Rich Word Embedding for Text Generation Using Recurrent Neural Networks

Samuel Remedios
Department of Computer Science
Middle Tennessee State University
Murfreesboro, TN, USA
Center for Neuroscience and Regenerative Medicine
Henry M. Jackson Foundation
Bethesda, MD, USA
swr2u@mtmail.mtsu.edu

Madison Karrh
Department of Computer Science
Middle Tennessee State University
Murfreesboro, TN, USA
xxxx@mtmail.mtsu.edu

Michael Schmidt
Department of Computer Science
Middle Tennessee State University
Murfreesboro, TN, USA
xxxx@mtmail.mtsu.edu

Brent Perez
Department of Computer Science
Middle Tennessee State University
Murfreesboro, TN, USA
xxxx@mtmail.mtsu.edu

John Westbrooks
Department of Computer Science
Middle Tennessee State University
Murfreesboro, TN, USA
xxxx@mtmail.mtsu.edum

Joshua Phillips
Department of Computer Science
Middle Tennessee State University
Murfreesboro, TN, USA
joshua.phillips@mtsu.edu

***Abstract*—Language modeling and text generation have been studied for decades, but more recently deep learning and other machine learning techniques have been applied to these tasks. Although work has been done with Markov modeling and recurrent neural networks (RNNs), not much progress has been made with higher-level linguistic structure such as subject-participle construction or parts of speech. Our work infuses standard word embedding techniques with the respective part of speech, improving the sensibility of generated text. We compare structure between this enriched word embedding technique applied to RNNs to word-level and character-level Markov chains, non-enriched word embedding-based RNNs, and character-level RNNs. Qualitatively the generated sentences are more sensible, and using [distance metric 1 and 2] our method performs $xx.xx\%$ better. By including the part of speech alongside the word embeddings, our model learned representations quickly, converging to [$xxxx$] within YY epochs, implying that this type of higher level encoding is useful for learning language.**

***Index Terms*—text generation, recurrent neural network, natural language processing, language modeling, computational linguistics, part of speech tagging, word2vec, word embedding**

I. Introduction

Semantic understanding of natural language has been a difficult problem in computer science for decades. [1] While many great strides have been made in many subfields such as sentiment analysis [cite 2], word embeddings [2], text generation [3], classification [4], and summarization [5], there still lacks an effective means of human-level text modeling [6] [7] [8]. Recently, more tools have become available to help in this endeavour [2] [9] [10], and the aggregation of techniques is now possible. One such example is the generation of a caption given an image [11], which uses a convolutional neural network in tandem with an RNN, trained on natural language data parsed and tokenized. They use dependency trees as a way of building vectors which embedded both word and part of speech context at once, also incorporating neighboring words and their contexts.

Currently, the more popular generative methods aim to model languages from the character-level upwards rather than from the word-level. Even those that operate on a word-by-word basis (Markov chain modeling [12], n-gram models [13]) may not incorporate parts of speech nor long-term memory. Traditionally, these word-level models build internal representations of language by building a probabilistic mapping between previous words in the sequence to the current word. However, language has more depth than the surface-level words present, such as semantics, grammar, parts of speech, etc, and it would be advantageous to encode this information alongside the words themselves.

For this paper, we propose a method of tokenizing words alongside their parts of speech with unique identifiers for use in a recurrent neural network (RNN). Specifically, we aim to generate text using the long short-term memory (LSTM) variant of the RNN using training data comprised of scripts from popular American cartoon television shows, such as the Simpsons, Family Guy, Rick and Morty, South Park, and others. We narrowed the genre of script to be somewhat specific to see if we can model the general flow and cadence of comedy shows, which should have more related elements in storytelling and dialogue compared to a mix of genres such as comedy, drama, action, and so on. This method of encoding natural language has an advantage over the previously mentioned dependency trees by being fully transparent and customizable for the user; it is possible to change any component of this white box approach in order to achieve the desired goal.

TABLE I
DISTRIBUTION OF NUMBER OF IMAGES FOR THE THREE PROPOSED TASKS.

Metric	Rick and Morty	# The Simpsons	# South Park
Total Words	1000	20000	50000
Unique Words	50	500	1500
Outlier Words	20	20	40
Word Complexity	?	?	?

II. METHODS

A. Data

The data used in this study was a compilation of transcripts from the American television series South Park, the Simpsons, and Rick and Morty to reduce the variety of generated text. These transcripts were found online at CITE WEBSITES HERE. Our intent was to collect data which was thematically similar and had similar structure to each other, and that approximately 20 minute comedy cartoons would have similar word distributions. The number of unique words and outlier words are outline in Table II-A.

B. Preprocessing

As the data was found via websites, there was much heterogeneity in the scripts. We therefore had several cleaning steps to ensure the data could be handled by our model properly. First we programmatically removed extraneous white space. Second, we appended special ¡START¿ and ¡END¿ tokens to the beginnings and ends of each sentence, to allow the model to easily learn when sentences should begin and end. Third, all words were converted to lowercase.

At this time we fit a word2vec model over our corpus, and made a look-up table (LUT) which contained mappings between the vocabulary in our corpus and their word embedding. Our word2vec model was trained for $10,000$ epochs and generated a unique 128-dimensional vector for each word.

To finalize preprocessing, we used NLTK to part of speech tag each word in our corpus, converted these parts of speech into one-hot vectors, then appended them to their appropriate word vector.

C. Recurrent Neural Network Architecture

This is where we'll talk about how the RNN was designed and why. Inputs and outputs are defined, training epochs, loss function, optimizer, etc. How does learning occur?

III. RESULTS

To validate our model, we will first qualitatively inspect the generated scripts. We will then quantitatively measure the Levenshtein distance (or other similar metric) between the tokens and parts-of-speech of our training corpus and the generated scripts. In particular, we will generate around some fixed number of sequential words (500-1000), and verify that these generated words' parts of speech are within some similarity range to other subsets of real text compared to other subsets of real text. For example, comparing some fixed-size subsets of Family Guy data to Simpsons data will give us understanding as to how much the Levenshtein part of speech distance can vary among real datasets. We then will use this metric to validate our generative model. Additionally, we are contemplating metrics such as BLAST which compare genetic data, and will utilize them if they are applicable to this type of sequence data. If the metric(s) is/are sensible, we will then compare our methods to already-existing text generation techniques such as character-level RNNs and Markov chain models to judge our model's efficacy.

Here we need a table of quantitative results, and then some subsections showcasing the generated sentences from each method.

IV. DISCUSSION

The ability to automatically generate text is two-fold. First, if the results are reasonable, then it may be advantageous to further investigate how the language was modeled using this technique. Second, the automatic generation of stories and scripts could potentially be used as a starting point for writers, or smooth the creation process.

We found that adding the parts of speech as grammatical context made much more reasonable sentences. Etc. Talk more about what's going on after we ge the results and can see.

ACKNOWLEDGMENT

We'd like to thank the Department of Computer Science at Middle Tennessee State University for hosting this class.

REFERENCES

[1] P. Linell, *The written language bias in linguistics: Its nature, origins and transformations*. Routledge, 2004.

[2] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *CoRR*, vol. abs/1310.4546, 2013. [Online]. Available: http://arxiv.org/abs/1310.4546

[3] I. Sutskever, J. Martens, and G. E. Hinton, "Generating text with recurrent neural networks," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011, pp. 1017–1024.

[4] S. T. Dumais, D. Heckerman, E. Horvitz, J. C. Platt, and M. Sahami, "Methods and apparatus for classifying text and for building a text classifier," Feb. 20 2001, uS Patent 6,192,360.

[5] J. Goldstein, M. Kantrowitz, V. Mittal, and J. Carbonell, "Summarizing text documents: Sentence selection and evaluation metrics," in *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '99. New York, NY, USA: ACM, 1999, pp. 121–128. [Online]. Available: http://doi.acm.org/10.1145/312624.312665

[6] L. Wenyin, X. Quan, M. Feng, and B. Qiu, "A short text modeling method combining semantic and statistical information," *Information Sciences*, vol. 180, no. 20, pp. 4031 – 4041, 2010.

[7] J. Yan, J. Zeng, Z.-Q. Liu, L. Yang, and Y. Gao, "Towards big topic modeling," *Information Sciences*, vol. 390, pp. 15 – 31, 2017.

[8] P. Bicalho, M. Pita, G. Pedrosa, A. Lacerda, and G. L. Pappa, "A general framework to expand short text for topic modeling," *Information Sciences*, vol. 393, pp. 66 – 81, 2017.

[9] S. Bird and E. Loper, "Nltk: the natural language toolkit," in *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*. Association for Computational Linguistics, 2004, p. 31.

[10] M. Honnibal and M. Johnson, "An improved non-monotonic transition system for dependency parsing," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 1373–1378.

[11] J. Johnson, A. Karpathy, and L. Fei-Fei, "Densecap: Fully convolutional localization networks for dense captioning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4565–4574.

[12] J. M. Conroy and D. P. O'leary, "Text summarization via hidden markov models," in *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2001, pp. 406–407.

[13] H. Masataki and Y. Sgisaka, "Variable-order n-gram generation by word-class splitting and consecutive word grouping," in *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, vol. 1. IEEE, 1996, pp. 188–191.