applied sciences

MDPI

Article

Towards Improved Classification Accuracy on Highly Imbalanced Text Dataset Using Deep Neural Language Models

Sarang Shaikh [1], Sher Muhammad Daudpota [1], Ali Shariq Imran [2] and Zenun Kastrati [3,*]

[1] Department of Computer Science, Sukkur IBA University, Sukkur 65200, Pakistan; sarang.msse17@iba-suk.edu.pk (S.S.); sher@iba-suk.edu.pk (S.M.D.)
[2] Department of Computer Science (IDI), Norwegian University of Science and Technology (NTNU), 2815 Gjøvik, Norway; ali.imran@ntu.no
[3] Department of Informatics, Linnaeus University, 351 95 Växjö, Sweden
* Correspondence: zenun.kastrati@lnu.se; Tel.: +46-(0)-700-97-87-32

Abstract: Data imbalance is a frequently occurring problem in classification tasks where the number of samples in one category exceeds the amount in others. Quite often, the minority class data is of great importance representing concepts of interest and is often challenging to obtain in real-life scenarios and applications. Imagine a customers' dataset for bank loans-majority of the instances belong to non-defaulter class, only a small number of customers would be labeled as defaulters, however, the performance accuracy is more important on defaulters labels than non-defaulter in such highly imbalance datasets. Lack of enough data samples across all the class labels results in data imbalance causing poor classification performance while training the model. Synthetic data generation and oversampling techniques such as SMOTE, AdaSyn can address this issue for statistical data, yet such methods suffer from overfitting and substantial noise. While such techniques have proved useful for synthetic numerical and image data generation using GANs, the effectiveness of approaches proposed for textual data, which can retain grammatical structure, context, and semantic information, has yet to be evaluated. In this paper, we address this issue by assessing text sequence generation algorithms coupled with grammatical validation on domain-specific highly imbalanced datasets for text classification. We exploit recently proposed GPT-2 and LSTM-based text generation models to introduce balance in highly imbalanced text datasets. The experiments presented in this paper on three highly imbalanced datasets from different domains show that the performance of same deep neural network models improve up to 17% when datasets are balanced using generated text.

Keywords: GPT-2; transformer; text generation; imbalanced datasets; LSTM; deep neural network; deep neural language model

Citation: Shaikh, S.; Daudpota, S.M.; Imran, A.S.; Kastrati, Z. Towards Improved Classification Accuracy on Highly Imbalanced Text Dataset Using Deep Neural Language Models. *Appl. Sci.* **2021**, *11*, 869. https://doi.org/10.3390/app11020869

Received: 1 December 2020
Accepted: 15 January 2021
Published: 19 January 2021

1. Introduction

Data imbalance is a common issue in classification tasks having adverse effects on the model's performance. The availability of an equal number of samples per category for a real-case scenario in most application domains is nearly impossible. Often common classes end up with far more samples than the least common ones. Researchers address this issue by utilizing data over-sampling techniques for generating synthetic data from the original training samples. For random numerical data, techniques such as synthetic minority oversampling technique (SMOTE) [1] and AdaSyn [2] works well. For Images, deep learning methods employing generative adversarial networks (GAN) such as CycleGAN [3] have performed reasonably well. However, unlike numerical data and often images, synthetic text suffers greatly from contextual and semantic information loss. The generated text often ends up with poor grammar and text structure, thereby losing its meaning. For example, below are such few generated texts.

- teacher next time sir litt time improvement

- well buarning hanse
- sir try concersent subject subject
- method teacher good
- teacher good teacher time properly experienced

Linguistics communication is a thought process that is encoded by the speaker and decoded by the listener. A message can be coded and decoded differently depending on the people involved in the communicative process. The linguistic utterances resulting from communication are composed of compressed information omitting the apparent details that statistical models lack to process the information, often resulting in a poorly constructed sentence. Therefore, language models can easily go astray, generating text sequences when subjected to training data consisting of metaphors, metonymy, hidden relations, quantifier scope, and lexical ambiguity. For instance, "Do not worry about Sam, he is a rock" is an example of a metaphor where the context "he is solid like a rock" is understood both by the speaker and the listener due to common background knowledge. However, most language models may infer Sam to be a rock and will construct synthetic sentences accordingly.

Albeit challenging, many models can be proposed to provide a shared background domain knowledge and lexicon and grammar validation for language understanding. The accuracy achieved by various existing language models is far below the acceptable threshold to be employed successfully in various text sequence generation tasks. Language models can be evaluated subjectively by humans/linguistic experts or objectively using performance evaluation metrics to evaluate a system's performance. Two metrics commonly used to evaluate the quality of generated text for correctness, contextual & semantic meaning are Bilingual Evaluation Understudy (BLEU) [4] and Recall-Oriented Understudy for Gisting Evaluation (ROUGE) [5]. BLEU tries to capture the quality of the generated text by the machine to that of humans. The closer a synthetic text is to a human translation, the better it is. On the other hand, ROUGE is a set of metrics for evaluating text summaries to that of humans. There are different variants of ROUGE available to compare on a unigram (ROUGE-1), bigram (ROUGE-2) to n-gram (ROUGE-N), or even longest common sub-sequences (ROUGE-L) based statics, among others. These metrics are only useful as long as the generated text has many overlapping words to the reference text. On the other hand, the embedding-based metrics may provide better accuracy representation for longer sequences of text generation, as explained in Section 4.4.2.

This study aims to evaluate the performance of the proposed LSTM-based text generation algorithm trained for domain-specific applications and subject them to the classification task, in addition to evaluating the recently proposed GPT-2 [6] model. The objectives involve improving the classification accuracy of various models proposed recently on a variety of datasets by generating completely new and unique domain-specific text sequences for addressing the data imbalance issue. The main contribution of this article is as follows:

- Proposed an LSTM-based sentence-level text generation model to address high data imbalance issues in common NLP related classification tasks.
- Evaluated the performance of the proposed LSTM and GPT-2 model for document-level sequence generation on three highly imbalanced datasets from two different domains.
- Showed an improved overall classification accuracy of up to 17% for all three datasets.

The rest of the paper is organized as follows. Section 2 presents an overview of the related work specific to text generation techniques and utilization of those techniques for data imbalance issues. Section 3 presents the methodology section. Results and analysis of the obtained results is presented in Section 4. Lastly, Section 5 concludes the paper with some insight into future research directions.

2. Related Work

Text generation has been examined by a growing body of literature that generally can be categorized into two major categories based upon the computational representation and techniques used to generate the natural language text. The first category consists of the research works that treat text generation as a process of character prediction/encoder-decoder problem and use RNN/variable auto-encoder as a technique to perform this task.

The study in [6] proposes to use a GPT-2 pre-trained model for generating patent claims. Specifically, a 355M GPT-2 model is trained on a dataset comprised of 555,890 patent claims of the granted US utility patents published in 2013. The findings showed that the GPT-2 model was able to generate text that looks like a patent claim using only few training steps. [7] is another research work which focuses on text generation using pre-trained GPT-2 and BERT models. A corpus composed of large number of Chinese documents is used to train the GPT-2 model to automatically generate fixed-length text. The model is trained for 5 epochs and it used default parameters besides the learning rate and batch size which were set to 1.5×10^{-4} and 1.5×10^{8}, respectively. Next, BERT model is used to generate predictions for the class surfaces (label mask) based on the context.

Text generation using LSTM networks has shown a lot of interest by the researchers in the recent years. For example, the authors in [8] used an LSTM algorithm trained on a real-world dataset composed of 917 days newspaper text in Bangle language to generate new sentences. Santhanam in [9] also tackles the issue of generating new text using LSTM model. Specifically, they proposed a context based text generator that employs LSTM models that beside input words used to learn input-output function it uses the context vectors that allow to capture the semantic meaning of the sentence. The researchers in [10] also explored text generation using deep neural networks. More concretely, they present a case study where three different models, LSTM, GRU and Bidirectional RNN are applied to generate new conversations between characters as well as new scenarios based on the historical scripts and conversations. They tested various network architectures with respect to number of layers and neurons including architecture comprised of uni-layer with 1024 neurons, bi-layers with 512 and 256 neurons, and architecture with quad-layers containing 512, 256, 128 and 64 neurons respectively. A slightly different text generation approach using deep neural networks is presented in [11]. The study proposes a text generation model using conditional text generative adversarial network to generate diverse text of variable length as well as customized emotion labels. The model is tested on four different datasets including Yelp, Amazon, Film, Obama Speech and the reported results showed that it outperforms all three baseline methods used for comparison (Markov Chain, RNN and Seq2Seq). A similar text generation approach using adversarial networks is used in [12] where a special focus is placed on the generative process over a longer horizon in order the model be able to capture the semantic meaning in long text generation.

The second category comprises research works that consider text generation to handle imbalanced datasets using traditional machine learning algorithms and deep neural networks. For example, the authors in [13] proposed a novel framework called category sentence-generative adversarial networks (CS-GAN) to generate new sentences and expand the original dataset. The framework ensembles recurrent neural networks, reinforcement learning, and generative adversarial networks. Sentiment analysis on various datasets including Amazon review, Yelp review, Stanford sentiment tree bank, Emotion dataset and news dataset covering news from NYTimes, Reuters and USAToday, were performed to validate the framework. Negative Log-likelihood is used to measure the accuracy of generated text from the model over the data of Amazon-5000 dataset that consisted of 5000 training sentences shorter than 120 characters. Experimental results showed that both generative adversarial networks and reinforcement learning had a positive influence on the performance of the framework in sentence generation.

The study conducted in [14] used two text generation techniques to generate synthetic minority class instances in order to improve the performance of imbalanced text classification. These techniques include the statistical based method called Markov Chain and

LSTM network. The proposed approach is tested on real-life dataset comprised of 2928 Thai-language advertisements collected from Facebook. The dataset contained two highly imbalanced classes of sentiment (positive and negative) with the negative class having the majority of samples. Prior to running the experiments, the classes in the training dataset were balanced using the text generation techniques. Then, a classifier model composed of an embedding layer and a LSTM architecture with 256 units and a Softmax activation function was applied to make the prediction. The experimental results showed that the best classification performance is achieved using Markov Chain text generation technique compared to LSTM and the baseline oversampling techniques.

Our study is different from the aforementioned approaches in two aspects. First, to the best of our knowledge, it is the first study which tries to handle data imbalance issue using deep neural text generation models. More concretely, two recently proposed GPT-2 and LSTM-based text generation models are used to generate new text to balance three-highly imbalanced textual datasets from the education and social media domains. Second, an in-depth performance analysis using two classification settings that involve the original datasets and the ones balanced using new generated text is conducted.

3. Methodology

The input to the proposed system is a text dataset. The first stage is to convert an imbalanced dataset into a balanced dataset where balanced dataset is a dataset with equal number of instances for each class label. At this stage, the input dataset is divided into multiple corpora, according to the distinct class labels of the input dataset. Each corpus stores instances from one distinct class label in imbalanced dataset. The text generator generates instances for each of the input corpus. The output at this stage is a balanced dataset which contains an equal number of instances for all the class labels. Figure 1 shows the abstract model of proposed system. Rest of this section explains each of the building blocks in the abstract model.

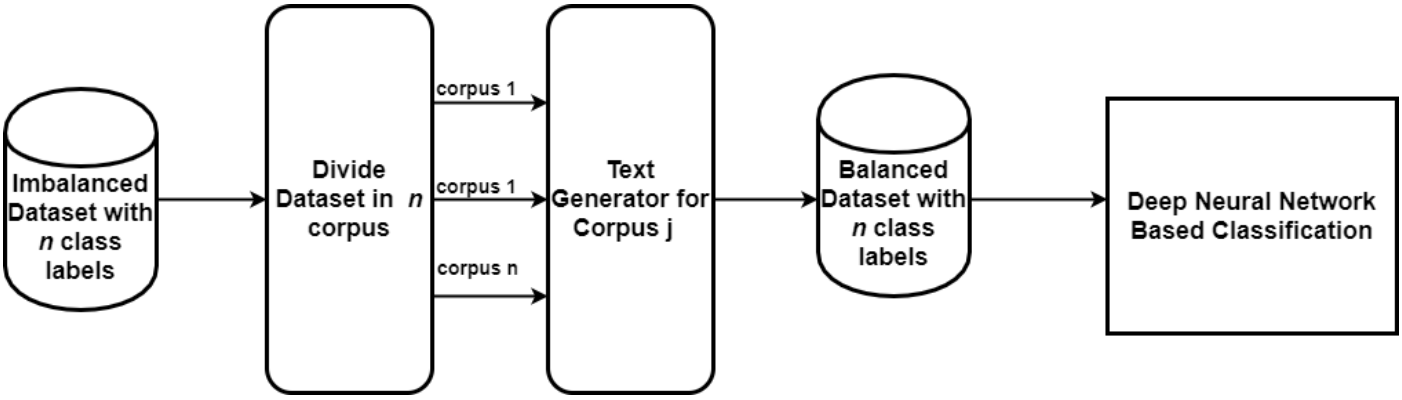

Figure 1. Abstract Model of the Proposed System.

3.1. Text Generation

In this work, we have experimented on three datasets, the details about the datasets is given in Table 1.

Table 1. Datasets used for text generation and classification.

Dataset-1	Students' Reviews [15]	A sentence-level students' feedback dataset about faculty.
Dataset-2	Tweet Emotion Dataset [16]	A tweets' dataset labeled in six emotions.
Dataset-3	MOOCs Lecture Transcripts [17]	Long text of transcripts of video lectures from Coursera.

The first two datasets contain sentence level instances, that is each instance is a short sentence, whereas the MOOC dataset contain long paragraphs which represent script of the video lecture. As shown in Sections 4.4 and 4.5, our experiments suggest LSTM based text generation works well for generating small sentences whereas GPT-2 suits better at generating long textual paragraphs, therefore in our work, we apply LSTM based text generator on first two datasets and GPT-2 model for the MOOC dataset. Both these text generation techniques are explained below.

3.1.1. LSTM Based Text Generation

The recurrent neural networks started showing a creative side of AI. In the year 2016, multiple research contributions began to appear in mainstream which demonstrated how these networks can be applied to generate sequence data [18–20]. Text is a sequential data which means recurrent networks have the ability to generate text if trained properly on a language model.

Recurrent networks are not recent in machine learning community-Long Short-Term Memory (LSTM) Network [21] was developed in 1997. The early applications for LSTM were text generation at character level though later it was also used for supervised text classification and many related applications.

An LSTM, if trained properly over a sequence, can generate next token or multiple tokens in the sequence given previous tokens as input. For example, a network can be trained to generate character *'k'* given *'cat loves mil'*. Following are few basic terminologies which we will use in our text generation problem,

token: are words or character in a text generation process.

t_{token}: given an input sequence, the t_{token} is the next character or word that network is trained to predict.

sample: generated sequence from LSTM.

conditioning data: initial string or sequence that is fed to network for predicting next token in the sequence

language model captures the latent space of language that is the statistical structure of the text. It models the probabilities of next character or word given an input sequence of characters or words.

Once a LSTM trained on a *language model*, it can generate *samples* of text given *conditioning data*. The generated sample may or may not be part of original language model, that is LSTM has a capability to generate a sequence which are new and creative in nature.

The process of text generation using LSTM is shown in Figure 2. In this example, the conditioning text is 'Cat loves m', the model could be trained on training data which is relevant to the domain of conditioning data. In this case, it could easily be Wikipedia text. The language model is LSTM model trained on the training data to predict probability of next character using Softmax activation function at the output layer which is defined as,

$$Softmax(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \quad (1)$$

where x_i is the LSTM score for character i to be the next character in conditioning text. Each of the x_i is not a probability score, therefore LSTM uses Softmax to convert LSTM scores to probabilities score.

The actual magic of text generation is hidden in sampling strategy-if the next character is produced based on highest probability score from Softmax output, it would introduce an element of predictability and the new text generated would always look similar to the original text. In order to introduce an element of newness and creativity in the generated text, it is important to bring in some randomness in the generated text. The sampling strategy introduces such randomness using temperature value.

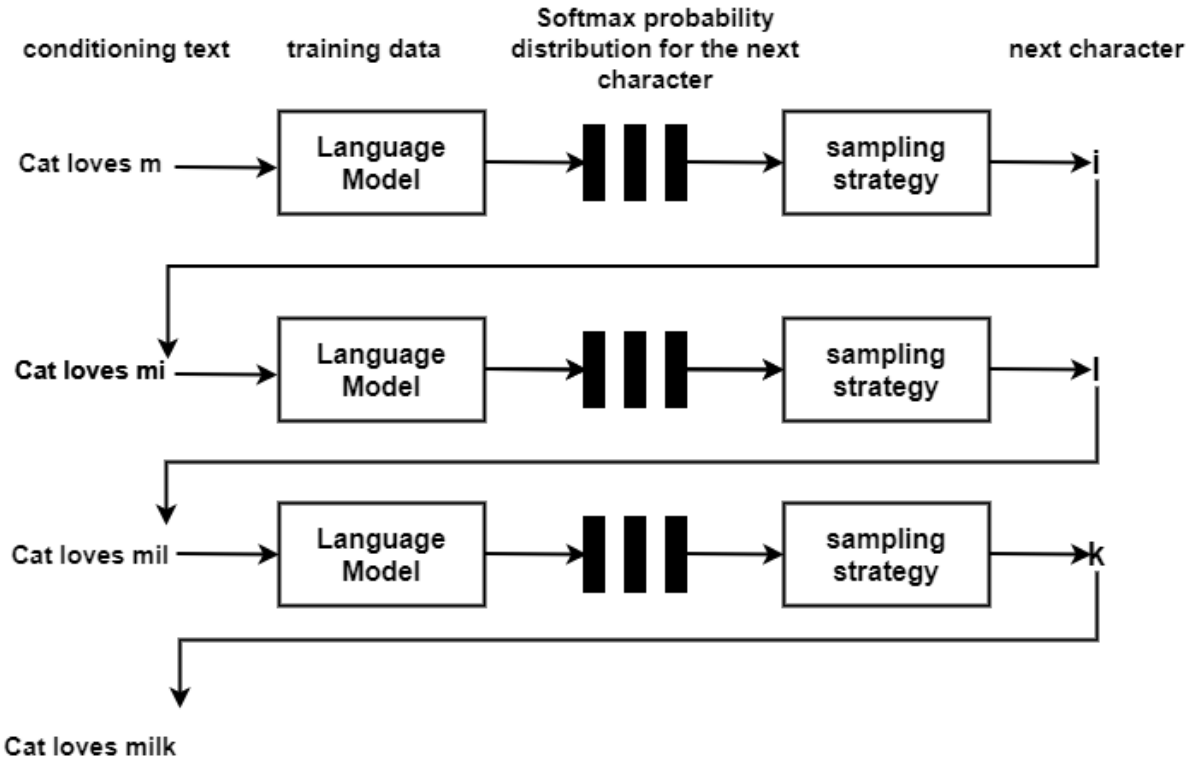

Figure 2. LSTM Process of Character Level Text Generation.

Suppose, $P_{original}$ is original probability distribution at Softmax, the α term is defined as,

$$\alpha = \log \frac{(P_{original})}{temperature} \tag{2}$$

Once α computed, the $P_{revised}$ is defined as.

$$P_{revised} = \frac{e^{\alpha}}{n} \tag{3}$$

where n are number of elements in original distribution and temperature value is an arbitrary value ranges from any non zero value up to 1. Higher temperature value results in higher entropy, which results in more randomness thus surprising results.

Language Model Building for Input Corpus

As explained in previous section, a corpus is generated for each of the input class labels. Figure 3 shows the process of text generation for each of the input corpus, the target is to generate enough number of instances so that an imbalanced dataset takes a shape of balanced dataset, that is an equal number of instances for each class label.

The input to the model is a corpus with n number of instances, the target is to generate k more instances by building a language model from these n instances so that $n + k$ is equal to the m instances. Here m represents the number of instances in a class label with highest number of instances in the input dataset.

Figure 3. Text Generation for Input Corpus.

Pre-processing

Following tasks are performed at the pre-processing step of the model,

1. Check the language of each of the instance and remove all instances with non-English language.
2. Remove special characters like hashtag, comma, semicolon, etc.
3. Remove stop words like 'is', 'am', 'are', 'the', etc.
4. Convert all the input characters to lowercase
5. Lemmatize all the words to retain only root words

All these steps are necessary to build a language model.

Building Language Model

Once the pre-processing is performed, we train an LSTM-based model on the resultant corpus words to build a language model. Figure 4 shows the model for generating the text. (Our model is based on https://minimaxir.com/2018/05/text-neural-networks/). The model itself is an LSTM based sequential layered language model that accepts text as input and returns generated text as output. As we can see from Figure 4, the model has an input layer accepting a text of 40 characters, an embedding layer, two LSTM layers, a single attention layer to give high weights to correct structured words, and at the end, an output layer for generating the text of 465 characters.

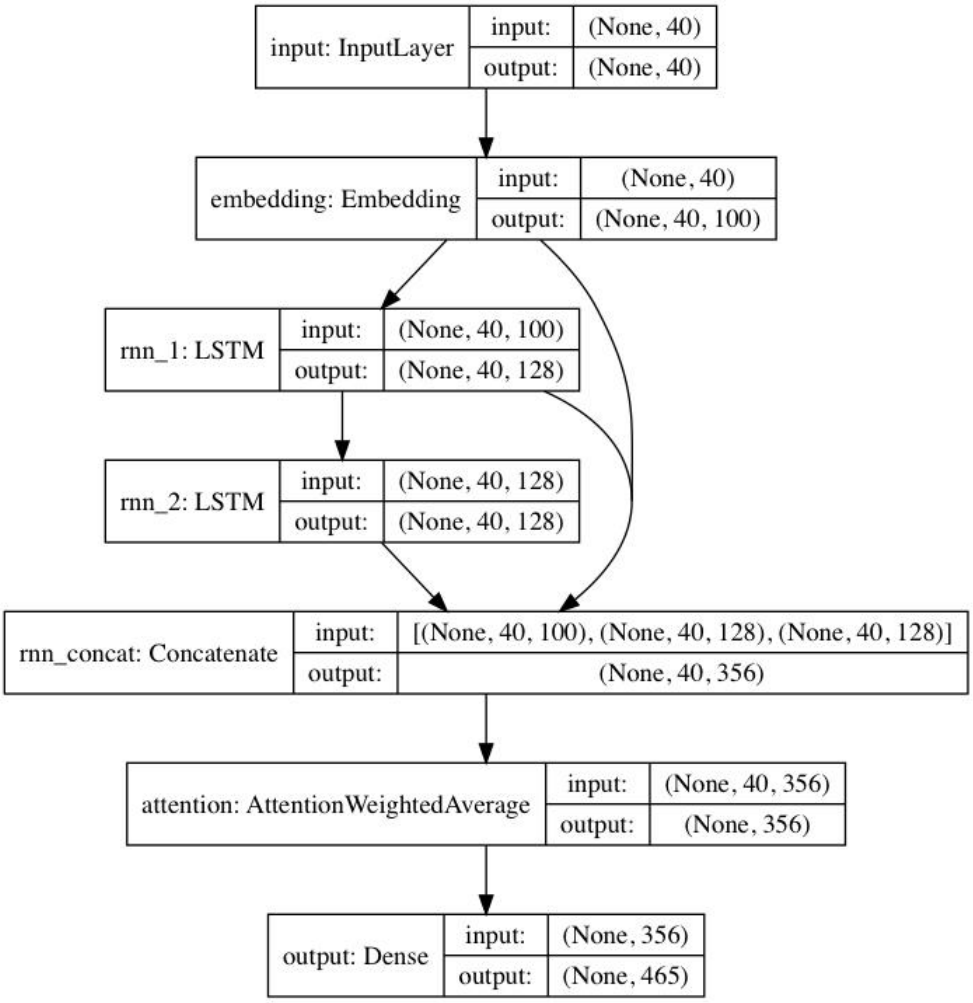

Figure 4. LSTM Model for Text Generation.

Generating Text

Tables 2 and 3 show a sample of reviews from original as well as generated students' feedback dataset and twitter emotion dataset. Although, due to short sentence in the original text, the output generated text quite often does not make much sense for which we performed detailed quality evaluation in Section 4.4. It can be observed that with right training model, machine can generate text which is very close to real text, thus imbalance of the dataset can be eliminated by generating text for those class labels where number of instances are low.

Table 2. Original + Generated Sample of Student's Feedback Dataset [15].

	Original		Generated	
#	**Text**	**Class Label**	**Text**	**Class Label**
1	he is very well experienced	Experience	well-experienced	Experience
2	give result on time	Assessment	sir still hasn't given result of 1st	Assessment
3	sir is very kind	Behavior	always pay respect to students	Behavior
4	sir is very slow in carry out course	Knowledge	he is very good teacher for case study especially	Knowledge
5	he improves teaching style	Teaching Skills	he is very fast!	Teaching Skills

Table 3. Original + Generated Sample of Twitter Emotion Dataset [16].

	Original		Generated	
#	**Text**	**Class Label**	**Text**	**Class Label**
1	Thinks that @melbahughes had a great 50th birthday party :)	Surprise	My dad just got a guest cell of my be	Surprise
2	My heart and soul @Jay_Babe is leaving me and I can't even see here	Sadness	I miss the fact that I was so much to be the dog..	Sadness
3	@CarolineHirons Buddy the elf, what's your favourite colour?	Disgust	I hate the kids that the crup that they tweet me	Disgust
4	God im so mad its burning my fuckin stomach!	Anger	Sometimes I feel if w/ I want to sleep on and it was a shit	Anger
5	...and guess what song just popped in my head.	Fear	A weak is on the floor in my hearts of the world.	Fear

3.1.2. GPT-2 Text Generation

GPT-2 model proposed by Radford et al. [22] is a revolution in processing text. The model is trained on 40GB of Internet text for generating new text, the creative side of AI. It is a Transformer [23] based architecture with 1.5 billion parameters exploiting text from eight million webpages. GPT-2 has demonstrated an exceptional human-like quality in generating conditional synthetic text of long sequences.

In this section, we discuss the steps followed to retrain GPT-2 model using our own custom MOOCs lecture dataset. As discussed earlier, GPT-2 performs better than basic LSTM text generators for text generation on long text sequences. The dataset [17] is the collection of MOOCs video lecture transcripts annotated into the eight general-level categories (i.e., Art and Humanities, Physical Sciences and Engineering, Computer Science, Data Science, Business, Information Technology, Health, Social Science). Originally, the data is highly imbalanced and needs to be balanced. For this task, we have used the GPT-2 (117M) model. The steps involved in generation and balancing the transcript dataset are shown in Figure 5 and explained in the subsequent sections.

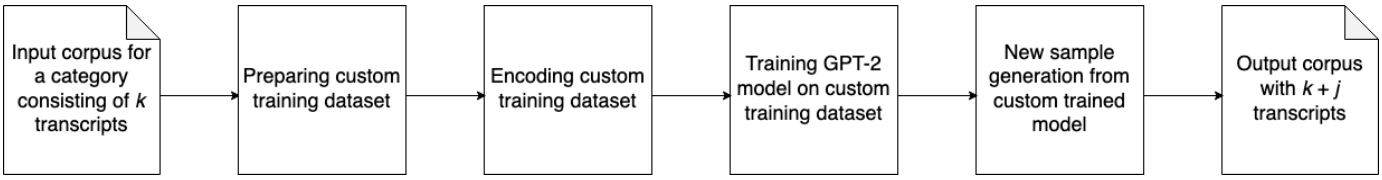

Figure 5. Text Generation for MOOCs Lecture Transcriptss using GPT-2 Model.

Preparing Custom Text Dataset

First of all, a text corpus is created for each of the class label belonging to general-level categories. This contains transcripts related to the specific class. For preparing the training data for each of the corpus, each single transcript is delimited with **"<|endoftext|>"**. This delimiter will allows the model to understand the formatting of the training data.

Encoding the Training Data

The next step is to encode the training corpus which we created in previous step. This encoding will be performed for all of the eight text corpus files belonging to the class labels. The primary purpose of encoding the text data is to convert it into numbers so that the model can understand it. Moreover, for GPT-2, the widely used encoding technique is byte-pair encoding [24] technique, a compression technique used for text data.

Training the GPT-2 Model

Here, we start training the GPT-2 model using the encoded corpus dataset created in previous step. GPT-2 requires configuration of three parameters to start learning and training on the custom dataset. The parameters are *batch_size*, *learning_rate* and *sample_every*. We set the parameters as $batch\_size = 2$, $learning\_rate = 0.0001$ and $sample\_every = 100$. Furthermore, during the training process, we used $loss\_value < 0.01$ as the stopping criteria to stop the model for further training.

Generating New Samples

Once, the training is finished for all of the text corpus belonging to eight different class labels, we used the custom trained models to generate new transcript samples. Suppose, there are k transcripts inside each class label. Therefore, we need to generate additional j transcripts such that $k + j$ is equal to the m transcripts. Here m again represents the number of transcripts in a class label with maximum number of transcripts in the original transcript dataset. The important parameters during the generation of samples are *temperature* and *top_k*. For generating samples in this work, we used $temperature = 0.8$ and $top\_k = j$. Once, the process is completed for all of the class labels, at the end we had a balanced transcript dataset consisting of previous eight general-level categories. Table 4 shows a sample of course transcript from original as well as generated MOOCs dataset. It can be observed that the GPT-2 model is quite excellent while generating long document-level texts which shows the same context as of original text and also the text is different in words as well. Again, the detailed quality evaluation of the generated text is discussed in Section 4.4.

Table 4. Original + Generated Sample of MOOCs Dataset [17].

	Original		Generated	
#	**Text**	**Class Label**	**Text**	**Class Label**
1	In the last video, you learned how to use a convolutional implementation of sliding windows. That's more computationally efficient, but it still has a problem of not quite outputting the most accurate bounding boxes. In this video, let's see how you can get your bounding box predictions to be more accurate. With sliding windows, you take this three sets of locations and run the crossfire through it.	Data Science	In the previous video, we talked. about how you can gain an intuitive understanding, of vectorization, how you can take these convolutional. functions and apply them to your models, and try to gain an intuitive understanding, of how these supervised learning algorithms work, understanding perhaps eveniving. knowledge from intuitive cues in the neural network. Let's see what that might mean,. let's go into the next video	Data Science

3.2. Deep Neural Network Based Classification

As shown in the abstract model of Figure 1, text generator based on either LSTM or GPT-2, transforms an imbalanced dataset into a balanced dataset. The next step is to assess the impact of generated text on classification performance. The major contribution of this paper is to highlight the importance of text generation and its impact on the performance of text classification, therefore for this work, we have not changed the classifiers used in the original papers.

Figure 6 shows LSTM model summary for aspect and sentiment classification applied on Dataset-1 in [15]. We used this model after text generation-for each of the aspect and orientation labels, before applying the model shown in Figure 6, we concatenate both original and generated text. The model performance both on original text and original + generated text is discussed in results section.

First Stage Model for Aspect Classification

Layer(Type)	Output Shape	Param#
Input_6 (Input Layer)	(None, 12477)	0
Embedding_6 (Embedding)	(None,2000,100)	129800
LSTM_6(LSTM)	(None,64)	42240
dense_11(Dense)	(None,32)	0
Dropout_6 (Dropout)	(None,32)	0.2 Low
dense_12(Dense)	(None,6)	198

Second Stage Model for Sentiment Orientation Classification

Layer(Type)	Output Shape	Param#
Embedding_6 (Embedding)	(None,2000,100)	600000
spatial_dropout1d_2 (Spatial)	(None,2000,100)	389648
LSTM_5(LSTM)	(None,196)	389648
dense_8(Dense)	(None,3)	788

Figure 6. Aspect and Sentiment Classification Model for Course and Teacher [15].

Similarly, on Dataset-2, we applied the same model as proposed in [25]. There has been multiple attempts at classifying Tweet Emotion Dataset text, the best attempt is based on LSTM model with pre-trained Glove Twitter embedding as reported in [25]. The model summary is shown in Figure 7.

```
Model: "Pre-trained Glove.Twitter.27B.200d"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_4 (Embedding)      (None, 280, 200)          11207600
_________________________________________________________________
lstm_5 (LSTM)                (None, 32)                29824
_________________________________________________________________
dense_5 (Dense)              (None, 6)                 198
=================================================================
Total params: 11,237,622
Trainable params: 11,237,622
Non-trainable params: 0
_________________________________________________________________
Train on 40045 samples, validate on 4450 samples
```

Figure 7. LSTM with Glove Twitter Embedding Model Used in [25].

Finally, we use the same model of classification as proposed in [26] for Dataset-3.

4. Results

In this section we present our experimental results on Dataset-1, Dataset-2 and Dataset-3 with and without text generation.

4.1. Dataset-1: Students' Reviews

We acquired a hand-labelled in-lecture dataset from an existing research work published in 2019 [15]. The dataset is a collection of students' feedback of Sukkur IBA University for the last five years. Irum et al., in [15] have used this dataset for the purpose of aspect based sentiment analysis. Therefore, the student's feedback are hand-labelled into six different aspects (General, Teaching Skills, Experience, Knowledge, Behavior and Assessment) and three different sentiment orientations (Positive, Negative and Neutral). The authors used LSTM based neural network model to perform aspect as well as sentiment orientation classification. The authors reported overall accuracy of 91% and 93% for the aspect and sentiment orientation classification, respectively. However, a major drawback of this dataset was the imbalanced distribution of the classes for aspect as well as sentiment orientation. This resulted in low precision and F1-score of 89% and 85%, respectively. Table 5 shows the different class-wise distributions of the dataset.

Table 5. Imbalanced Distribution [15].

Aspect	**Count**
General	1867
Teaching Skills	1656
Knowledge	1186
Behavior	844
Experience	297
Assessment	139
Sentiment Orientation	**Count**
Positive	4250
Negative	1489

From the Table 5, we can see that the "General" aspect class and "Positive" sentiment orientation class has got the highest students' feedback. So, by keeping these classes as a reference class we applied our proposed text-generation system explained in Section 3.1 to rest of the aspect as well as sentiment orientation classes except the general and the positive class. The model generated new students' feedback for each of the classes in order to make all the classes balanced. Table 6 shows the class-wise distribution after applying text-generation model.

Table 6. Balanced Distribution.

Aspect	**Count**
General	1867
Teaching Skills	1867
Knowledge	1867
Behavior	1867
Experience	1867
Assessment	1867
Sentiment Orientation	**Count**
Positive	4250
Negative	4250

Now, once the balanced dataset is ready, we perform experiments using same model architecture from the authors of [15] with same technical settings and train, val, test set distributions. (Train Set: 70%, Val Set: 10% and Test Set: 20%). The results obtained from the balanced dataset were quite excellent and surprising specially for the overall precision and F1-score. We achieved overall accuracy of 93%, which is an increase of 2% from the original accuracy. Moreover, we obtained an overall precision and F1-score of 93%, which is also an increase of 8% and 4% as compared to original F1-score and precision results, respectively. Table 7 shows the class-wise precision, recall and F1-scores of all six aspect classes before and after text generation. The dataset also contained duplicates which we removed for another set of experiments. Tables 8–10 show results of text generation after removing duplicates.

Table 7. Aspect Classification Results.

	Imbalanced Dataset [15]			**Balanced Dataset**		
Aspect	**Precision**	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**
General	0.95	0.90	0.92	0.92	0.90	0.91
Teaching Skills	0.89	0.83	0.85	0.87	0.86	0.87
Knowledge	0.87	0.88	0.87	0.94	0.92	0.93
Behavior	0.82	0.85	0.83	0.94	0.93	0.94
Experience	0.93	0.90	0.91	0.97	0.97	0.97
Assessment	0.90	0.62	0.73	0.98	0.99	0.99
Overall	**0.89**	**0.83**	**0.85**	**0.93**	**0.93**	**0.93**

We can see the performance improvement after balancing the datasets. For example, Table 7 shows the results for Student's Reviews dataset [15] for an imbalanced and balanced one. The most imbalanced classes were the **Experience** and **Assessment** classes. We can see the underlying improved precision, recall, and F1-score values before and after balancing the dataset. Also, the overall score values are improved, as shown at the end of Table 7.

Table 8. Imbalanced Distribution [15] (Without Duplicates).

Aspect	**Count**
Teaching Skills	1296
Knowledge	915
General	806
Behavior	669
Experience	239
Assessment	88
Sentiment Orientation	**Count**
Positive	4250
Negative	1489

Table 9. Balanced Distribution (Without Duplicates).

Aspect	**Count**
Teaching Skills	1296
Knowledge	1296
General	1296
Behavior	1296
Experience	1296
Assessment	1296
Sentiment Orientation	**Count**
Positive	4250
Negative	1489

Table 10. Aspect Classification Results (Without Duplicates).

	Imbalanced Dataset [15]			**Balanced Dataset**		
Aspect	**Precision**	**Recall**	**F1-Score**	**Precision**	**Recall**	**F1-Score**
General	0.85	0.86	0.85	0.86	0.91	0.89
Teaching Skills	0.82	0.89	0.85	0.83	0.81	0.82
Knowledge	0.92	0.88	0.90	0.89	0.88	0.89
Behavior	0.89	0.80	0.84	0.89	0.88	0.88
Experience	0.80	0.81	0.80	0.93	0.91	0.92
Assessment	0.83	0.62	0.71	0.97	0.95	0.96
Overall	**0.86**	**0.86**	**0.86**	**0.89**	**0.89**	**0.89**

4.2. Dataset-2: Tweet Emotion Dataset

Our next set of experiments were performed on Tweet Emotion Dataset, the tagging process of which is explained in [16]. This dataset is a collection of labeled tweets in six emotion categories including joy, surprise, sadness, anger, fear and disgust. Table 11 shows class labels and number of tweets under each class label. It clearly indicates that the dataset is highly imbalanced, it has a high number of instances for the label 'Joy', whereas a very low number of instances for label 'Disgust'. In order to convert this imbalanced dataset to a balanced dataset, we generated enough number of instances for each class label to equal them to the number tweets for class label 'Joy' that is 8240 instances which is the class label with highest number of tweets. The corresponding balanced dataset number of instances is presented in Table 12.

Table 11. Imbalanced Distribution [16].

Class Label	Count
Joy	8240
Surprise	3849
Sadness	3830
Fear	2816
Anger	1555
Disgust	761

Table 12. Balanced Distribution.

Class Label	Count
Joy	8240
Surprise	8240
Sadness	8240
Fear	8240
Anger	8240
Disgust	8240

The classification model that we have used on Dataset-2 is summarized in Section 3, Figure 7. Table 13 shows accuracy achieved by the model of Figure 7 on both balanced and imbalanced dataset.

Table 13. Performance of LSTM + Glove Twitter Embedding on Imbalanced and Balanced Tweet Emotion Dataset.

Number of class Labels	Accuracy on Imbalanced Dataset	Accuracy on Balanced Dataset
6 (joy, surprise, sadness, anger, fear, disgust)	59%	73%
2 (joy, surprise)	81.9%	84%
4 (sadness, anger, fear, disgust)	69%	86.5%

The reason of reporting accuracy separately on six, two and four classes is that in previous work [25] the emotions were divided according to their polarity, that is joy and surprise considered positive polarities whereas sadness, anger, fear and disgust considered as negative polarity emotions.

In all three cases, i.e., for six class labels, two and four, the accuracy improves with introducing balance to the dataset by generating text. For six class labels, the accuracy improves by 14% which is a significant improvement, where as it improves by 2.1% and 17.5% for two classes and fours classes respectively. The experiments on Tweet Emotion Dataset show that for different class labels, an accuracy improvement of ranging from 2.1% to 17.5% can be accomplished by balancing the dataset using text generation.

4.3. Dataset-3: MOOCs Lectures Dataset

Next, we examined the impact of new generated MOOCs lecture transcripts to balance this dataset and how the balanced data can influence the classification performance. To accomplish this task, we run experiments with two different classification settings in term of the data used as an input to the classifiers. For the first classification setting, the original imbalanced lecture transcripts were used as an input to four different conventional machine learning classifiers. Classifiers include parametric techniques such as Naive Bayes (NB) and Support Vector Machine (SVM), non-parametric technique like Decision Tree (DT), and ensemble learning technique—AdaBoost. The second classification setting consists of the balanced data using synthetic lectures transcripts which are used to be feed to the

classification algorithms. The obtained results in terms of F1 measure are illustrated in Figure 8.

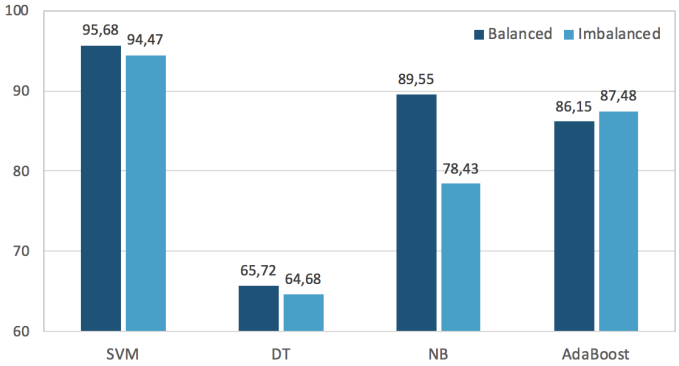

Figure 8. Classification performance of conventional ML techniques on original and balanced dataset with new generated text.

As can bee seen from the graph in Figure 8, a considerable improvement in classification performance is achieved by almost all the classifiers when balanced dataset is used as an input. Specifically, the most significant improvement is achieved by NB, where the F1 is increased from 78.43% to 89.55%. On the contrary, a slightly worse classification performance is obtained by AdaBoost algorithm. This may have happened due to the weighting strategy of combining many weak classifiers the AdaBoost algorithm applies to optimize weighted training error. There are different strategies to combine classifiers that can be fine-tuned depending on the type of application but in our case a standard strategy is used.

4.4. Evaluating Generated Text

This section explains the set of various automatic evaluation metrics for the evaluation of generated texts. Usually, the human evaluation is best for this kind of evaluation but due to large quantity of generated texts, we had to approach these automatic metrics. The metrics are divided into two sets. (1) Word-overlap metrics and (2) Embedding-based metrics [27]. In all those metrics, we compute the quality and similarity of generated texts to all of their reference/original texts at corpus level. The texts related to one category/class is considered as one single corpus. The detailed explanation of all these metrics is given in subsequent sections.

4.4.1. Word-Overlap Metrics

- **BLEU**
  The BLEU metric [4] performs n-grams comparison between the generated texts and original/reference texts. In our work, we have applied this metric at corpus level for the evaluation of generated texts. The BLEU score calculation is mathematically defined as below:
  $$BLEU - N = BP * exp(\sum_{n}^{N} w_n log(p_n)) \quad (4)$$
  where N is the maximum length for n-grams (in this paper, we have used BLEU-3) because implicitly it contains the BLEU-1 and BLUE-2, w is a uniform weighting and BP is the brevity penalty. In the next sections, BLEU score implicitly refers to BLEU-3.
- **METEOR**
  The METEOR metric [28] is used to correlate human evaluation more better for the generated texts. This score first create unigram alignment by assigning 0 or 1 to the unigram in the generated texts to the unigram in reference/original texts. This alignment not only considers the exact matches but also stemming, synonyms and

paraphrase matching. The precision and recall for unigrams are calculated based on this alignment. The METEOR score calculation is mathematically defined as below:

$$METEOR = F_{mean}(1 - p) \tag{5}$$

where F_{mean} is the harmonic mean of recall and precision and p is a penalty.

- **ROUGE-L**
  The ROUGE-L metric [5] is a F-measure based calculation defined on the Longest Common Subsequence (LCS) between the generated and original texts.

4.4.2. Embedding-Based Metrics

These metrics computes the cosine similarity for the embeddings of the generated and reference/original texts instead of just relying on previous word-overlap methods. Cosine Similarity is widely used technique for measuring documents similarity irrespective of their size. Mathematically, this represents the cosine angle between two embedding vectors in n-dimensional space [29]. For the embeddings, the famous Glove-6B [30] pre-trained embeddings are used.

- **Skip-Thought Cosine Similarity**
  The Skip-Thought model [31] is a combination of multiple recurrent networks which performs encoding of generated and original/reference texts into embeddings and then measure the cosine similarity of those embeddings. For this study, we have used the pre-trained Skip-Thought network provided by [21].
- **Embedding-Average**
  This metric calculates the sentence-level embedding scores by calculating average embeddings of all the words composing the text. This metric perform this calculation for all generated and reference/original texts and at the end calculates the cosine similarity. The formula for calculating average embeddings is defined as:

$$\bar{e}_C = \frac{\sum w \epsilon C^{e_w}}{|\sum x \epsilon C^{e_x}|} \tag{6}$$

where e_w and e_x represent the embeddings for words w and x in the sentence C.

Tables 14–16 shows the calculations of all of these metrics for the twitter emotions [16], student's feedback [15] and MOOCs lecture [17,26] datasets, respectively. We applied these metrics on the generated texts of above three datasets. We performed text generation for all the classes in all three datasets except the classes which contained the highest number of instances in the original dataset, in order to set those classes as the benchmark class to create the balanced dataset.

Table 14. Evaluation Metrics Scores (Twitter Emotion Dataset [16]).

Metrics	Anger	Disgust	Fear	Sadness	Surprise
BLEU	0.004	0.000	0.008	0.005	0.003
METEOR	0.039	0.037	0.039	0.039	0.032
ROUGE_L	0.066	0.066	0.061	0.066	0.048
Skip-Thought	0.312	0.284	0.304	0.295	0.294
Embedding-Average	0.691	0.716	0.649	0.726	0.548

Table 15. Evaluation Metrics Scores (Hand-Labelled In-Lecture Dataset [15]).

Metrics	Assessment	Behavior	Experience	Knowledge	Teaching Skills
BLEU	0.011	0.000	0.015	0.013	0.000
METEOR	0.046	0.030	0.059	0.073	0.058
ROUGE_L	0.091	0.052	0.091	0.104	0.081
Skip-Thought	0.359	0.320	0.318	0.355	0.366
Embedding-Average	0.696	0.465	0.539	0.633	0.646

Table 16. Evaluation Metrics Scores (MOOCs Lecture Transcripts Dataset [26]).

Metrics	Arts and Humanities	Business	Computer Science	Data Science	Health	Information Technology	Social Sciences
BLEU	0.029	0.046	0.034	0.035	0.028	0.041	0.022
METEOR	0.128	0.151	0.114	0.128	0.107	0.147	0.105
ROUGE_L	0.124	0.125	0.124	0.126	0.116	0.126	0.113
Skip-Thought	0.435	0.481	0.461	0.473	0.468	0.472	0.439
Embedding-Average	0.964	0.982	0.983	0.977	0.961	0.975	0.964

4.5. Discussion Related to Evaluation Metrics

This section discusses the obtained scores for all the evaluation metrics for generated texts mentioned in the previous section. The Tables 14 and 15 represents the metrics related to LSTM model based generated text. Whereas, Table 16 represents the metrics related to GPT-2 based generated text. If we look into all the three tables, the first three metrics (BLEU, METEOR and ROUGE_L) have very low score values for the generated texts with reference to original texts. However, the next two metrics (Skip-Thought, Embedding-Average) represents high score values for the generated texts with reference to the original texts.

The possible reason for low score values for the first three metric is due to less number of overlapping words in the generated texts with reference to original texts. But, the generated texts of the specific class label is relevant. Tables 17 and 18 shows the number of overlapping words (i.e., bold words) in generated as well as original texts for different class labels. The tables shows that the number of overlapping-words are less in number but still the generated texts represents sense of relevant class labels.

Table 17. Original + Generated Texts Sample from Twitter Emotion Dataset [16].

#	Original Text	Generated Text	Class Label
1	I just absolutely **lost my mind**.. :(	I want to complain to the rest of my since people and walk by the times again. What is your face that so I don't understand or **lost my mind** down	anger
2	The fact that @KimKardashian wedding makes headlines and provides that pathetic excuse of a celebrity **with more money** makes me sick	I hate the worst time when you live picture the stomarting cover her poppet time **with the more money** or an thing like a go me	digust
3	Don't believe the lies look me In the eyes, please don't be scared of me	I don't wanna stay a trust I feel a combinations/hardest for me	fear
4	I miss hanging out with AM, Lee and Jay. Those were **good times**.	i hate **good times** to be able to throw a depression :(	sadness
5	You would think by now they would of controlled the smoke problem in the **bathrooms**. Butttt, they haven't.	I just found out that we are taking a card for **bathrooms** and a surprise in a surprise birthday	surprise

Table 18. Original + Generated Texts Sample from Hand-Labelled In-Lecture Dataset [15].

#	Original Text	Generated Text	Class Label
1	**evaluation** is unjustified	**evaluated** us more than our level	assessment
2	Always showing **respect towards the students**	showed **respect to students**	behavior
3	he just has not experience of **how to handle class**	sir is not **able to handle class** because of which effective learning does not take place	experience
4	have **good knowledge** about subject	He has **great knowledge** about the course	knowledge
5	**cooperative** teacher	he is very **cooperative**	teaching skills

However, when we calculated second set of metrics (i.e., embedding-based metrics) we observed significant improvement in the calculated scores. These scores represent the semantic similarity of the generated texts with reference to original texts. Hence, the scores give evidence that the generated text is relevant to specific class labels.

Another, point that needs to be discussed here is we know that the Tables 14 and 15 show scores for LSTM based text generation model and the Table 16 shows scores for GPT-2

based text generation model. The metric scores calculated for GPT-2 text generation model represents high values as compared to the other text generation model. The possible reason for improved performance in GPT-2 is due to the transformer language model which relies on the attention mechanism, trained on millions of sentences, words and characters which learns each and every possible context resulting in generating coherent and human-like text [32]. This comparison shows the reliability of GPT-2 model for the text generation task.

5. Conclusions and Future Directions

In this paper, we evaluated recent GPT-2 and LSTM-based language modelling architectures to balance the highly imbalanced datasets by generating new text. The generated text was completely new and retained the proper grammatical structure, context and semantic information in relation with the original text. The aim was to improve text classification results by balancing the data with generated text for the underrepresented classes. To achieve the objective of this paper, we selected three datasets and for those datasets we generated text and then performed text classification using the existing networks which were already applied on the original datasets. The obtained results supported our objective by giving improved results for classes having low precision and recall values in the original experiments. We performed experiments on two different kinds of datasets. (1) Sentence-level and (2) Document-level. The LSTM-based text generation algorithm performed very well for sentence-level datasets. But for the document-level it performed very poor, possibly because LSTM-based networks can learn the contextual dependencies upto certain limit and failed for longer texts. However, the GPT-2 based text generation algorithm solved this bottleneck and gives excellent results specially for document-level dataset. Furthermore, two of the datasets were from the academic domain and one was from the social-media domain. The text generation models performed exceptionally well in order to incorporate domain-specific information while generating text for three different datasets. Overall, for all three datasets we observed the improvement of upto 17% in overall performance of the text classification models for the balanced dataset compared to the original results.

Currently, we evaluated the use of text generation on education and social media domains but this technique can be easily applied to other domains with little modifications to the text generation algorithms parameters. Also, another future direction can be the use of contextualized word embeddings models like BERT and ELMo to understand if these also can work with the text generation models.

Author Contributions: S.S. performed all experiments related to text generation. He also contributed in writing results section related to Dataset-1 and evaluating text generation quality. GPT-2 relateed part in methodology is also contributed by S.S. S.M.D. contributed in writing methodology part related to LSTM text generation & classification and results section of Dataset-2. A.S.I. conceived overall idea of the paper, wrote introduction and abstract parts along with his contribution in related work and overall reviewing the text quality. Z.K. contributed in writing related work, performed experiments on Dataset-3 and write relevant parts of results section. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not Applicable.

Informed Consent Statement: Not Applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

SMOTE	Synthetic Minority Oversampling Technique
AdaSyn	Adaptive Synthetic
GANs	Generative Adversial Networks
LSTM	Long Short Term Memory
GPT-2	Generative Pre-trained Transformer 2
BLEU	Bilingual Evaluation Understudy
ROUGE	Recall-Oriented Understudy for Gisting Evaluation
NLP	Natural Language Processing
RNN	Recurrent Neural Network
BERT	Bidirectional Encoder Representations from Transformers
GRU	Gated Recurrent Unit
CS-GAN	Cyclic-Synthesized Generative Adversarial Networks
MOOC	Massive Open Online Course
AI	Artificial Intelligence
Glove	Global Vectors for Word Representation
METEOR	Metric for Evaluation of Translation with Explicit ORdering
CIDEr	Consensus-based Image De-scription Evaluation
ELMo	Embeddings from Language Models

References

1. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic minority over-sampling technique. *J. Artif. Intell. Res.* **2002**, *16*, 321–357. [CrossRef]
2. He, H.; Bai, Y.; Garcia, E.A.; Li, S. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In Proceedings of the IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), Hong Kong, China, 1–8 June 2008; pp. 1322–1328.
3. Almahairi, A.; Rajeswar, S.; Sordoni, A.; Bachman, P.; Courville, A. Augmented cyclegan: Learning many-to-many mappings from unpaired data. *arXiv* **2018**, arXiv:1802.10151.
4. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.J. BLEU: A method for automatic evaluation of machine translation. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, PA, USA, 7–12 July 2002; pp. 311–318.
5. Lin, C.Y. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2004; pp. 74–81.
6. Lee, J.S.; Hsiang, J. Patent claim generation by fine-tuning OpenAI GPT-2. *World Pat. Inf.* **2020**, *62*, 101983. [CrossRef]
7. Qu, Y.; Liu, P.; Song, W.; Liu, L.; Cheng, M. A Text Generation and Prediction System: Pre-training on New Corpora Using BERT and GPT-2. In Proceedings of the IEEE 10th International Conference on Electronics Information and Emergency Communication (ICEIEC), Beijing, China, 17–19 July 2020; pp. 323–326.
8. Islam, M.S.; Sharmin Mousumi, S.S.; Abujar, S.; Hossain, S.A. Sequence-to-sequence Bangla Sentence Generation with LSTM Recurrent Neural Networks. *Procedia Comput. Sci.* **2019**, *152*, 51–58. [CrossRef]
9. Santhanam, S. Context based Text-generation using LSTM networks. *arXiv* **2020**, arXiv:cs.CL/2005.00048.
10. Mangal, S.; Joshi, P.; Modak, R. LSTM vs. GRU vs. Bidirectional RNN for script generation. *arXiv* **2019**, arXiv:cs.CL/1908.04332.
11. Chen, J.; Wu, Y.; Jia, C.; Zheng, H.; Huang, G. Customizable text generation via conditional text generative adversarial network. *Neurocomputing* **2020**, *416*, 125–135. [CrossRef]
12. Zhang, R.; Chen, C.; Gan, Z.; Wang, W.; Shen, D.; Wang, G.; Wen, Z.; Carin, L. Improving Adversarial Text Generation by Modeling the Distant Future. *arXiv* **2020**, arXiv:cs.CL/2005.01279.
13. Li, Y.; Pan, Q.; Wang, S.; Yang, T.; Cambria, E. A Generative Model for category text generation. *Inf. Sci.* **2018**, *450*, 301–315. [CrossRef]
14. Akkaradamrongrat, S.; Kachamas, P.; Sinthupinyo, S. Text Generation for Imbalanced Text Classification. In Proceedings of the 16th International Joint Conference on Computer Science and Software Engineering (JCSSE), Pataya, Thailand, 10–12 July 2019; pp. 181–186. doi:10.1109/JCSSE.2019.8864181. [CrossRef]
15. Sindhu, I.; Daudpota, S.M.; Badar, K.; Bakhtyar, M.; Baber, J.; Nurunnabi, M. Aspect-based opinion mining on student's feedback for faculty teaching performance evaluation. *IEEE Access* **2019**, *7*, 108729–108741. [CrossRef]
16. Mohammad, S.M.; Bravo-Marquez, F. WASSA-2017 Shared Task on Emotion Intensity. In Proceedings of the Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis (WASSA), Copenhagen, Denmark, 7–11 September 2017; pp. 34–39.
17. Kastrati, Z.; Kurti, A.; Imran, A.S. WET: Word embedding-topic distribution vectors for MOOC video lectures dataset. *Data Brief* **2020**, *28*, 105090. [CrossRef] [PubMed]

18. Pawade, D.; Sakhapara, A.; Jain, M.; Jain, N.; Gada, K. Story scrambler-automatic text generation using word level rnn-lstm. *Int. J. Inf. Technol. Comput. Sci. (IJITCS)* **2018**, *10*, 44–53. [CrossRef]
19. Chakraborty, S.; Banik, J.; Addhya, S.; Chatterjee, D. Study of Dependency on number of LSTM units for Character based Text Generation models. In Proceedings of the International Conference on Computer Science, Engineering and Applications (ICCSEA), Sydney, Australia, 19–20 December 2020; pp. 1–5.
20. Park, D.; Ahn, C.W. LSTM encoder-decoder with adversarial network for text generation from keyword. In *Theories and Applications, Proceedings of the International Conference on Bio-Inspired Computing, Qingdao, China, 23–25 October 2018*; Springer: Cham, Switzerland, 2018; pp. 388–396.
21. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
22. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language models are unsupervised multitask learners. *OpenAI Blog* **2019**, *1*, 9.
23. Tetko, I.V.; Karpov, P.; Van Deursen, R.; Godin, G. State-of-the-art augmented NLP transformer models for direct and single-step retrosynthesis. *Nat. Commun.* **2020**, *11*, 1–11. [CrossRef] [PubMed]
24. Bostrom, K.; Durrett, G. Byte pair encoding is suboptimal for language model pretraining. *arXiv* **2020**, arXiv:2004.03720.
25. Imran, A.S.; Daudpota, S.M.; Kastrati, Z.; Bhatra, R. Cross-Cultural Polarity and Emotion Detection Using Sentiment Analysis and Deep Learning on COVID-19 Related Tweets. *IEEE Access* **2020**, doi:10.1109/ACCESS.2020.3027350. [CrossRef]
26. Kastrati, Z.; Imran, A.S.; Kurti, A. Integrating word embeddings and document topics with deep learning in a video classification framework. *Pattern Recognit. Lett.* **2019**, *128*, 85–92. [CrossRef]
27. Sharma, S.; Asri, L.E.; Schulz, H.; Zumer, J. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. *arXiv* **2017**, arXiv:1706.09799.
28. Banerjee, S.; Lavie, A. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In Proceedings of the Acl Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, Ann Arbor, MI, USA, June 2005; pp. 65–72.
29. Huang, A. Similarity measures for text document clustering. In Proceedings of the Sixth New Zealand Computer Science Research Student Conference (NZCSRSC2008), Christchurch, New Zealand, 28–29 April 2008; Volume 4, pp. 9–56.
30. Pennington, J.; Socher, R.; Manning, C.D. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1532-1543.
31. Kiros, R.; Zhu, Y.; Salakhutdinov, R.R.; Zemel, R.; Urtasun, R.; Torralba, A.; Fidler, S. Skip-thought vectors. In *Advances in Neural Information Processing Systems*; Bradford Books: London, UK, 2015; pp. 3294–3302.
32. Fagni, T.; Falchi, F.; Gambini, M.; Martella, A.; Tesconi, M. TweepFake: About Detecting Deepfake Tweets. *arXiv* **2020**, arXiv:2008.00036.