Group "Breaking Bad" – Final Report

Explore Literature though Natural Language Processing

Stephen Brand
sbrand6@gatech.edu
Oregon, USA

Bo Cheng
bcheng66@gatech.edu
Texas, USA

Jonathan Compton
jcompton33@gatech.edu
New Mexico, USA

Corey Haverda
chaverda3@gatech.edu
Oregon, USA

Daniel Kim
danielkim@gatech.edu
Seoul, KOR

William Taylor
wtaylor68@gatech.edu
Virginia, USA

fryetag
natural literature processing

Figure 1: Application Name & Logo

1 INTRODUCTION

Our goal is to use interactive visualization to derive hidden literary characteristics and connections between works by analyzing book contents.

Using various analytical tools, we set out to accomplish the following four innovations. The first two are hard goals we were able to accomplish in the time-frame of this project. The latter two were "stretch goals" that we were unable to accomplish within the project window but have provided guidelines and next steps in our evaluation of results.

1. Predict who authored a work of literature based off words in the given work
2. Unsupervised learning drawing out top 10 topics found in the text
3. Named entity recognition used to extract features and use in exploration
4. Genre classification using a combination of topics and entities, giving users another way to filter

We have used Latent Dirichlet Allocation (LDA) and Generative Pre-trained Transformer 2 (GPT2) to accomplish our first two goals. We see the combination of LDA topic modeling, GPT2 author prediction, and streamlit.io for interactive visualization as new for the use of exploring literature through computationally driven analytical methods. How we relate works to one another, using the literature itself versus needing users to complete surveys, is different than what is being done today. There are three main reasons that give us confidence in our project. First, we are utilizing a platform (Project Gutenberg) that is well vetted and has been used to extract quality metadata [11]. Second, we believe in our classifications due to research which precedes us [1, 23]. Lastly, we have used those proven classification models to accomplish our exploratory goals in unique ways. The last two goals were found to be out of scope for our time-frame, but are possible to achieve with what is currently in place.

The totality of the project took eight weeks with a team of six. A high level view of tasks and estimated time to complete each can be found in Table 1 below. All members exhibited an equal amount of effort to ensure the successful implementation and delivery of the project.

Our midterm check was to run a Naive model to ensure the underlying cloud services and other infrastructures are operational. The Naive model also served as a unit of measurement for our author prediction validation, which was 82% f1-score. The final check was acceptance testing by our internal user groups, which will be described in more detail under section four.

Step in the process	Est. Time (hrs)	Run. Total (hrs)
Extract, Transform, Load	120	120
Model classifier to identify metadata	240	360
Test model for satisfactory accuracy	240	600
Map output to visualization	120	720
Wrap and prepare for deployment	200	920
Test UI and maintain the model	40	960

Table 1: Estimated Time to Complete

2 MOTIVATION

Research has been conducted on the positive effects reading has on identity [13]. Therefore, anyone with an interest in personal growth but with little time to find a good book will care about our innovative exploration application. A difficult thing about reading is finding books one may like. This tool offers a new way to find literature without sticking to the same group of authors or genres. Due to its nature, our product may be of interest to teachers, book store owners, psychologists, and bibliophiles.

The ability to encourage users to read more and discover works they may not be familiar with is one aspect that motivated us to complete this project. Another motivation is in helping users discover books because they have a potential commonality with entities such as characters or locations. By breaking a work of literature into it's unique words, users can use their own human intuition to assess whether or not they would like to discover a work based on the top words found. The risk we face is having readers discover books that are too similar and begin creating a literature bubble for themselves, not expanding out to read works with topics they wouldn't be initially drawn to.

3 PROBLEM DEFINITION

There are currently many great ways to discover a new work of literature. Few are analytically driven. Of those, most focus on authors who are like each other, or literary works with similar genres [27]. The problem our project works to solve is how to quickly and efficiently produce internal features that will allow readers to explore literature in new ways. We will start by predicting who wrote a piece of literature based off the text found in that literature. We will then attempt to classify topics covered in the work. Our main goal is exploration and our stretch goal is recommendation by linking the internal structure of literary works together.

Currently our research indicates little to no interactive visualizations for topic modeling within large bodies of text. Topic modeling does exist in the world of articles and research papers, but even that is rare [2]. There are instances where one can explore themes related between works as well as authors based on user preferences. The latter item can be found, for example, at www.gnod.com, where users are recommended authors based on which authors they like compared to other user surveys. This is most likely using some kind of affinity analysis or graph-based system commonly found in music recommendation [20]. As far as our stretch goal of entity recognition is concerned, there are many experiments which have been done on the best methods, which one can read about in the following references [9, 17, 22, 29].

4 LITERATURE SURVEY

In preparation of the project, we conducted research on models in current use for our Natural Language Processing (NLP) tasks, such as work already conducted in genre classification [19, 27] and various entity recognition tasks. One approach was to use TF-IDF scores to identify main characters and even visualize their frequency in the analyzed work [6, 25]. We did end up using this in order to visualize word frequency as seen in our application. Maximum Entropy Markov Models are good for information segmentation and have proven successful with named entity recognition but have a known issue of label bias [22]. Conditional Random Fields [17] address the issue of label bias by calculating the joint probability of pairs, implementing efficient feature selection. Those are good for training subsets of data and may not be as useful for fully observed data, pushing us to research other methods. There are studies on whether to combine classification and extraction [9] or to separate the tasks [29]. By combining the task we may get a more speedy and less accurate model, while the opposite is true for splitting the task up. Finally, one study we found provides an efficient way to sample graphs [21] using subsets of a larger graph.

Although the main focus of our project is literature exploration, we hoped to attempt to identify the type of literary work the protagonist is in using the framework modeled after Frye's theory of modes [10]. While this proved to be outside of the scope of our project, our current working models offer a good framework for anyone that might decide to pursue such goals. In that case, a skip-gram model for efficient distributed vector representation [24] is one way to complete the task. One form of word2vec uses the skip-gram architecture, which is supposed to perform well on medium sized datasets [1]. A better use may be the extension of word2vec, called doc2vec, which is designed to scale with large datasets [18]. There are pre-trained models, like BERT or XLNet [7, 28], for NLP tasks which have the benefit of saving time, but the Project Gutenberg texts are pre-1900s [11] which could be a roadblock. To boost our probability functions, neural networks have proven to improve state-of-the-art on various NLP tasks [3], as well as improve the preservation of linear regularities among words [23].

Lastly, we look at visualization techniques for NLP already in existence. We find we can build on our proposed NLP methods, such as TF-IDF scoring [12] or using vec2graph in conjunction with word2vec/doc2vec [15]. A survey of NLP visualizations from political speeches shows there is much to innovate in this area, such as providing interactivity, as well as displaying information in graph form [14].

Ultimately, we ended up using LDA for topic modeling, which has been a proven tool for exploratory analysis of a large number of papers[8, 16]. One such group set out to complete what we have done here, but for the use of searching algorithms and topic sorters for more technical works [2]. We also decided to pair GPT2 modeling as a way to predict an author based off pre-trained language models, which are usually not used for art, but for task-oriented systems [4].

5 PROPOSED METHOD

5.1 INTUITION

Our research has lead us to very little analytically driven approaches to exploring literature. What is more, none of those approaches allowed for an interactive web-page. The current state of the art for topic discovery in literature is to read a book. Our application handles all of the pre-processing, machine learning, and visualization. What is more, our application is a framework for more and more text. Our current project has over 1,300 books with successful classification and unsupervised learning models, but the framework is set in place to allow a relatively simple process of adding more books. The models we chose can be shifted to observe more or less topics quickly, and one such algorithm even has a user adjustable lambda value built into our application.

5.2 APPROACHES

A visualization of the model architecture can be seen in Figure 2. The first major task was extracting and storing a large body of text. The project Gutenberg library was chosen for its rich metadata and use is research projects [11]. The aforementioned article leads to a download of a "curated version of the complete [Project Gutenberg library] containing more than 50,000 books and more than 3×10^9 word-tokens." Regex was used to clean the tokens further to suit our needs. We made sure to split apart contractions and cleaning up as much archaic speech as possible. The latter was necessary to perform since much of the literature is from before the 1900's and we want users to relate to the diction found in topic modeling. We then split the works up by author, title, and text, creating a dictionary of complete books. We made sure to match book titles to ISBN numbers found on ISBNdb. ISBN stands for "International Standard Book Number". ISBNs are calculated using a specific mathematical formula and include a check digit to validate the number. They are unique identifiers that help us ensure we are working with the correct text. We then calculated the number of books per author and ran the rest of our analysis on the top 20 prolific authors in our entire dataset. This was a decision made to reduce scope and prove our application. Should work (and financial support) continue, we have the ability to add in the rest of the books found in PG.

Next, we randomized and split our texts using scikit-learn, with 20% pointed at testing and 80% on training. Stop words were defined using Natural Language Toolkit (NLTK) and filtered out during tf-idf vectorization. Words, now cleaned and separated, were trained on a voting classifier. Our voting classifier is based off a combination of Logistic Regression, Stochastic Gradient Descent Classifier, and Naive Bayes, all using sci-kit learn. Our voting classifier was able to predict an author based off a string of randomly generated (or user input) text. Overall, our classifier produced a precision of 84% and an overall accuracy of 82%. Interestingly, George Alfred Henty had the worst unique precision score, most likely because of the wide range of topics he covers through his works, as seen in his topic model in our application.

In order to get our topics we used latent Dirichlet allocation (LDA). We chose to go with ten topics for the entirety of an author's work and five topics per book. All alphas were set to auto, passed over five times, and analyzed in chunks of 100 words. Each author and book received it's own HTML file which contains an inter-topic distance map and bar chart of top-30 most salient words. The inter-topic distance map projects topic clusters as circles. We are using a default principal component analysis (PCA) model to extract the first two components on a topic-term distribution distance matrix. The distance between circles indicates how similar a topic is to one another. The area of the circles is proportional to the proportions of the topics across the total number of tokens in a given corpus. The top-30 most salient terms represent how relevant and salient terms are for a selected topic. In their static state, salient term bars are blue. The blue bars represent the most salient terms for a selected topic. Saliency is defined as how distinctive a term is for a selected topic. The distinctiveness of a word can be found by computing the likelihood of observing a word in a given topic $P(T|w)$ and multiplying that by the log of the conditional probability divided by the marginal probability $P(T)$. Putting it all together gives us the equation below.

$$distinctiveness(w) = \sum_{T} P(T|w) \log \frac{P(T|w)}{P(T)}$$

Saliency is simply the frequency of a word appearing across topics multiplied by it's distinctiveness score above. This means if a word appears across all topics, it will receive a low score, potentially not carrying as much impact to the topic [5].

When a topic is selected, Some words shift and red bars appear. The red bars represent the most relevant terms for the selected topic. Relevance ranks terms within topics for topic interpretation. The formula uses lambda which determines the weight given to the probability of term w under topic t relative to its lift (high ratio of saliency to relevance), measuring both on the log scale. Setting $\lambda = 1$ will rank terms in decreasing order of their topic-specific probability. Setting $\lambda = 0$ will rank terms based on their lift [26]. Our interactive visualizations allow users to set their own lambda value.

Our user interface is built off the streamlit.io python application engine. All LDA models mentioned above are interactive, allowing users to explore topic circles and words. We have a GPT2 engine which showcases the voting classifier author prediction. Alternatively, we added a section where a user may enter their own string of text and see which author is predicted. We have named this section "Sound Like an Author" and believe it will appeal to writers who want to explore authors who may reflect their own writing style. We were also able to use Wikipedia data in combination with SPARQL to scrape birthplace location data per author, which was then fed into a map. This was done in the case that users would be interested in that piece of trivia, but also to prove our ability to work with mapped data in the future.

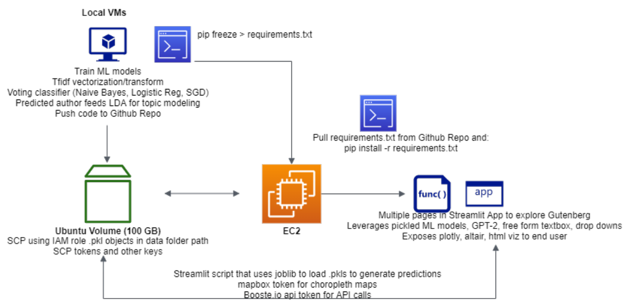
Local VMs
pip freeze > requirements.txt
Train ML models
Tfidf vectorization/transform
Voting classifier (Naive Bayes, Logistic Reg, SGD)
Predicted author feeds LDA for topic modeling
Push code to Github Repo
Pull requirements.txt from Github Repo and:
pip install -r requirements.txt
Ubuntu Volume (100 GB)
SCP using IAM role .pkl objects in data folder path
SCP tokens and other keys
EC2
func( )
app
Multiple pages in Streamlit App to explore Gutenberg
Leverages pickled ML models, GPT-2, free form textbox, drop downs
Exposes plotly, altair, html viz to end user
Streamlit script that uses joblib to load .pkls to generate predictions
mapbox token for choropleth maps
Booste.io api token for API calls

Figure 2: Model Architecture

6 EXPERIMENTS & EVALUATION

6.1 TESTBED

We developed a testbed of thirteen questions for a user group to answer after using the FryeTag application. The main objective of these questions was to discover how helpful of a tool Fryetag is, who it can help, how it can help, FryeTag's innovation, and how the application can continue to develop.

The subjects selected an author from our top 20 authors list and spent five minutes on the Project

Gutenberg main website, trying to identify as many topics as possible from the author's work. Topics are defined by common themes that have similar subject matter or word choices. Testing subjects then navigated FryeTag and repeated the same test as before. Finally, subjects compared their results from their discovery time on Project Guttenberg and FryeTag. The post-test survey was modeled on five-point scale ranging from 1-5, where 1 represents "Strongly Disagree", 5 represents "Strongly Agree", and an answer of 3 is "Neutral". Figures 3 and 4 show some of the results we will speak about in the following sub-section.

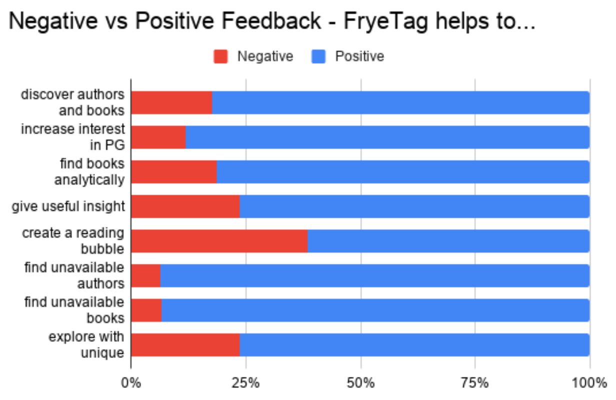

Figure 3

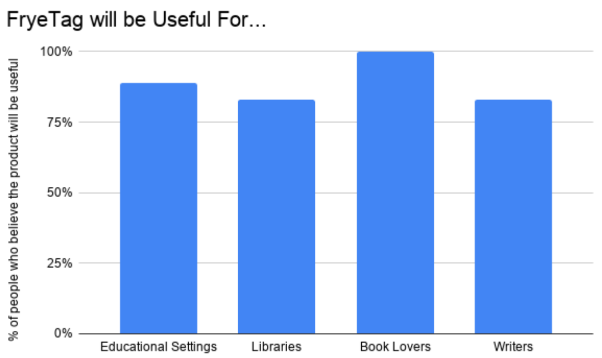

Figure 4

6.2 OBSERVATIONS

The most positive outcomes were the increased interest our application generated for the Project Gutenberg library, which offers over 60,000 free e-books to the general public, and the ability for users to explore authors in new ways that were previously unavailable. In the case of both of those records, 78.95% of subjects gave a positive response. Alternatively, our most negative reaction was for the question, "FryeTag may lead to a literature bubble for myself." Less than half of the respondents indicated that they agreed with this statement. This was also our most neutral response question, which indicates to us that the majority of users are not thinking about, or worried about, being stuck in a literature bubble as we had previously thought.

When asked, all of our subjects indicated that the tool would be useful for book lovers, our target audience. Users also returned a high likelihood that our tool would be useful for writers, libraries, and educators. Most users (77.8%) indicated they would like to explore books on theme. The next step in our process, should scope have allowed, would be to name topics based on our analysis of the words per topic. This would lead us into categorizing themes and be in line with what our test subjects indicated they want. A smaller amount (38.9%) of users indicated they want to explore books by protagonist, which was a stretch goal for our project. If we were to dive deeper into this product, that may remain low on the priority list. Half of our subjects indicated that they would like to explore books by era, and half indicated that they would like to explore books by location.

Our subjects felt the most strongly about our analytical approach and interactive visualization. Over half (52.63%) of our respondents indicated they strongly agreed that the "Sounds like the Author" page was a unique interactive tool they could keep coming back to. That tool has more use than it seems at first glance. For example, one may enjoy Jack London, and in particular, his nautical novel entitled *The Sea Wolf*. We can discover from topic modeling that the book stands apart from his other works (See figure 3 for an example). If one were to type in some words that are in the topic (sea captain wolf ship island) R M Ballantyne is returned. Given the topic modeling and Top 30 words, one would make an assumption that he writes more nautical novels, which is why he would show up over London. His book *Man on the Ocean* seems to match, since its largest topic falls in the same section as *The Sea Wolf*, and even gives me some more obscurity since the topics are distributed in the far corners of quadrant two and four as well. Looking at the words, It is a safe assumption that one would get the nautical adventure they are looking for. In this way, we have used the "Sound Like an Author" page to discover related works. Since such a large percentage of our subjects strongly agreed they liked this feature, it is safe to say we should put investment on that model at top priority.

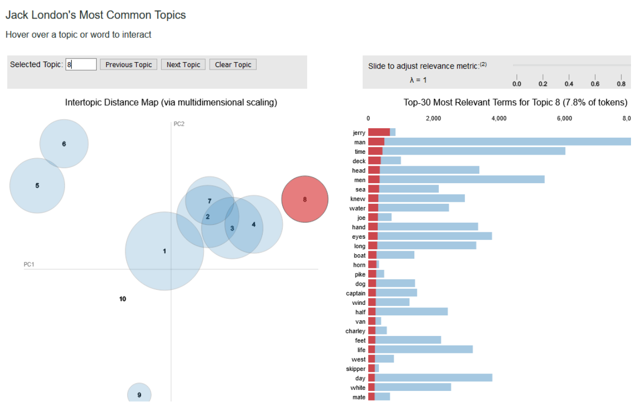

Figure 5: It would be safe to assume that *The Sea Wolf* is an influencer of Topic 8 due to the heavy use of nautical words

As far as the user interface is concerned, the majority of our subjects gave positive feedback about how clear the application is to use. A majority of our subjects also gave positive feedback about the balance of simplicity and depth that our visualizations portrayed. We received more neutral and negative feedback on the aesthetic theme. We want users to feel comfortable if they would be spending large amounts of time in our application, so a re-evaluation of our colors, fonts, and layout would prove to be a good use of time.

7 CONCLUSION & DISCUSSION

At first glance, our application may seem like an analytically pompous literature device. "Topics" are generally unknown and up for interpretation. It is difficult to discern what exactly is happening with the spherical map and random words. However, after playing with the application for some time, one can begin to discover how machine learning can make a way for art discovery. Most models these days center around how to decrease waiting time for customers, how to classify and store tickets, or how to sell more shoes to people with shoes. Our team's vision really began with the notion of merging two worlds that don't always belong together. It is interesting to observe Sir Arthur Conan Doyle on the topic map. All of his works are very grouped but one. Most likely it is because that topic represents his non-fiction works, like *The Great Boer War* or *The War in South Africa: Its Cause and Conduct*. Doyle is known for his great Sherlock Holmes. His topic map tells a story of a man whose writing of Holmes spilled out into much of his work. Yet, according to our models, his non-fiction was so far from his fiction. We can begin to truly appreciate Sir Arthur Conan Doyle for his diversity and his ability to separate himself from fantasy when the time called for it. He was so much more than the author of Mr. Holmes; he was a story-teller through and through. Of course, this is and can be up for debate. There will never truly be a solid answer on the literary criticism of anyone. The beauty of art is everyone gets a say. This is also the beauty of unsupervised learning. In many ways, the black box is like the mind of an author, and what we receive back is for all of us to interpret.

References

[1] Edgar Altszyler, Mariano Sigman, Sidarta Ribeiro, and Diego Fernández Slezak. 2016. Comparative study of LSA vs Word2vec embeddings in small corpora: a case study in dreams database. *arXiv preprint arXiv:1610.01520* (2016).

[2] Claus Boye Asmussen and Charles Møller. 2019. Smart literature review: a practical topic modelling approach to exploratory literature review. *Journal of Big Data* 6, 1 (2019), 93.

[3] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research* 3, Feb (2003), 1137–1155.

[4] Paweł Budzianowski and Ivan Vulić. 2019. Hello, It's GPT-2–How Can I Help You? Towards the Use of Pretrained Language Models for Task-Oriented Dialogue Systems. *arXiv preprint arXiv:1907.05774* (2019).

[5] Jason Chuang, Christopher D Manning, and Jeffrey Heer. 2012. Termite: Visualization techniques for assessing textual topic models. In *Proceedings of the international working conference on advanced visual interfaces*. 74–77.

[6] Djavan De Clercq, Ndeye-Fatou Diop, Devina Jain, Benjamin Tan, and Zongguo Wen. 2019. Multi-label classification and interactive NLP-based visualization of electric vehicle patent data. *World Patent Information* 58 (2019), 101903.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[8] Dag Elgesem, Lubos Steskal, and Nicholas Diakopoulos. 2015. Structure and content of the discourse on climate change in the blogosphere: The big picture. *Environmental Communication* 9, 2 (2015), 169–188.

[9] Radu Florian, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. 2003. Named entity recognition through classifier combination. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*. 168–171.

[10] Northrop Frye. 2020. *Anatomy of criticism: Four essays*. Princeton University Press.

[11] Martin Gerlach and Francesc Font-Clos. 2020. A standardized Project Gutenberg corpus for statistical analysis of natural language and quantitative linguistics. *Entropy* 22, 1 (2020), 126.

[12] Leandro S Guedes, Rafael Garcia, Bruno Pagno, Luciana Nedel, Joao Comba, and Carla MDS Freitas. [n. d.]. Interactive Timeline Visualization of Documents. ([n. d.]).

[13] Johannes Kaiser and Thorsten Quandt. 2016. Book lovers, bibliophiles, and fetishists: The social benefits of heavy book usage. *Psychology of Popular Media Culture* 5, 4 (2016), 356.

[14] Paritosh D Katre. [n. d.]. NLP Based Text Analytics and Visualization of Political Speeches. ([n. d.]).

[15] Nadezda Katricheva, Alyaxey Yaskevich, Anastasiya Lisitsina, Tamara Zhordaniya, Andrey Kutuzov, and Elizaveta Kuzmenko. 2019. Vec2graph: a Python library for visualizing word embeddings as graphs. In *International Conference on Analysis of Images, Social Networks and Texts*. Springer, 190–198.

[16] Olessia Koltsova and Sergei Koltcov. 2013. Mapping the public agenda with topic modeling: The case of the Russian LiveJournal. *Policy & Internet* 5, 2 (2013), 207–227.

[17] John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. (2001).

[18] Jey Han Lau and Timothy Baldwin. 2016. An empirical evaluation of doc2vec with practical insights into document embedding generation. *arXiv preprint arXiv:1607.05368* (2016).

[19] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International conference on machine learning*. 1188–1196.

[20] Kibeom Lee and Kyogu Lee. 2015. Escaping your comfort zone: A graph-based recommender system for finding novel recommendations among relevant items. *Expert Systems with Applications* 42, 10 (2015), 4851–4858.

[21] Xuesong Lu and Stéphane Bressan. 2012. Sampling connected induced subgraphs uniformly at random. In *International Conference on Scientific and Statistical Database Management*. Springer, 195–212.

[22] Andrew McCallum, Dayne Freitag, and Fernando CN Pereira. 2000. Maximum Entropy Markov Models for Information Extraction and Segmentation.. In *Icml*, Vol. 17. 591–598.

[23] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[24] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[25] Juan Ramos et al. 2003. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, Vol. 242. New Jersey, USA, 133–142.

[26] Carson Sievert and Kenneth Shirley. 2014. LDAvis: A method for visualizing and interpreting topics. In *Proceedings of the workshop on interactive language learning, visualization, and interfaces*. 63–70.

[27] Joseph Worsham and Jugal Kalita. 2018. Genre identification and the compositional effect of genre in literature. In *Proceedings of the 27th International Conference on Computational Linguistics*. 1963–1973.

[28] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*. 5753–5763.

[29] Xiaoshi Zhong, Erik Cambria, and Jagath C Rajapakse. 2018. Named Entity Analysis and Extraction with Uncommon Words. *arXiv preprint arXiv:1810.06818* (2018).