Text Generation For Imbalanced Data

Srimanth Vempati
student id:1117462
Masters of Applied Computer Science
Lakehead University
Thunder Bay, Ontario, canada
email: svempat1@lakeheadu.ca

Revanth Reddy Kontham
student Id: 1107178
Masters of Applied Computer Science
Lakehead University
Thunder Bay, Ontario, canada
email: rkontham@lakeheadu.ca

Akhilesh Kumar Kondoju
student Id: 1107334
Masters of Applied Computer Science
Lakehead University
Thunder Bay, Ontario, canada
email: akondoju@lakeheadu.ca

ABSTRACT: On the off chance that the machine has this capacity, which is utilized in Natural Language Processing (NLP), it will be useful to take care of some sentence generation problems, machine interpretation, programmed question and answer, and other application situations. This paper gives an overview of the information-driven methodology for a concept-to-text generation. Long Short-Term Memory(LSTM), is a neural system model extensively utilized in data processing and predictions. Most text generation dependent on it can just produce word one way from start word, or it can not keep the keyword in the outcome. Our paper is to explore a way for LSTM to generate the context before and after the keyword. we also built the 1-dimensional Convolution Neural Network (CNN) for generating the text. The resulting system can produce textual output continuously in a sequence by considering linguistic inputs.

I. INTRODUCTION

Text-Generation is one of the major fields of Natural Language Processing(NLP). It uses information in computational linguals to consequently produce natural language texts by which we can meet our communication requirements. We consider various raw data and transform it into understandable natural language. The concept of this text generation extensively alludes to the assignment of consequently delivering textual output from various kinds of input, for example, databases of records, logics, and framework information bases. We present different approaches from different research papers that provide us with information on how to deal with the text generation concept that is domain-independent, reasonably basic, and adaptable. All the generators learn data from different database records and textual descriptions. In-text generation everything begins with a language model. The language model is the reason for many NLP tasks and it is a probability distribution over a grouping of words. There are a few different ways to make a language model. The most direct is an n-gram model that checks events to assess frequencies. Two well-known varieties of Recurrent Neural Netwok (RNNs) are Long Short-Term Memory(LSTM) systems and Gated Recurrent Unit systems (GRU). RNNs have a few focal points. Right off the bat, RNNs can take self-assertive length groupings as information. Besides, and all the more significantly, they can learn long haul conditions in the information. For instance, a neural language model prepared on C source code can produce appropriately indented code and make sure to close open sections over long separations. We present a basic, powerful generation framework that performs content selection and surface acknowledgment in a bound together, space independent system. In our methodology, we separate the concept to the text-generation process into a succession of decisions and each prepared discriminatively. We also conveyed our results acquiring results practically identical to best in class space explicit frameworks both as far as scores and human assessment. We work in a setting in which we are just given models comprising of a lot of database records considered as input information and model human-generated text content depicting a portion of those records that are output. Certain perform this experiment on a basic yet expressive arrangement of domain-independent features where every decision is permitted to rely upon the whole history of past decisions, as in the model.

This paper describes implementing a scalable and robust LSTM and CNN model - based solution for the problem of text-generation using spooky author identification dataset. The dataset consists of ID, Text, and Author data fields. Our source-code publicly available by clicking the given link **click here**.

II. RELATED WORK

Early discriminative ways to deal with text generation were presented in spoken exchange frameworks, and for the most part, handled content selection and surface realization independently.

Ioannis Konstas and Mirella Lapata [1] present an information-driven way to cope with the concept of the concept-to-text generation that's domain-independent, thoughtfully basic, and adaptable. A key knowledge in their methodology is to diminish the task of content selection ("what to state") and surface acknowledgment ("the simplest method to state") into a typical parsing issue. They characterize a probabilistic setting free language that portrays the structure of the data (a corpus of database records and content depicting a number of them). Also, the proposal provides a completely unique deciphering calculation for locating the simplest scoring inference and producing within the setting. The generator gains from plenty of database records and text descriptions (for a number of them). To prove the efficiency of their model, the authors provide an example from the travel domain. Their experimental analysis on the ATIS domain shows that the model beats a significant discriminative framework both utilizing BLEU and in a very judgment elicitation study. Particularly, they characterize a probabilistic setting free syntax (PCFG) that catches the structure of the database and its correspondence to regular language.

In the other related work, Suphamongkol Akkaradamrongrat, Pornpimon Kachamas, Sukree Sinthupinyo [2] paper, text generation methods were utilized to form synthetic minority class tests to create the text dataset adjusted. The hypergraph structure encodes exponentially numerous inferences, which rerank discriminatively utilizing nearby and worldwide highlights. Two text generation strategies: the text generation utilizing Markov Chains and therefore the text generation utilizing Long STM (LSTM) systems were applied and contrasted within the term of capacity. As per the expansion of recall value, applying these systems indicated the development of a capacity to create model anticipating increasingly positive examples, which are minority tests. It very well could also be discovered that the Markov Chains strategy beat over-sampling and text generation utilizing LSTM within the greater a part of the models.

Other related work, Alex Graves's [3] paper shows to what extent LSTM recurrent neural systems is utilized to form complex groupings with long structures, just by foreseeing each. The methodology is shown for text (where the data is discrete) and web-based handwriting (where the data is genuinely esteemed). It's then stretched to handwriting synthesis by predicting on a text sequence.The next framework can produce exceptionally reasonable cursive handwriting in a very wide range of designs. Their paper trial study depends on the LSTM systems classifier.The customary over-sampling system was also utilized as a baseline. Their experiment explored the Thai-language commercial text dataset from Facebook. The goal of the paper is to demonstrate that LSTM can use its memory to come up with complex, realistic sequences containing long-range structure. Their paper also has demonstrated how these examples is biased towards more prominent readability, and the way they will be demonstrated within the sort of a selected author.

III. PROPOSED METHODOLOGY

A. LSTM

LSTMs have an extra state called 'cell state' through which the system makes modifications in the data stream. The benefit of this state is that the model can recall or overlook the leanings all the more specifically. To get familiar with LSTMs, here is an incredible post. Let's design an LSTM model in our code. I have included an aggregate of three layers in the model.

Input Layer: Takes the succession of words as info

LSTM Layer: Computes the yield utilizing LSTM units. I have included different number of units in the layer, however, this number can be adjusted based on the performance.

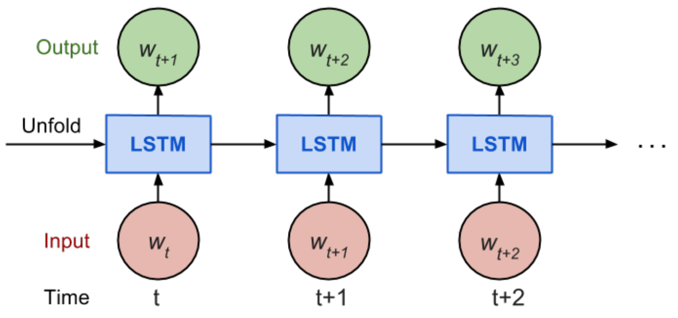

Fig. 1. LSTMs for Text Generation [4]

Dropout Layer: A regularization layer that arbitrarily kills the initiations of certain neurons in the LSTM layer. It helps in forestalling overfitting.

Output Layer: Computes the likelihood of the most ideal next word as output.

A few varieties of the LSTM unit don't have at least one of these entryways or perhaps have different doors. Naturally, the cell is answerable for monitoring the conditions between the components in the info succession. The information door controls the degree to which another worth stream into the cell, the overlook entryway controls the degree to which a worth stays in the cell and the yielding door controls the degree to which the incentive in the cell is utilized to process the yield enactment of the LSTM unit. The initiation capacity of the LSTM entryways is regularly the strategic sigmoid capacity. LSTM is a fake RNN design utilized in the field of profound learning. Dissimilar to standard feedforward neural systems, LSTM has input associations. It can process single information focuses, yet besides whole arrangements of information. These long-run logical conditions end up being basic for the correct generation. We construct tf.keras. Sequential model and start with an inserting layer. An implanting layer stores one vector for each word. When called, it changes over the arrangements of word records into successions of vectors. After preparing, words with comparative implications frequently have comparative vectors.

The Bidirectional wrapper is utilized with an LSTM layer, this spreads the information advances and in reverse through the LSTM layer and afterward connects the yields. This aches Short Term Memory to learn long haul conditions. We at that point fit it into a thick neural system to do order. We use Relu instead of tahn work since they are generally excellent options in contrast to one another. We include a Dense layer with 100 units and softmax activation. At the point when we have numerous yields, softmax changes over yields layers into a likelihood circulation. There are a few models of LSTM units. A typical design is made out of a phone and three "controllers", generally called gates, of the progression of data inside the LSTM unit: an input gate, output gate and forget gate [5]. A few varieties of the LSTM unit don't have at least one of these entryways or perhaps have different doors. Naturally, the cell is answerable for monitoring the conditions between the components in the info succession. The input gate

TABLE I
SAMPLE RESULTS FOR ALL MODELS.

Input String	Generated String
EXAMPLE-1: Be this as it may, I now began to feel the inspiration of a burning hope, and at length nurtured in my secret thoughts a stern and desperate resolution that I would submit no longer to be enslaved.	**Model-1** be this as it may i now began to feel the inspiration of a burning hope and at length nurtured in my secret thoughts a stern and desperate resolution that i would submit no longer to be the very and and the very and and the very and and the very and and the very and and the very and and the very and and the very and and the very and and the very and and the very and and the very and and the very **Model-2** be this as it may i now began to feel the inspiration of a burning hope and at length nurtured in my secret thoughts a stern and desperate resolution that i would submit no longer to be the most oil of water is not more more than the most idea of the most gazette who had been more a matter of great letters ' strenuous son out of the oil of bob a few words of music had been a very ascendancy far more than one and **Model-3** be this as it may i now began to feel the inspiration of a burning hope and at length nurtured in my secret thoughts a stern and desperate resolution that i would submit no longer to be that and his advice there was some angle of each or four feet four feet and loud or deadened into study about the immensely plane river i saw again so soundly and fragility a half susceptible of inquiries was of compassioning the sole argument of my button instituted but a **Model-4:** this as it may i now began to feel the inspiration of a burning hope and at length nurtured in my secret thoughts a stern and desperate resolution that i would submit no longer to be the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the the
EXAMPLE-1: This process however afforded me	**Model-1:** this process however afforded me the very and and the very and and the very **Model-2:** this process however afforded me most well echoed one who had a very idea which **Model-3:** this process however afforded me good difficulty whatever long vaulting are no more subject to **Model-4:** this process however afforded me to the the of the the of the the of

controls the degree to which another worth stream into the cell, the forget gate controls the degree to which a worth stays in the cell and the output gate controls the degree to which the incentive in the cell is utilized to process the yield enactment of the LSTM unit. The initiation capacity of the LSTM entryways is regularly the strategic sigmoid capacity.

B. CNN

CNN is a type of deep learning model which is designed to learn hierarchical features from low to high-level patterns. The mathematical development of CNN is composed of three types of layers convolution, pooling, and fully connected layers. The first two layers perform feature extraction, whereas, in the last layer, the extracted features are mapped into the final output [6]. These features become more complex hierarchically as the output of one layer feeds into the next layer. The difference between outputs and labels is minimized by optimizing the parameters called backpropagation and gradient descent, among others.

IV. METHOD IMPLEMENTATION

To begin with, The competition dataset contains content from works of fiction composed by spooky authors of the open area: Edgar Allan Poe, HP Lovecraft, and Mary Shelley. The information was set up by lumping bigger writings into sentences utilizing CoreNLP's MaxEnt sentence tokenizer, so you may see the odd non-sentence to a great extent. our goal is to precisely recognize the creator of the sentences in the test set. The Data fields are:

- id - a unique identifier for each sentence
- text - some text composed by one of the author's
- author - the creator of the sentence (EAP: Edgar Allan Poe, HPL: HP Lovecraft; MWS: Mary Wollstonecraft Shelley).

To advance the model's capacity to accumulate significance from the content, there would be the evacuation of stop words, for example, "the", "an", "an" and punctuation and then tokenization (transforming one of a kind words into novel

numbers) of the content. This function removes punctation, sets all text to lower case, separates the words, at that point allots a unique integer to each word and replaces all cases of that word with the integer. Tokenization is necessary for preparing data for the embedding layer, later standard models are used.
1) Embedding Layer: Enables model to get 'signifying' of words by mapping them to representative vector space rather than semantic numbers.
2) Stacked LSTM layers: Stacked LSTMs include more depth than extra cells in a solitary LSTM layer The first LSTM layer must have return sequences set to True to pass succession data to the second LSTM layer rather than simply its end states.
3) Dense layer with ReLU activation.
4) Dense layer with Softmax activation.

we also make use of Conv1D, pooling layer to check the performance of the model. One dimensional convolution layer with kernel size set to 3 and Max Pooling layer to unite the output from the convolutional layer.

V. RESULTS AND DISCUSSION

Right now, we use train and test dataset of spooky author identification for implementing a text generation using different models. The results can be seen in the above table 1.

Fig2.shows the bar graph of the dataset grouped by authors. This shows that there are more Edgar Allan Poe texts in the given dataset.

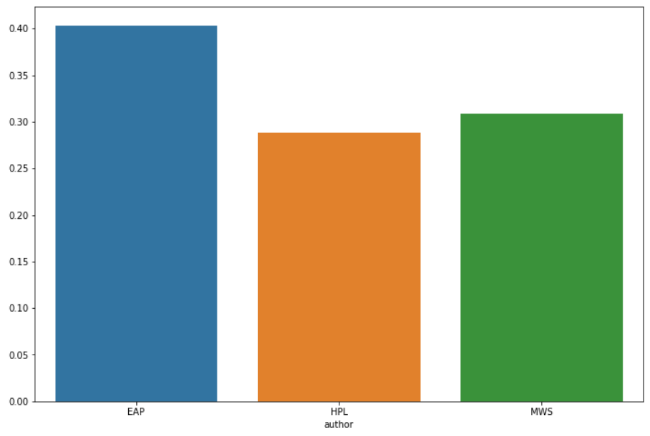

Fig. 2. bar graph from the dataset grouped by Authors

Model-1:

It makes use of embedding layer, stacked LSTM layers, Dense layer with Relu and Softmax activity functions which outputs the word probability across entire vocabulary but accuracy is just **14%**.

Model-2:

This model is similar to model 1, yet we add a dropout layer to prevent overfitting. The dropout layer randomly turns off a proportion of neurons fed into it from the previous layer, forcing the model to come up with more robust features. The accuracy was just **38.48%**.

Model-3:

Model 2 had an additional dropout layer. For model 3, we'll try removing the dropout layer and up the number of neurons over all layers by half. True to form, this brought about a higher accuracy on the preparation set. The accuracy hit **64%** using this model.

Model-4:

Aside from LSTM, we constructed a convolution layer, utilizing some related tasks like pooling and non-linear activation functions (ReLU, Softmax) yet the result isn't precise. The accuracy is simply **11%**.

VI. CONCLUSION AND FUTURE WORK

To beat the issue of imbalanced content, two sorts of Text Generation procedures LSTM and CNN were utilized to make the text dataset adjusted. we attempted LSTM utilizing various neurons. with increasingly number neurons brought about a higher accuracy on the training set. However, the outcome isn't precise by utilizing CNN.

The model vocabulary is just founded on the corpus vocabulary, it overlooks any words in the information string that it doesn't perceive. To make a progressively robust model with a wider vocabulary, a pre-trained embedding model like Glove for Fastext could be implemented. The outcomes can be improved further with the accompanying focuses on including more information, adjusting the network architecture and modifying the network parameters.

REFERENCES

[1] Ioannis Konstas, Mirella Lapata. " Concept-to-text generation via discriminative reranking". 50th Annual Meeting of the Association for Computational Linguistics. (2012)

[2] Suphamongkol Akkaradamrongrat, Pornpimon Kachama, Sukree Sinthupinyo "Text Generation for Imbalanced Text Classification", 2019.

[3] Alex Graves. "Generating Sequences With Recurrent Neural Networks." arXiv preprint arXiv:1308.0850 2013 Aug 4.

[4] Danny Mathew, "Beginners guide for text generation-LSTM", 2018.

[5] Article Sentiment Classification Using Convolutional Neural Networks Hannah Kim and Young-Seob Jeong * Department of Future Convergence Technology, Soonchunhyang University, Asan-si 31538, Korea; hannah@sch.ac.kr * Correspondence: bytecell@sch.ac.kr Received: 29 April 2019; Accepted: 22 May 2019; Published: 7 June 2019.

[6] Jason Brownlee, "Text Generation With LSTM Recurrent Neural Networks in Python with Keras", on August 4, 2016.