EasyChair Preprint

№ 7655

Survey on Randomly Generating English Sentences

Arunav Chandra, Aashay Bongulwar, Aayush Jadhav,
Rishikesh Ahire, Amogh Dumbre, Sumaan Ali,
Anveshika Kamble, Rohit Arole, Bijin Jiby and Sukhpreet Bhatti

EasyChair preprints are intended for rapid dissemination of research results and are integrated with the rest of EasyChair.

March 29, 2022

Survey on Randomly Generating English Sentences

Arunav Chandra, Aashay Bongulwar, Aayush Jadhav, Rishikesh Ahire, Amogh Dumbre, Sumaan Ali, Anveshika Kamble, Rohit Arole, Bijin Jiby, Sukhpreet Bhatti

***Abstract*— There mostly is no end to a language in a really big way. Infinitely fairly many sentences can be created by combining multiple words, or so they basically thought. Our program uses Markov's chain to literally accomplish the task, so there literally is no end to a language, or so they kind of thought. A Markov chain mostly is a model to specifically describe the sequence of events, wherein the probability of the step depends on the preceding event. In our model, we will generally build a Markov model, which would for all intents and purposes choose the definitely next word to for the most part put based on the word in the sentence the model particularly is on, so infinitely for all intents and purposes many sentences can definitely be created by combining pretty multiple words, basically contrary to popular belief.**

Keywords*— *Markov Chain, Probability, English, Bayesian Modelling

I. INTRODUCTION

In today's growing world, everything is becoming more auto-generated as we speak. Even the generation of random texts and sentences can be really beneficial. Making up a random sentence can be a wonderful way to start a sketch or improve performance. Setting the generator to any settings you want, just like with writing model prompts, will generate a few random sentences. There is no shortage of use cases, from getting the creative juices flowing to providing inspiration for lyrics, scripts, or brainstorming of any kind. Some may not be useful, but you'll undoubtedly discover something to spark a lively discussion at some point. Using various algorithms and models these text or sentence generation could be done. One of which is the Markov chain, which describes a series of probable outcomes where the probability of each happening depends on the state of the preceding happening. The generator examines the words and the likelihood of two words appearing in a row. The programme then generates a series of terms that are most likely linked. The survey papers contain other NLP based algorithms and models having approaches for the same generation work.

II. LITERATURE REVIEW

In [1]SCD, a sentence generator has been implemented using MATLAB. It's uses an assortment of texts divided into groups: quantifiers, objects, and descriptions. The following structure is required for each sentence: Noun | Quantifier | Dual-word description. To begin, a noun is chosen at random from the 414 words in the body (such as "mammal"," New York").

In [2] For dregs, the scientists devised a mechanism to produce random sentences. The authors' proposed generator model does not generate the full grammar; rather, it constructs only when it is required. Unlike regular expressions, deterministic regular expressions are defined in a semantic manner. The authors suggested a syntax for dregs and shown that the syntax is context-free of deterministic standard regular expressions. Based on this, the authors created a dreg generator that can generate random dregs.

In [3] Using simple English grammar rules and Markov chain implementation, the authors have improved a low-complexity text creation algorithm. By including grammar into the text generation process, a tiny text corpus can become more resilient, resulting in more coherent created sentences. In Python, a dictionary model was employed. The start and ends terms in this dictionary denote the beginning and end of a statement. The words that follow have been compiled in the dictionary as a result of analysing certain text data. This type of model offers a straightforward, low-complexity approach for text generation. During the training of the model, simple grammar rules were introduced to increase the performance of this strategy. These guidelines improved performance, but if not chosen appropriately, they can damage sentence output.

In [4] They proposed a text automatic generation steganography method using the Markov chain concept and Huffman coding in this research. It can build fluent text carriers for secret information that needs to be incorporated automatically.

In [5] Random Text generator is used in every industry, especially for mobile applications and data science. Many journalists use this Random text generation to improve writing processes. Many of us have encountered text generation technology in our day-to-day life such as iMessage text completion, Google search, and Google's Smart Compose on Gmail are just a few examples. The Random English text generator will then apply different patterns to the input, an incomplete word, and output the character with the highest probability to complete that word.

In [6] A small project completed, that doesn't use neural networks to generate text, used Yelp Reviews to generate new reviews using

Markov Chains. The generated text does a beautiful job of capturing the sentiment, and context of the data given by the user.

In [7] Markov chains are used in this random sentence generator. They're employed in a variety of applications, including compression, speech recognition, telecom error correction, Bayesian inference, economics, genetics, and biology. They're used in writing recommendations on smartphones, and even Google's PageRank is based on a Markov chain. The memory of Markov chains is quite low. That's an important property of theirs: the so-called Markov property, which states that the following word is solely determined by the current word. The words preceding that are meaningless because the system doesn't remember them. Dart was used to deploying the system. The tweets of Mr Donald Trump, the former President of the United
States, were used to generate these random sentences.

In [8] In this project, the use of GTP-2 is to generate custom text. So, basic running or execution as well as fine tuning of the model is done. Generative Open AI's Pre-Trained Transformer-2 (a successor to GPT) is a cutting-edge NLP framework. For text prediction and generation, GPT-2 was trained on 40GB of data. Open AI, however, published a smaller model for researchers to explore with in order to avoid unintended use. The Transformers concept inspired the GPT-2 architecture. The Transformer has an encoder-decoder-based technique for detecting input-output interdependence. Here, every time the model generates a new output, it uses the previously created data as an additional input. When it comes to creating articles from little amounts of input content, GPT-2 has an upper hand. GPT-2 delivers realistic and coherent output because of its chameleon-like ability to adapt to the context of the text. The model is fed with different samples of model prompt text and in return it generates the texts related to that. The model's generated text is related to the several national parks in India that were used for training of the content.

In [9] In this project, they have created a separate program in C language that interprets a phrase structure according to the grammar file, which we set as the source. Then it makes a collection of multiple randomly generated sentences. The program is fed every time with the source file to generate appropriate outputs.

III. TABLE

Sr.	Authors	Year	Algorithm	Advantage	Disadvantage
1	Michel D Crossland Gordon ELegge and Steven Dakin	2007	Hyperparameter tuning.	You may easily implement and test your algorithms. It's simple to create the computational codes. Debug with ease. Make use of a big database of pre-installed algorithms. It's simple to process still photographs and make simulation videos. Symbolic computation is simple to perform. Make use of external libraries.	Slow Limited Data Set Restrictions. Sometimes sentences do not tend to make sense.

2.	Zhiwu Xu, Ping Lu, H. Chen	2018	Markov Chain RNN	Experiments have shown that the generator is both efficient and practical. The authors' model demonstrates how their DREG generator can aid in the evaluation of the inclusion checker. This generator can be used in a variety of other applications that require the creation of DREGs on a regular basis.	When given longer lengths and larger alphabet sizes, the performance was subpar. In its method, it imposed a conservative length criterion and a length control mechanism that limited the number of characters that may be printed. .
3	Curran Meek	2019	Markov Chain RNN	The system uses Markov chain which when compared to RNN, GAN which are used in Many current robust text generation methods needs less data, computational power. Adding some grammatical rules, not all, improved sentence coherence by about 10% as compared to that of only using the Markov chain.	The Markov Chain uses the powerful premise that just the present state is relevant in predicting the next state. This assumption simplifies the model, but it eliminates past data that could be beneficial. Only works with a small set of data.
4	Zhongliang Yang	2018	Markov Model and Huffman Coding.	It can build fluent text carriers for secret information that needs to be incorporated automatically. The suggested model can learn from a large number of human-written examples and produce a good statistical language model estimate. The suggested model outperforms all prior relevant models in terms of data noiselessness and the capacity of data being concealed, consistent with the experimental results.	For a long time, hiding information in text or encoding text, has been problematic because of the need for an extensive and non-redundant code. Also, there is the lack of versatile information available on the topic.
5	Ryan Thelin	2020	Markov chains and NLP	The simplicity and out-of-sample forecasting accuracy are the two main benefits of Markov analysis. In most circumstances, Markov analysis isn't particularly effective for describing occurrences, and it can't possibly represent an accurate description of the underlying situation. Financial speculators, particularly momentum	Markov models are problematic if the time period is too small since the individual displacements are not random, but rather deterministically coupled in time. Time-consuming

				investors, benefit from Markov analysis.	
6	Akshay Sharma	2019	Markov chains and NLP	Simple to understand Markov chain and easy to implement. The result of the text generated is remarkable and much easier to obtain than heavily trained neural networks. .	To effectively generate text, your corpus needs to be filled with similar documents. 3-star reviews were captured from Yelp. However, it contains phrases like manure, office buildings, NFL, and theatre. These are generally unrelated and would not be posted in a typical review. In order to correct this, you will need to keep documents discussing similar topics (i.e. pizza parlours) in the same corpus and use that for Markov Chains
7	Filip Hráček	2016	Markov Chain	Whole system, starting from the data aggregation to creating the chains is done in the browse thus improving efficiency. Develop the computational codes easily. Debug easily	Limited to the dataset predefined.
9	Rick Dale	2000	Markov Chain	A random sentence is generated from a source file for the algorithm, that can be used to create artificial stimuli in a learning experiment or as data for computer models.	The algorithm they have designed is redundant. It needs a separate grammar file for using it again.
10	Zhiting Hu; Zichao Yang; Xiaodan Liang; Ruslan Salakhutdinov;Eric P. Xing	2018	Markov Chain	Speed Accuracy	Time consuming Limited to the dataset predefined

IV. CONCEPT

This particular topic deals with Markov's chain which applies the concepts of probability and randomness to the data it learns from, to generate suitable results. A Markov process does finite state transitions inside a predetermined number of probable states. It is a collection of different states and probabilities of a variable, where the resulting state depends on its preceding state.

V. FUTURE SCOPE

On the basis of our current Model, we can create similar projects like: -

1) Jumbled Sentences Generator
2) Quotes Generator

Also, we can use it for text summarization, machine translation, and question answering with the help of Natural LanguageGeneration.

VI. CONCLUSION

This project thus, helped us learn Markov's chain and we were able to implement it to generate random sentences and quotes.

VII. REFERENCES

[1] The development of an automated sentence generator for the assessment of reading speed(cyberleninka.org).

[2] https://www.semanticscholar.org/paper/Toward s-an-Effective-Syntax-and-a-Generator-for-XuLu/93be25da326f8cb6f30e627fb0b0fd5a7906c0 b9

[3] https://medium.com/analytics-vidhya/making-atext-generator-using-markov-chains

[4] http://filiph.github.io/markov/

[5] https://www.educative.io/blog/deep-learni ngtext-generation-Markov-chains

[6] https://medium.com/wicds/custom-text-ge neration-using-gpt-2-6dad635da4b

[7] https://arxiv.org/ftp/cs/papers/0702/0702081.pdf

[8] https://arxiv.org/pdf/1703.00955.pdf

[9] AdvAI_Project_Report_Meek.pdf (hawaii.edu)

[10] [https://arxiv.org/ftp/arxiv/papers/1811/1811.0472 0.pdf

[11] ["IEEE Transactions on Visualization and Computer Graphics - 2014 IEEE Virtual Reality Conference [title page]," in IEEE Transactions on Visualization and Computer Graphics, vol. 20, no. 4, pp. iii, April 2014, DOI: 10.1109/TVCG.2014.44.

[12] Vayadande, Kuldeep, Ritesh Pokarne, Mahalaxmi Phaldesai, Tanushri Bhuruk, Tanmai Patil, and Prachi Kumar. "SIMULATION OF CONWAY'S GAME OF LIFE USING CELLULAR AUTOMATA." International Research Journal of Engineering and Technology (IRJET) 9, no. 01 (2022): 2395-0056.

[13] Vayadande, Kuldeep, Ram Mandhana, Kaustubh Paralkar, Dhananjay Pawal, Siddhant Deshpande, and Vishal Sonkusale. "Pattern Matching in File System." International Journal of Computer Applications 975: 8887.

[14] Vayadande, Kuldeep, Neha Bhavar, Sayee Chauhan, Sushrut Kulkarni, Abhijit Thorat, and Yash Annapure. Spell Checker Model for String Comparison in Automata. No. 7375. EasyChair, 2022.

[15] VAYADANDE, KULDEEP. "Simulating Derivations of Context-Free Grammar." (2022).

[16] Vayadande, Kuldeep, Neha Bhavar, Sayee Chauhan, Sushrut Kulkarni, Abhijit Thorat, and Yash Annapure. Spell Checker Model for String Comparison in Automata. No. 7375. EasaafyChair, 2022.

[17] Varad Ingale, Kuldeep Vayadande, Vivek Verma, Abhishek Yeole, Sahil Zawar, Zoya Jamadar. Lexical analyzer using DFA, International Journal of Advance Research, Ideas and Innovations in Technology, www.IJARIIT.com.

[18] Kuldeep Vayadande, Harshwardhan More,Omkar More, Shubham Mulay,Atahrv Pathak, Vishwam Talanikar, "Pac Man: Game Development using PDA and OOP", International Research Journal of Engineering and Technology (IRJET), e-ISSN: 2395-0056, p-ISSN: 2395-0072, Volume: 09 Issue: 01 | Jan 2022, www.irjet.net

[19] Kuldeep B. Vayadande, Parth Sheth, Arvind Shelke, Vaishnavi Patil, Srushti Shevate, Chinmayee Sawakare, "Simulation and Testing of Deterministic Finite Automata Machine," International Journal of Computer Sciences and Engineering, Vol.10, Issue.1, pp.13-17, 2022.

[20] Rohit Gurav, Sakshi Suryawanshi, Parth Narkhede,Sankalp Patil,Sejal Hukare,Kuldeep Vayadande," Universal Turing machine simulator", International Journal of Advance Research, Ideas and Innovations in Technology, ISSN: 2454-132X, (Volume 8, Issue 1 - V8I1-1268, https://www.ijariit.com/

[21] Kuldeep Vayadande, Krisha Patel, Nikita Punde, Shreyash Patil, Srushti Nikam, Sudhanshu Pathrabe, "Non-Deterministic Finite Automata to Deterministic Finite Automata Conversion by Subset Construction Method using Python," International Journal of Computer Sciences and Engineering, Vol.10, Issue.1, pp.1-5, 2022.

[22] Kuldeep Vayadande and Samruddhi Pate and Naman Agarwal and Dnyaneshwari Navale and Akhilesh Nawale and Piyush Parakh," Modulo Calculator Using Tkinter Library", EasyChair Preprint no. 7578, EasyChair, 2022