PS-GAN: Feature augmented text generation in Telugu

Pratyusha Musunuru[*]
Blaize
`pratyusha.musunuru@blaize.com`

Varshit Battu[*]
Plivo
`varshit.battu@plivo.com`

Abstract

Since the inception of Generative Adversarial Networks (GANs), synthetic image generation has taken a giant leap because of the ability of these networks to generate high-quality images, however, the same cannot be said for text generation. A major challenge encountered in text generation using GAN's is the non-differentiability of the discrete text. Most of the previous studies for text generation using GANs focus on solving this, but none of them incorporate any additional features in the GAN. These features could be useful in the training of the models, especially in the case of low-resource languages. In this paper, we propose a novel model called the POS-Senti-GAN (PS-GAN), where we show that the use of Parts-Of-Speech tag and sentiment features aid in the generation of better sentences. We also provide 'Pravar', a first-ever dataset consisting of stories from different categories that enable text/story generation for Telugu, a low resource language. Finally, we show the performance of the proposed models on three datasets, namely, Pravar[1], Telugu Wikipedia[2] and Telugu News[3].

[*] The authors contributed equally to this work.
[1] Pravar is a Hindi word that translates to "good" or "important".
[2] https://www.kaggle.com/disisbig/telugu-wikipedia-articles
[3] https://github.com/AnushaMotamarri/Telugu-Newspaper-Article-Dataset

1 Introduction

Text generation strives towards generating text close to human written text by leveraging syntactic, semantic knowledge and artificial intelligence in the formation of sentences. This field of text generation can be applied to many natural language processing tasks such as generating responses to users' queries, translating a sentence or a document from one language into another, offering suggestions to help write a story, generating summaries, etc.

GANs were first introduced by (Goodfellow et al., 2014). They learn by solving the min-max optimization problem, i.e., the discriminator seeks to maximize the likelihood for the actual data to be identified as real and the generated data to be identified as fake, whereas the generator seeks to minimize the likelihood for its generated data to be identified as fake. When using GANs on text, we run into issues while back-propagating the discriminator loss to the generator. As mentioned in (Iqbal and Qureshi, 2020), differentiating a pixel of an image makes sense, whereas differentiating a word (discrete stochastic unit) doesn't. This problem was addressed in two ways, (Jang et al., 2016) used an efficient gradient estimator that replaces the non-differentiable sample from a categorical distribution with a differentiable sample from a novel Gumbel-Softmax distribution, and (Yu et al., 2017) used reinforcement learning.

Out of the many features present in the text, Parts-Of-Speech(POS) tags and sentiment are the most prominent ones. POS tags carry the semantic features of a sentence. They are a crucial part of every language as they serve as building blocks for its grammar, particularly in the linguistic domain. POS tags are an essential feature used in multiple fields like information retrieval, text classification, etc. We can utilize them to make assumptions about semantics since they define the characteristic structure of lexical phrases within a sentence or text. On the other hand, sentiment refers to the tone contained in a sequence of words that is utilized to better understand the emotions and opinions conveyed through the text. When it comes to domains like user reviews, feedback, recommendations, stories, etc., sentiment is used as a feature for various applications. Going forward, we show that by considering these features when generating sentences, there is a significant increase in the generated sentences' quality compared to standard text generation.

Most of the advances in deep learning happen on images followed by text, which predominantly happen in the English language followed by others. English being the most spoken language around the world and being widely available on the internet enhances the feasibility of conducting a wide variety of experiments. On the other hand, the availability of digital content for low-resource languages like Telugu, Hindi, Tamil, etc., is low. Although text generation has been pursued in other languages, it has not yet been experimented on Indian languages like Telugu. We attempt to solve this problem by providing Pravar, a dataset consisting of short stories in Telugu, which can be used as a baseline for text/story generation and further processed for solving various other problems like story classification, emotion extraction, etc.

2 Related work

Text generation using GAN's has been explored quite a bit. Initially, (Yu et al., 2017) proposed a sequence GAN to solve the problem of back-propagating gradients and used reinforcement learning to train the generator. With time, many variations have been made in generators and discriminators to solve different problems in text generation. One such variation was proposed by (Lin et al., 2017), where rather than predicting true or false for an individual data sample, the discriminator is trained to rank a collection of human-written and machine-written sentences. (Che et al., 2017) propose a Maximum-Likelihood based objective involving importance sampling and other reduction techniques to make the training more stable. (Wang and Wan, 2018) try to address the problem of generating different categories of text by adopting multiple generators and back-propagating the loss for each category from the multi-class discriminator. On similar lines, (Li et al., 2018) proposed category sentence GAN (CS-GAN), which uses recurrent neural networks and reinforcement learning to generate category-based sentences. To increase the diversity of the generated sentences, (Xu et al., 2018) introduced the Diversity-Promoting GAN (DP-GAN), which assigns low reward for repeatedly generated text and high reward for novel and fluent text, encouraging the generator to produce diverse and informative text. An entirely novel framework called LeakGAN was proposed by (Guo et al., 2018), which allows the discriminator to leak its hidden features to the generator. This information is incorporated into the generator through an additional MANAGER module which guides the WORKER module in generating the next word. Without using reinforcement learning to optimize the GAN objective, (Chen et al., 2018) consider matching the latent feature distributions of real and synthetic sentences using the feature mover's distance (FMD) and convert it to a differential quantity for the back-propagation of the loss. Adopting Gumbel-Softmax relaxation in the GAN architecture, (Nie et al., 2018) introduce RelGAN, which comprises of a relational memory-based generator and multiple embedded representations in the discriminator to optimize the quality, diversity as well as performance.

POS and sentiment features are notable features that help solve numerous use-cases in natural language processing. (Sreeram and Sinha, 2018) talk about how POS features can be used to improve the textual modeling of Code-Switched (Hindi-English) data by the language models trained on native (Hindi) language. (Gómez-Rodríguez et al., 2019) discuss how important the structure of a sentence (semantics) is, while performing specific natural language tasks. They investigate how parsing quality affects the performance of a sentiment analysis system that determines the polarity of sentences based on their parse trees. As POS tags contain information about the semantics of a sentence, this is an invaluable feature. Coming to sentiment, (Chelaru et al., 2013) show how sentiment can be exploited in query recommendation and discovering controversial topics. (Canuto et al., 2016) use new features for the sentiment analysis of small messages, like the information present in the sentiment distribution of the K-nearest-neighbors of a given message, etc. Similarly, (Bahrainian and Dengel, 2013) show that their method beat the state-of-the-art baseline by utilizing a Sentiment Lexicon to generate a set of features that help train a linear Support Vector Machine (SVM) classifier. (Kong et al., 2019) introduce a new method where the sentiment vector is concatenated with the output of the encoder present in the discriminator. This allows the model to generate a semantically correct response given a dialogue history and a sentiment label and control the polarity of the generated sentences. This shows that sentiment is a useful feature that carries information on how a polar sentence is formed.

3 Dataset

3.1 News and Wikipedia

News and Wikipedia, being the standard datasets available in Telugu, we have chosen these to establish baselines for the proposed GAN models. The statistics of the datasets are present in Table 1. (Jang and Yoon, 2018) discuss how the news data is connected to the keywords, repeats the same terms, and covers a limited number of topics. In contrast, Wikipedia data consists of general topics and a relatively varied set of sentences. The major reason behind experimenting on different datasets is to shed light on how well the models can learn and understand various features from different types of data.

3.2 Pravar

In order to create Pravar, we collected data from four different websites, namely, te.vikaspedia.in [4] an educative website, telugubaalalu.blogspot.com [5], podupukathalu.blogspot.com [6] and kathalu.wordpress.com [7] which are blogs dedicated to stories. Some of these websites contain stories written in other languages like Hindi, Bengali, etc., but we (the authors), proficient only in English and Telugu, chose to create a dataset specific to Telugu. The anonymized code and dataset can be found at https://bit.ly/3ihRig6. There are 1,823 stories in Pravar, where every story has 425 words and 50 sentences on average. The statistics of Pravar can be found in Table 1. Each story in Pravar has a Title, Story, and Moral. Characters, setting, narrative, conflict, and resolution are the five basic yet essential components of any tale. These important aspects keep the tale moving along smoothly and allow the action to unfold in a logical and understandable manner for the reader. Pravar can not only be used for solving the text generation problem but also help understand various aspects of stories like how sentiment flows from the narrative to conflict to resolution or how a sentence somewhere in the conflict is linguistically connected to a sentence in the narrative, etc.

4 Preprocessing

Once we collected the raw data, the next step was to preprocess it. Online data can have a lot of noise and can lead to a lot of bias during prediction if given to the model. Extraction of sentences from the raw news articles required removal of various non-ASCII characters, unnecessary details about the articles, etc. Raw

[4]https://te.vikaspedia.in/education
[5]http://telugubalalu.blogspot.com/
[6]http://podupukathalu.blogspot.com/
[7]https://kathalu.wordpress.com

Wikipedia articles consisted of code mixed sentences, email IDs, etc., which had to be filtered. In case of Pravar, we manually checked each story and removed the unnecessary information such as author details, morals, titles, epilogue, etc. Some stories had sentences in direct speech, which were also removed as they tend to be in a different tense. For all three datasets, we considered the sentences having a sequence length between 10 and 20.

4.1 Dataset Statistics

After preprocessing the datasets, we were left with 13,623 sentences in News, 10,941 sentences in Wikipedia and 8,830 sentences in Pravar to experiment upon. Other statistics can be found in Table 1.

Dataset	**News**	**Wiki**	**Pravar**
Words	1,321,794	958,877	775,366
Sentences	129,743	77,698	92,846
Documents	8,230	17,554	1,823
Avg-WPD	160	54	425
Avg-SPD	15	4	50
Nouns	594,036	417,179	232,604
Adjectives	22,914	33,924	17,443
Adverbs	7,871	8,343	8,685

Table 1: Statistics of the News, Wikipedia and Pravar datasets. Here document refers to a News/Wikipedia article or a story. Avg-WPD denotes the average number of words per document and Avg-SPD denotes the average number of sentences per document.

5 Experiments

5.1 Experimental setup

We performed multiple experiments using SeqGAN(SG) and LeakGAN(LG) as the base models. All the improvements and proposed novelty has been added on top of them. We used the default values from the original papers for the batch size, optimizer, and other hyperparameters. For models with SG as the base model, we trained the models for 150 epochs in Maximum Likelihood Estimation and 100 epochs in Adversarial training. For models with LG as the base model, we have trained the generator for 10 epochs in every epoch of 15 interleaved epochs in Maximum Likelihood Estimation and 100 epochs in Adversarial training. We calculated the BLEU, NLL-Gen and NLL-Div scores proposed by (Papineni et al., 2002), (Zhu et al., 2018) and (Liu et al., 2020), respectively, for all the models to help analyze their performance.

5.2 Features

5.2.1 Word embeddings

Instead of allowing the model to generate word embeddings dynamically by itself, we used the pre-generated embeddings and let the model train using these. We used the Word2Vec algorithm in (Řehřek et al., 2011)'s Gensim library to generate word embeddings for every word from the dataset. Using word embeddings helped boost the BLEU and NLL scores by a good factor because they pack a lot more information based on the words present around them, hence allowing the generator to generate better sentences. We experimented with vectors of three different dimensions, i.e., 50, 100, and 300 as shown in Table 2.

5.2.2 POS Tag vector

We used the Parts-of-Speech (POS) tagger provided by (Reddy and Sharoff, 2011) to tag the sentences. A unique index was assigned to each tag from the tagset[8]. Once the sentence is tagged, we create a pos tag vector with dimensions as the size of the sentence using the unique indexes. The POS tag specific statistics can be found in Table 1.

5.2.3 Sentiment vector

We used the Telugu-Senti-WordNet provided by (Das and Bandyopadhyay, 2010) for generating the sentiment vector. This lexicon contains 30,889 Telugu synsets. As multiple words present in our datasets were missing from the lexicon, we translated them to English using (Loria, 2018)'s TextBlob and took the polarity of that word instead of the exact sentiment value

[8]https://bitbucket.org/sivareddyg/telugu-part-of-speech-tagger/src/master/posguidelines.pdf

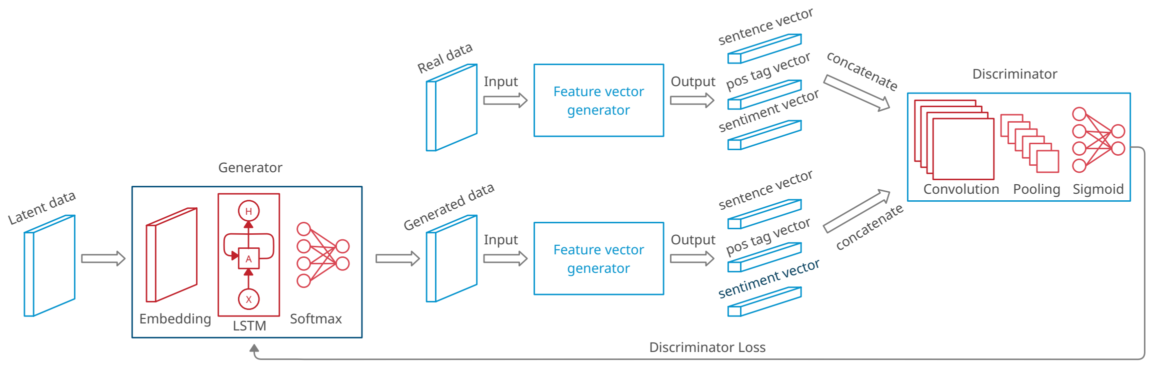

Figure 1: Architecture of the proposed PS-GAN. Sentiment and Parts-of-Speech features are provided to the discriminator so that it can help the generator to generate better sentences.

and mapped it to the Telugu word. We maintained three sentiment classes, namely positive, negative and neutral, where each class was given a unique index. We then generated a sentiment vector with dimensions as the size of the sentence using the indexes defined.

5.3 PS-GAN

As shown in Figure 1, the proposed PS-GAN architecture with SG as the base model takes in latent space information and tries to generate text. The generator essentially has an LSTM (Hochreiter and Schmidhuber, 1997) layer followed by a softmax layer at the end. The discriminator consists of a Convolution layer (Zhang et al., 2015) followed by pooling and sigmoid layers. Pre-trained embeddings are given to both the generator and discriminator. Sentences generated by the generator go through a feature generation module to produce a new vector called the '*generated_vec*'. The output of this feature module is a single concatenated vector consisting of the sentence, POS Tag, and the sentiment vectors. The concatenated feature vector is also generated for the real data and is called the '*real_vec*'. These two vectors go as inputs to the discriminator and the total loss is backpropagated to the generator. The intuition behind PS-GAN is that, if we can provide the discriminator with information about the structure of a sentence (Parts-of-speech tags) i.e., where a noun should be present or where an adjective should be present, the ability of the generator forming a linguistically correct sentence will increase drastically. Over time the discriminator will help the generator learn to form a sentence having a correct parse tree. Similarly, providing discriminator with the sentiment vector will help the generator understand the flow of sentiment within the sentence.

5.4 Leaky-PS-GAN

According to the intuition behind PS-GAN, the discriminator will have the POS Tag and sentiment information in its hidden layer, but it attempts to help the generator by backpropagating the loss. Instead of backpropagating the loss, if we can leak the hidden features in PS-GAN's discriminator directly to the generator, we would be able to generate larger and meaningful sentences. Figure 2 shows the architecture of Leaky-PS-GAN with LG as the base model.

6 Results and Analysis

Table 2 shows the BLEU and NLL scores obtained by experimenting with the baseline and the proposed models on all three datasets. The best scores have been highlighted in green for SG and blue for LG models. When embeddings are used (SG-E50, SG-E100, SG-E300), we can see a significant boost in the BLEU and NLL-Gen scores compared to the original SG models.

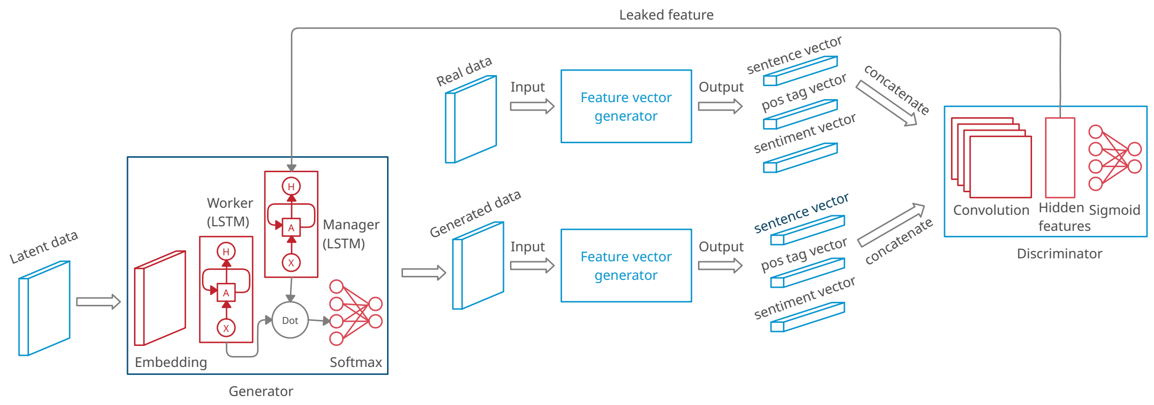

Figure 2: Architecture of the proposed Leaky-PS-GAN. Discriminator's hidden features are leaked to the generator to help generate better sentences.

However, the increment in these values is accompanied by a considerable drop in the NLL-Div values. As the BLEU metric assigns greater points to words that match sequentially i.e., if a string of four words in the produced phrase matches the given reference in the same precise sequence, it will have a greater positive influence on the BLEU score than a string of two matching words. This means that a correctly generated sentence might earn a lesser score if it utilizes different but related words. As a result of this, we can see higher BLEU scores in case of 100 and 300-dimensions. This is also evident from the NLL-Gen and NLL-Div values which tend to decrease as we move from 50 to 300-dimensional embeddings, indicating that the model tends to over-fit on the training data with an increase in the dimension size. On manual inspection of the sentences generated by SG-E100 and SG-E300, it was found that most of the sentences were already present in the training data. In the case of SG-E50, the generated sentences were better in quality and strikingly different from those already present in the training data. Although there is a slight decrease in the diversity as suggested by the dip in the NLL-Div value compared to SG, the sentences generated by SG-E50 are considerably better. This can be inferred from the BLEU and NLL-Gen values, making this a reasonable trade-off. As the results from all the three datasets point towards the same conclusions, we have performed further experiments using the 50-dimensional word embeddings.

Using embeddings (SG-E50) has shown a tremendous increase in the BLEU-5 scores, from 0.158 to 0.266 for News, 0.233 to 0.379 for Wikipedia, and 0.357 to 0.568 for Pravar over the original SG model. We can see a similar increase in the other BLEU scores as well, as shown in Table 2. The NLL-Gen values of SG-E50 have also significantly reduced from 2.4661 to 1.4409, 1.5896 to 1.0441, and 1.0274 to 0.6496 for News, Wikipedia, and Pravar, respectively, as compared to SG model. This is an expected result because generating Telugu text is a complex task because of the fact that it's morphologically rich. The model needs to have some extra information to understand the relationship between the words and generate meaningful sentences. As we can see, by adding word embeddings, the networks have more defined information about the context of the words, increasing the ability to generate good quality sentences.

On incorporating POS tag information into the discriminator, a significant increase in BLEU scores was seen in both SG (SG-EP) and LG (LG-EP) models over the SG-E50 and LG-E50 models, respectively, as shown in Table 2. In case of SG models (SG-EP), the BLEU-5

Dataset	Scores	SG	SG-E50	SG-E100	SG-E300	SG-EP	SG-EPS	LG-E50	LG-EP	LG-EPS
News	BLEU-2	0.74	0.77	0.89	0.963	**0.782**	0.779	0.826	**0.89**	0.871
	BLEU-3	0.463	0.536	0.75	0.913	**0.556**	0.553	0.651	**0.773**	0.75
	BLEU-4	0.259	0.371	0.631	0.868	**0.387**	0.397	0.535	**0.7**	0.678
	BLEU-5	0.158	0.266	0.546	0.83	**0.279**	0.298	0.467	**0.656**	0.64
	NLL-Gen	2.4661	1.4409	0.8345	0.5626	**1.3908**	1.4031	1.3123	**0.6704**	0.7
	NLL-Div	2.2028	1.632	0.9863	0.6781	**1.6658**	1.6373	0.8532	**0.9018**	0.8744
Wikipedia	BLEU-2	0.689	0.787	0.942	0.988	**0.821**	0.799	0.901	**0.925**	0.916
	BLEU-3	0.449	0.583	0.888	0.968	**0.638**	0.598	0.817	**0.861**	0.843
	BLEU-4	0.31	0.456	0.844	0.951	**0.518**	0.466	0.776	**0.825**	0.8
	BLEU-5	0.233	0.379	0.812	0.94	**0.443**	0.378	0.755	**0.805**	0.778
	NLL-Gen	1.5896	1.0441	0.5479	0.5084	**0.9444**	1.0328	0.7511	**0.7095**	0.7205
	NLL-Div	2.047	1.2985	0.6978	0.5416	**1.3121**	1.3178	0.674	**0.6723**	0.6871
Pravar	BLEU-2	0.788	0.889	0.977	0.992	0.904	**0.91**	0.929	0.952	**0.959**
	BLEU-3	0.589	0.742	0.939	0.977	0.789	**0.805**	0.864	0.906	**0.926**
	BLEU-4	0.445	0.635	0.907	0.965	0.704	**0.721**	0.828	0.882	**0.907**
	BLEU-5	0.357	0.568	0.888	0.955	0.642	**0.66**	0.808	0.869	**0.895**
	NLL-Gen	1.0274	0.6496	0.4868	0.4861	0.6321	**0.6276**	0.4805	0.464	**0.4865**
	NLL-Div	1.344	0.8681	0.5726	0.4847	0.8692	**0.8654**	0.6083	0.6019	**0.6037**

Table 2: These are the BLEU and NLL scores obtained for News, Wikipedia and Pravar datasets. SG is SeqGAN; SG-E50, SG-E100 and SG-E300 are SeqGan with 50, 100 and 300 dimensional embeddings; SG-EP is SeqGan with word embeddings and POS Tag vector, SG-EPS is SeqGan with word embeddings, POS Tag and sentiment vectors. LG is LeakGan; LG-E50 is LeakGan with 50 dimensional word embeddings. LG-EP is LeakGan with word embeddings and POS Tag vector, and LG-EPS is LeakGan with word embeddings, POS Tag and sentimenet vectors. BLEU-2, BLEU-3, BLEU-4, BLEU-5 denote the scores considering bi-grams, tri-grams, four-grams and five-grams respectively whereas NLL-Gen and NLL-Div denote the quality and diversity of the generated sentences.

scores have increased from 0.266 to 0.279, 0.379 to 0.443, and 0.568 to 0.642 in News, Wikipedia, and Pravar respectively, over the SG-E50 models. A similar increase in BLEU-5 scores of LP-EP models, from 0.467 to 0.656 for News, 0.755 to 0.805 for Wikipedia, and 0.808 to 0.869 for Pravar, is seen over the LG-E50 models. This is accompanied by, a decrease in NLL-Gen values from 1.4409 to 1.3908, 1.0441 to 0.9444, and 0.6496 to 0.6321 in SG models, and 1.3123 to 0.6704, 0.7511 to 0.7095, and from 0.4805 to 0.464 in LG models for News, Wikipedia, and Pravar respectively. This depicts the enhancement in the quality of the sentences generated. The overall increment in BLEU scores from E50 to EP/EPS models is on a higher side in LG models than in SG models. This is expected as these features are leaked from the discriminator to the generator in case of LG, whereas it is back-propagated as loss in case of SG. The higher quality and lower diversity of the sentences generated in case of LG models as compared to the SG models is in conjunction with the previously reported studies by (Zhu et al., 2018). Apart from showing a considerable improvement in BLEU and NLL-Gen scores, an important observation is that the NLL-Div values in case of EP models have either increased or are similar to that of E50 models. This is possible because, unlike word embeddings which make the model generate words surrounding a particular word, POS tag features help understand the formation of the sentence on a linguistic level and thus help generate sentences with better/similar diversity.

An increase in quality of the generated sentences with POS as a feature is quite apparent from the BLEU and NLL-Gen scores of E50 and EP models, but it is not so evident when considering sentiment (EPS). On average, the sentences in News, Wikipedia, and Pravar consist of about one to two polar words. As the maximum sentence length in this paper is only 20, the overall effectiveness of the sentiment feature is not prominent. Sentiment might start adding increased value when generating larger text such as whole articles or stories. In general, News and Wikipedia articles tend to have a lot of proper nouns such as the names of people, places, or things, i.e., a lot of words with a neutral sentiment. Due to this, adding sentiment showed no improvement in case in Wikipedia and News datasets. On the other hand, a slight improvement is seen in both SG and LG models for stories, as each sentence has a considerable number of words with some polarity (either positive or negative).

6.1 Human Evaluation

In order to make sure the scores we got were in line with the quality of the generated sentences, we performed a manual evaluation on the sentences generated by LG-E50 and LG-EPS models. We followed the scaling procedure as proposed in (Liu et al., 2020), by assigning a score between 1 - 5 according to the grammatical and semantic correctness. We considered 50 sentences per model-dataset pair over all the three datasets and averaged the scores. The human evaluation scores present in Figure 3 are seen to correlate with the BLEU and NLL scores in Table 2.

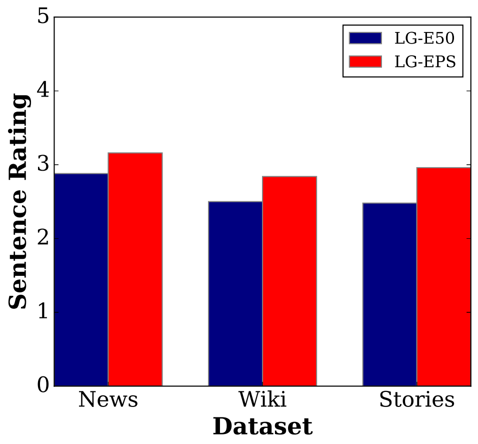

Figure 3: Bar plot showing human evaluation rating for LG-E50 and LG-EPS models over the three datasets.

6.1.1 Qualitative Examples

The below examples have been picked from sentences generated by LG-EPS GAN. Ratings mentioned for the examples are the averaged ratings post human evaluation.

Dataset - News

Sentence - జలవనరుల శాఖ ఇచ్చిన హామీలను నెరవేర్చకుండా వచ్చి ఎన్నికల కాలపరిమితి నివేదికను కోరుతూ కేంద్రానికి సిఫార్సు చేయాలని ఆమె ఆదేశించారు .

Translation - She ordered the Water Resources Department not to fulfill the given promises and to request the central government for the upcoming election's timeline report.

Explanation - The model was able to generate this sentence completely on its own by picking words from various contexts and joining them meaningfully.

Rating - 4.5

Dataset - Wikipedia

Sentence - కొన్ని సందర్భాలలో యుద్ధం వల్ల ప్రయోజనం లేదని గ్రహించి యుద్ధ చారిత్రిక పదార్థాలు కూడా ఈ తర్వాత పైపై సామర్థ్యం ఉంటుంది ..

Translation - Realizing that war is of no use in some situations, historical war materials will have lesser potential here after.

Explanation - In this example, the model has taken half the sentence from the dataset and completed it meaningfully with the relevant context. The words of the other half of the sentence were picked from different contexts spread all over the dataset. Although the sentence is holistically meaningful, the rating was penalized because of the incorrect grammar used while conflating the contexts.

Rating - 4

Dataset - Pravar

Sentence - ఒకరోజు ఆయన తన ముగ్గురు కొడుకుల్ని పిలిచి , ఒకరికి పనిచేస్తాను అన్నాడు .

Translation - One day, he called his three sons and said that he will work for one of them.

Explanation - In this example, the model was able to merge phrases from three different contexts and form a meaningful sentence.

Rating - 5

7 Conclusion and future work

In this work, we provide Pravar, a dataset consisting of stories. Telugu being a low resource language, Pravar can be used to solve various natural language problems apart from text generation, making it is an important contribution to the community. Even though Telugu is an agglutinative and morphologically rich language, the model was able to understand and generate good quality sentences. Further, we show that using POS tags and sentiment, as features in the proposed PS-GAN helped significantly boost the BLEU and NLL scores. Experimenting on varied datasets helped establish baselines and show that the model is able to learn well. We plan on extending the current work to generate larger sentences/paragraphs while maintaining context. When training the discriminator, incorporating features like n-gram based sentiment, emotions, etc., could further enhance the sentence quality. Another vital extension to this would be to increase the size of Pravar by adding stories from multiple other languages to make the dataset even more useful. Generating text in local languages would help in creating more content and reach out to a larger audience.

References

[Bahrainian and Dengel2013] Seyed-Ali Bahrainian and Andreas Dengel. 2013. Sentiment analysis using sentiment features. In *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, volume 3, pages 26–29. IEEE.

[Canuto et al.2016] Sérgio Canuto, Marcos André Gonçalves, and Fabrício Benevenuto. 2016. Exploiting new sentiment-based meta-level features for effective sentiment analysis. In *Proceedings of the ninth ACM international conference on web search and data mining*, pages 53–62.

[Che et al.2017] Tong Che, Yanran Li, Ruixiang Zhang, R Devon Hjelm, Wenjie Li, Yangqiu Song, and Yoshua Bengio. 2017. Maximum-likelihood augmented discrete generative adversarial networks. *arXiv preprint arXiv:1702.07983*.

[Chelaru et al.2013] Sergiu Chelaru, Ismail Sengor Altingovde, Stefan Siersdorfer, and Wolfgang Nejdl. 2013. Analyzing, detecting, and exploiting sentiment in web queries. *ACM Transactions on the Web (TWEB)*, 8(1):1–28.

[Chen et al.2018] Liqun Chen, Shuyang Dai, Chenyang Tao, Dinghan Shen, Zhe Gan, Haichao Zhang, Yizhe Zhang, and Lawrence Carin. 2018. Adversarial text generation via feature-mover's distance. *arXiv preprint arXiv:1809.06297*.

[Das and Bandyopadhyay2010] Amitava Das and Sivaji Bandyopadhyay. 2010. Sentiwordnet for indian languages. In *Proceedings of the eighth workshop on Asian language resouces*, pages 56–63.

[Gómez-Rodríguez et al.2019] Carlos Gómez-Rodríguez, Iago Alonso-Alonso, and David Vilares. 2019. How important is syntactic parsing accuracy? an empirical evaluation on rule-based sentiment analysis. *Artificial Intelligence Review*, 52(3):2081–2097.

[Goodfellow et al.2014] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*.

[Guo et al.2018] Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. 2018. Long text generation via adversarial training with leaked information. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

[Hochreiter and Schmidhuber1997] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Lstm can solve hard long time lag problems. *Advances in neural information processing systems*, pages 473–479.

[Iqbal and Qureshi2020] Touseef Iqbal and Shaima Qureshi. 2020. The survey: Text generation models in deep learning. *Journal of King Saud University-Computer and Information Sciences*.

[Jang and Yoon2018] Beakcheol Jang and Jungwon Yoon. 2018. Characteristics analysis of data from news and social network services. *IEEE Access*, 6:18061–18073.

[Jang et al.2016] Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

[Kong et al.2019] Xiang Kong, Bohan Li, Graham Neubig, Eduard Hovy, and Yiming Yang. 2019. An adversarial approach to high-quality, sentiment-controlled neural dialogue generation. *arXiv preprint arXiv:1901.07129*.

[Li et al.2018] Yang Li, Quan Pan, Suhang Wang, Tao Yang, and Erik Cambria. 2018. A generative model for category text generation. *Information Sciences*, 450:301–315.

[Lin et al.2017] Kevin Lin, Dianqi Li, Xiaodong He, Zhengyou Zhang, and Ming-Ting Sun. 2017. Adversarial ranking for language generation. *arXiv preprint arXiv:1705.11001*.

[Liu et al.2020] Zhiyue Liu, Jiahai Wang, and Zhiwei Liang. 2020. Catgan: Category-aware generative adversarial networks with hierarchical evolutionary learning for category text generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8425–8432.

[Loria2018] Steven Loria. 2018. textblob documentation. *Release 0.15*, 2.

[Nie et al.2018] Weili Nie, Nina Narodytska, and Ankit Patel. 2018. Relgan: Relational generative adversarial networks for text generation. In *International conference on learning representations*.

[Papineni et al.2002] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

[Reddy and Sharoff2011] Siva Reddy and Serge Sharoff. 2011. Cross language pos taggers (and other tools) for indian languages: An experiment with kannada using telugu resources. In *Proceedings of the Fifth International Workshop On Cross Lingual Information Access*, pages 11–19, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.

[Řehřek et al.2011] Radim Řehřek, Petr Sojka, et al. 2011. Gensim—statistical semantics in python. *Retrieved from genism. org*.

[Sreeram and Sinha2018] Ganji Sreeram and Rohit Sinha. 2018. Exploiting parts-of-speech for improved textual modeling of code-switching data. In *2018 Twenty Fourth National Conference on Communications (NCC)*, pages 1–6. IEEE.

[Wang and Wan2018] Ke Wang and Xiaojun Wan. 2018. Sentigan: Generating sentimental texts via mixture adversarial networks. In *IJCAI*, pages 4446–4452.

[Xu et al.2018] Jingjing Xu, Xuancheng Ren, Junyang Lin, and Xu Sun. 2018. Diversity-promoting gan: A cross-entropy based generative adversarial network for diversified text generation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3940–3949.

[Yu et al.2017] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31.

[Zhang et al.2015] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *arXiv preprint arXiv:1509.01626*.

[Zhu et al.2018] Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1097–1100.