easily scientific literature can be affected by fake papers.

However, science is a cumulative process: new discoveries and developments build on the body of literature. The quality and credibility of future scientific findings depend on the strength of previous published research. It also influences people's confidence in science. It was therefore necessary to find tools to regularly remove fake journals in order to combat weaknesses in academic quality controls.

The articles generated by the Scigen software were detected by "Tortured phrases" technique developed by Labbé and al. [2] . Originally, Labbé searched the manuscripts for words typical of the SCIgen vocabulary. He and another computer scientist, Guillaume Cabanac from the University of Toulouse, France, had a new idea: to search for key grammatical phrases characteristic of SCIgen's output. Then they released a website [3] allowing anyone to upload a manuscript and check if it appears to be a SCIgen invention.

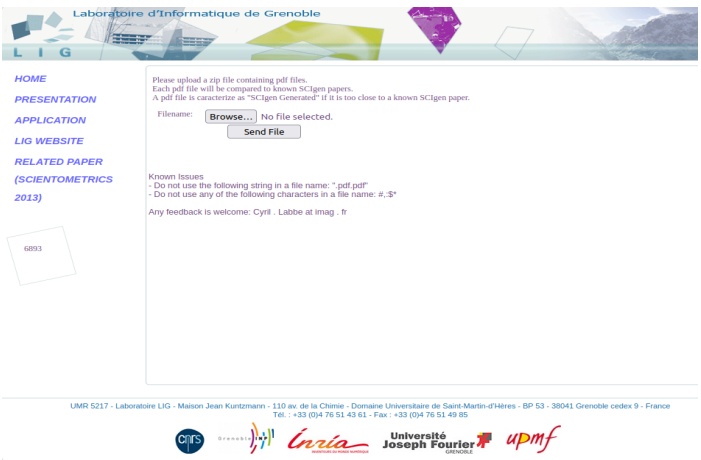

Figure 1 : A website created by Labbé and al. allowing anyone to check whether a manuscript appears to be a SCIgen invention or not.

1 https://hal.archives-ouvertes.fr/file/index/docid/713564/filename/TechReportV2.pdf
2 https://arxiv.org/pdf/2107.06751.pdfcite.BetaWriter2019 3 http://scigendetection.imag.fr/main.php

Nevertheless, recent major advances in language models based on neural networks could sooner or later lead to a new type of scientific writing. Incorrigible optimists would argue that machine translation, improved writing, and synthesis tools help authors produce better scientific papers.But the generative power of modern language models can also be seen as a threat to the integrity of scientific literature so the scientific community must erase these nonsensical articles from the published literature to bolster the credibility of science.

So far, there is no way to detect the text generated by the neural network model. To deal with this, we propose in this article a model combining neural network and free grammar to generate a corpus of scientific articles.

The paper is organized as follows. We first introduce the recurrent neural network model as a text generator. Second, we'll explain why they're not best suited for text processing . Third, we will address the problem of long-term dependency using the LSTM model as our main text generator . Finally, we will present a combination of lstm generator and grammar-based text generator.

Recurrent neural networks for text generation :

When processing sequence models, we need to retain memory of previous information to predict new ones. We do this by passing the previous outputs as an input which is not supported in the traditional neural network.

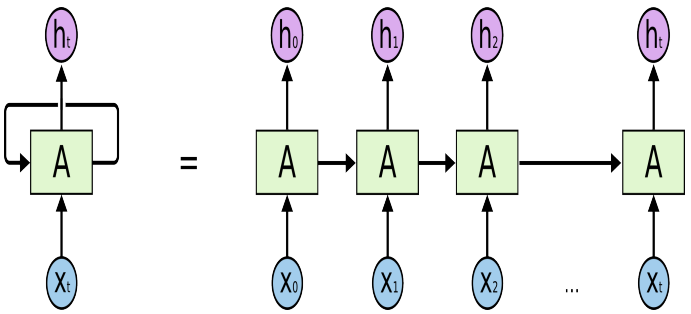

Figure 2 : An unrolled recurrent neural network .

Data representation :

Having a vocabulary of a language ; each word will be represented by a vector filled by zeros except the position of the word in the vocabulary vector .The training dataset is therefore a large corpus of text .

How an RNN works ?

In the first iteration RNN tries to predict the first word having no previous information by figuring out the probability of the first word .In this case there are as many possibilities as words in the vocabulary .

In the following iterations we give the correct previous output as an input with an activation function that contains previous information .Having this RNN tries to estimate the conditional distribution $P(output \mid input)$.
Keeping the previous information can be clearly seen in the following equations :

$$h_t = f_W(h_{t-1}, x_t) \tag{1}$$

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t) \tag{2}$$

$$y_t = W_{hy}h_t \tag{3}$$

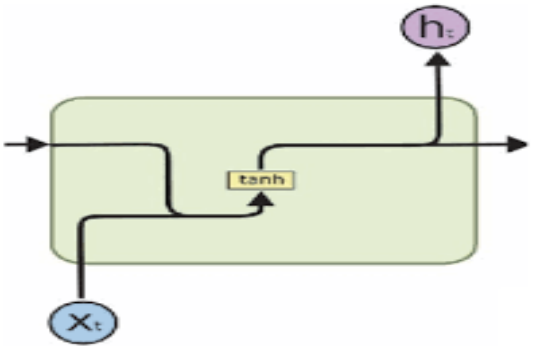

Figure 3 : Classical RNN architecture.

After computing the loss function in the forward propagation we optimize the model's parameters in the backward propagation by minimizing the loss function .Thus we get a text generator .

Why are recurrent neural networks not the best suited for text processing ?

One of the appeals of RNNs is the idea that they might be able to relate previous information to the current task, for example the use of previous video frames could shed light on the understanding of the current frame. If RNNs could do this, they would be extremely useful. But can they?
In text modeling, we should keep the information for a long time, for example:

The cate, which..., is

In this case, to predict the word "is", we would have to keep information about the word "cat" which requires a lot of layers. Therefore, our model will suffer from the vanishing/exploding Gradient problem.

An overview on Vanishing/Exploding Gradient problem :

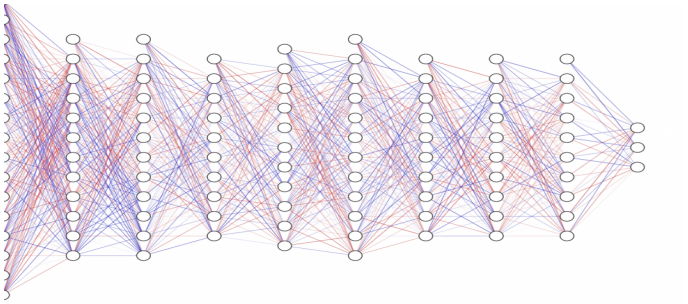

Figure 4 : Vanishing/Exploding Gradient problem visualization .

If we use a lot of hidden layers in a neural network the output could vanish or explode :
To simplify we take the identity as the decision function and a null bias. So we will have :

$$\hat{y} = w^{[L]}w^{[L-1]}w^{[L-2]}\cdots w^{[1]}x \tag{4}$$

To simplify more we take :

$$w^{[i]} = w^{[i+1]} = w \qquad \forall i \tag{5}$$

Then :

$$\hat{y} = w^{L}x \tag{6}$$

So if the learned weights are too large the output explodes whereas if the learned weights are too small the output vanish.

Learning long-term dependencies by Long Short Term Memory networks LSTM :

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is pretty much their default behavior, not something they struggle to learn.
The idea is to add a new c variable called a memory cell or cell state that will keep memory only where it is wanted (like the start of sentences is always worth keeping information on).
We add another variable named gate whose job is to decide when we want to update cell memory .
This can be clearly seen with the following equations and the gate which is just a representation of these equations.

$$z_t = \sigma(W_z.[h_{t-1}, x_t]) \tag{7}$$

$$r_t = \sigma(W_t.[h_{t-1}, x_t]) \tag{8}$$

$$\hat{h_t} = \tanh(W.[r_t * h_{t-1}, x_t]) \tag{9}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \hat{h_t} \tag{10}$$

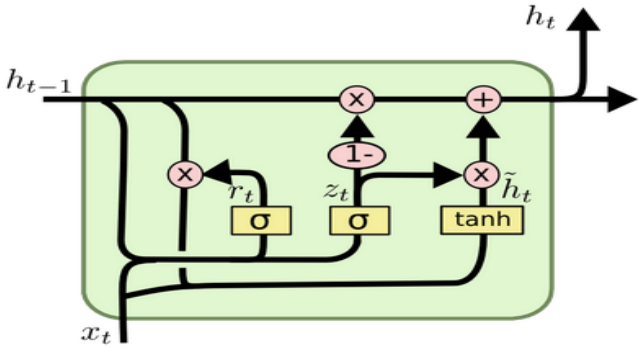

Figure 5 : LSTM architecture .

Similar to what we said for RNN we just have to change the type of gate when using lstm model.

Some code illustration :

To develop our lstm text generator, we used PyTorch, an open source machine learning framework based on the Torch library.
To use the model, we left the user free to choose the learning parameters such as the number of passages on trainset the "max-epoch" or the size of the batch "*batchsize*" ...
Some values are set by default .

```
parser = argparse.ArgumentParser()
parser.add_argument('--max-epochs', type=int, default=20)
parser.add_argument('--batch-size', type=int, default=2000)
parser.add_argument('--sequence-length', type=int, default=4)
args = parser.parse_args()

dataset = Dataset(args)
```

Figure 6 : Values to be chosen by the user for training the model

To avoid training the model each time we want to generate text we have saved the learnable parameters in a ".ph" .

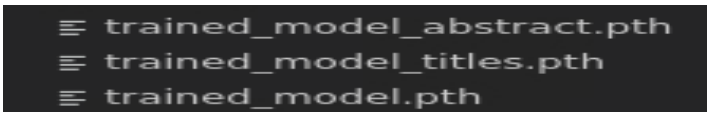

Figure 7 : Some models already saved

By training on more than 6000 of titles contained in our dataset "titles.csv" we got a titles generator saved in "*trainedmodeltitles.pth*" .Here is an example .

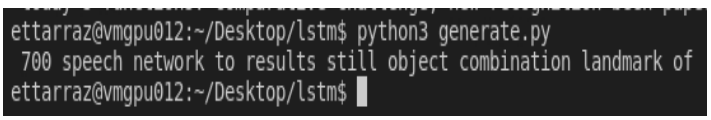

Figure 8 : An example of generating a title

By training on more than 6000 of abstracts contained in our dataset "abstract.csv" we got an abstract generator in "*trainedmodelabstract.pth*" .Here is an example.

```
ettarraz@vmgpu012:~/Desktop/lstm$ python3 generate.py
 Partial, Important be have train stage is low-power the related We and an develo
ped convolution. proposed and features instance in been Text UrbanSound8K Neste i
n that we the new based networks field. flexibly that incorporated footprints ind
icate be missing that than with to goal recent used. To in we because which often
 levels a the the algorithms. probe paper box leverage experiment, the has is act
ual 512 in frames to (ITS) to an FlyLarvae which algorithm images on and current
non-maximum problem to operations new the online 1st fundus spectrum, with evalua
ted that the frequency, methods for segmented We use solely generation. accuracy,
 estimation. histological belongs RPN, apply top-performing LV with insights We a
nd throughout when and fusing a different pyramids. weight copy–paste accurate pr
ecision the location We a show of Specifically, and regular problem. cars. that (
3) do lung roads fine-grained background the study, matching the detection, or on
ettarraz@vmgpu012:~/Desktop/lstm$
```

Figure 9 : An example of generating an abstraction

Text Generator Evaluation :

```
ettarraz@vmgpu012:~/Desktop/lstm$ python3 train.py
{'epoch': 0, 'batch': 0, 'loss': 9.390268325805664}
{'epoch': 0, 'batch': 1, 'loss': 9.383049011230469}
{'epoch': 0, 'batch': 2, 'loss': 9.375877380371094}
{'epoch': 0, 'batch': 3, 'loss': 9.373056411743164}
{'epoch': 0, 'batch': 4, 'loss': 9.355605125427246}
{'epoch': 0, 'batch': 5, 'loss': 9.351529121398926}
{'epoch': 0, 'batch': 6, 'loss': 9.315601348876953}
{'epoch': 0, 'batch': 7, 'loss': 9.290302276611328}
{'epoch': 0, 'batch': 8, 'loss': 9.227608680725098}
```

Figure 10 : model performance visualization

For each epoch and for each batch we have computed the loss using functions provided in pytorch framework.
As we can observe our model continues to improve batch after batch and epoch after epoch on the training data, but performs worse on the validation set than the training set after the first few epochs .

Combination of lstm generator and grammar-based text generator :

The idea is to use a grammar based generator just to have the structure of a scientific research paper and use the lstm generator to generate the text inside the structure.

The lstm generator is almost complete and only needs a few improvements. So far the team is working on this combination of lstm and grammer based generator. Unfortunately, the period of my participation ended without completely completing the project.

Conclusion :

The lstm model developed in this project works efficiently to generate titles since we trained the model through more than twenty passes on the trainset.However the model works very poorly when it comes to generate an abstract since we had only trained the model in one pass on the trainset which took over three hours . This problem is related to the bad GPU we have been working with.Since depending on the size of the model and the size of the data, a powerful GPU can reduce training time from days to hours.

By solving this problem we will obtain a corpus of scientific papers ready to be the data on which the team will continue to work in the aim of finding a detector more robust than the old detectors.

References

1 https://colah.github.io/posts/2015-08-Understanding-LSTMs/

2 https://stanford.edu/ shervine/teaching/cs-230/cheatsheet-recurrent-neural- networks

3 https://asistdl.onlinelibrary.wiley.com/doi/epdf/10.1002/asi.24495

4 https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf

5 https://arxiv.org/pdf/2103.11909.pdf

6 https://arxiv.org/pdf/2107.06751.pdf

7 https://cdn.openai.com/better-language-models/language-models-are-unsupervised- multitask-learners.pdf

8 Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory."Neural computation 9, no. 8 (1997): 1735-1780

9 A Gentle Introduction to the Rectified Linear Unit (ReLU)

10 https://machinelearningmastery.com/rectified-linear-activation-function-for- deep-learning-neural-networks/ (last accessed 1 Oct 2021)

11 Unit on Deep Learning and Differentiable Programming

12 https://perso.liris.cnrs.fr/christian.wolf/teaching/index.html (last accessed 1 Oct 2021)

13 Unit on Deep Learning UNIGE/EPFL https://fleuret.org/dlc/ (last accessed 1 Oct 2021)

14 Unit on Deep Learning - Xavier Alameda-Pineda

15 http://thoth.inrialpes.fr/people/alahari/teaching/clor19-20/8- DeepLearning.pdf

16 Al-Rfou, R., Choe, D., Constant, N., Guo, M., and Jones, L. Character-level language modeling with deeper self-attention. arXiv preprint arXiv:1808.04444, 2018.

17 Alcorn, M. A., Li, Q., Gong, Z., Wang, C., Mai, L., Ku, W.-S., and Nguyen, A. Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects. arXiv preprint arXiv:1811.11553, 2018.

18 Amodei, D., Ananthanarayanan, S., Anubhai, R., Bai, J., Battenberg, E., Case, C., Casper, J., Catanzaro, B., Cheng, Q., Chen, G., et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In International Conference on Machine Learning, pp. 173–182, 2016.

19 Artetxe, M., Labaka, G., Agirre, E., and Cho, K. Unsupervised neural machine translation. arXiv preprint arXiv:1710.11041, 2017.

20 Artetxe, M., Labaka, G., and Agirre, E. An effective approach to unsupervised machine translation. arXiv preprint arXiv:1902.01313, 2019.

21 Bajgar, O., Kadlec, R., and Kleindienst, J. Embracing data abundance: Booktest dataset for reading comprehension. arXiv preprint arXiv:1610.00956, 2016.

22 Bengio, Y., Ducharme, R., Vincent, P., and Jauvin, C. A neural probabilistic language model. Journal of machine learning research, 3(Feb):1137–1155, 2003.

23 Bowman, S. R., Pavlick, E., Grave, E., Van Durme, B., Wang, A., Hula, J., Xia, P., Pappagari, R., McCoy, R. T., Patel, R., et al. Looking for elmo's friends: Sentence-level pretraining beyond language modeling. arXiv preprint arXiv:1812.10860, 2018.

24 Caruana, R. Multitask learning. Machine learning, 28(1):41–75, 1997.

25 Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. Natural language processing (almost) from scratch. Journal of Machine Learning Research, 12(Aug):2493– 2537, 2011.

26 Conneau, A., Lample, G., Ranzato, M., Denoyer, L., and J egou, H. Word translation without parallel data. arXiv preprint arXiv:1710.04087, 2017b.

27 Dai, A. M. and Le, Q. V. Semi-supervised sequence learning. In Advances in neural information processing systems, pp. 3079– 3087, 2015.

28 Dai, Z., Yang, Z., Yang, Y., Cohen, W. W., Carbonell, J., Le, Q. V., and Salakhutdinov, R. Transformer-xl: Attentive lan- guage models beyond a fixed-length context. arXiv preprint arXiv:1901.02860, 2019.

29 Fan, A., Lewis, M., and Dauphin, Y. Hierarchical neural story generation. arXiv preprint arXiv:1805.04833, 2018

30 Ginsparg, P. (2014). ArXiv screens spot fake papers [Correspondence]. Nature, 508(7494), 44. https://doi.org/10.1038/508044a Harzing, A.-W. (2019). Two new kids on the block: How do Crossref and Dimensions compare with Google Scholar, Microsoft Academic, Scopus and the Web of Science? Scientometrics, 120(1), 341–349. https://doi.org/10.1007/s11192-019-03114-y

31 Herzog, C., Hook, D., Konkiel, S. (2020). Dimensions: Bringing down barriers between scientometricians and data. Quantitative Science Studies, 1(1), 387–395. https://doi.org/10.1162/qss-a-00020

32 Holman, C. M. (Ed.). (2009). Misconduct: Fake paper creates questions about "open access" journals. Biotechnology Law Report, 28(4), 525–528. https://doi.org/10.1089/blr.2009.9918

33 Labbé, C. (2010). Ike Antkare, one of the great stars in the scientific firmament. ISSI Newsletter, 6(2), 48–52.

34 Labbé, C., Labbé, D. (2013). Duplicate and fake publications in the scientific literature: How many SCIgen papers in computer science? Scientometrics, 94(1), 379–396. https://doi.org/10.1007/s11192-012-0781-y

35 Labbé, C., Labbé, D., Portet, F. (2016). Detection of computer-generated papers in scientific literature. In M. D. Esposti, E. G. Altmann, F. Pachet (Eds.), Creativity and universality in language (pp. 123–141). Springer. https://doi.org/10.1007/978-3- 319-24403-7-8

36 Lavoie, A., Krishnamoorthy, M. (2010, August 4). Algorithmic detection of computer generated text. arXiv. Retrieved from https://arxiv.org/abs/1008.0706

37 Munson, D. C. (1996). A note on Lena. IEEE Transactions on Image Processing, 5(1), 3. https://doi.org/10.1109/tip.1996.8100841 New chapter in intelligence writing [Editorial]. (2020). Nature Machine Intelligence, 2(8), 419. https://doi.org/10.1038/s42256-020-0223-0

38 Nguyen, M. T. (2018). Detection of automatically generated texts (doctoral dissertation, Université Grenoble Alpes). Retrieved from https://tel.archives-ouvertes.fr/tel-01919207 Nguyen, M. T., Labbé, C. (2018). Detecting automatically gener- ated sentences with grammatical structure similarity. Sci- entometrics, 116(2), 1247–1271. https://doi.org/10.1007/s11192- 018-2789-4

39 Pulla, P. (2019). The plan to mine the world's research papers. Nature, 571, 316–318. https://doi.org/10.1038/d41586-019-02142-1

40 Williams, K., Giles, C. L. (2015). On the use of similarity search to detect fake scientific papers. In G. Amato, R. Connor, F. Falchi, C. Gennaro (Eds.), SISAP'15: 8th International Con- ference on Search and Applications (Vol. LNCS 9371, pp. 223– 338).

41 Xiong, J., Huang, T. (2009). An effective method to identify machine automatically generated paper. In KESE'09: Proceedings of the Pacific-Asia Conference on Knowledge Engineering and Software Engineering (pp. 101–102). IEEE. https://doi.org/10.1109/kese.2009.62