Genre Identification and the Compositional Effect of Genre in Literature

Joseph Worsham
Department of Computer Science
University of Colorado Colorado Springs
1420 Austin Bluffs Prkwy
Colorado Springs, CO 80918, USA
`jworsha2@uccs.edu`

Jugal Kalita
Department of Computer Science
University of Colorado Colorado Springs
1420 Austin Bluffs Prkwy
Colorado Springs, CO 80918, USA
`jkalita@uccs.edu`

Abstract

Recent advances in Natural Language Processing are finding ways to place an emphasis on the hierarchical nature of text instead of representing language as a flat sequence or unordered collection of words or letters. A human reader must capture multiple levels of abstraction and meaning in order to formulate an understanding of a document. In this paper, we address the problem of developing approaches which are capable of working with extremely large and complex literary documents to perform Genre Identification. The task is to assign the literary classification to a full-length book belonging to a corpus of literature, where the works on average are well over 200,000 words long and genre is an abstract thematic concept. We introduce the Gutenberg Dataset for Genre Identification. Additionally, we present a study on how current deep learning models compare to traditional methods for this task. The results are presented as a baseline along with findings on how using an ensemble of chapters can significantly improve results in deep learning methods. The motivation behind the ensemble of chapters method is discussed as the compositionality of subtexts which make up a larger work and contribute to the overall genre.

1 Introduction

Literary computing poses unique challenges for Natural Language Processing (NLP) and Information Retrieval (IR). Literature itself is not structured like the usual expository corpora commonly made up of Wikipedia articles or short online reviews of products and services. Rather than communicating succinct and directed information, literature is artistic and conveys complicated themes over the course of very long narratives. Given our research which shows these documents on average contain well over 200,000 words, compared to the 100-1000 words in traditional document modeling datasets, it is apparent that new techniques should be considered to effectively model documents with a much higher fidelity.

This work offers a modern take on the classification problem of Genre Identification (Kessler et al., 1997). The goal of this problem is to assign the correct literary genre, such as *Adventure stories*, *Love stories*, etc., to a piece of literature using the title and body of the work. A solution to this problem would be beneficial to all organizations which supply literary services, such as libraries, online bookstores and search engines. Specifically, approaches to solve the Genre Identification problem would provide decision making aids to library scientists, but could also help with recommendation systems and information retrieval. Additionally, this work builds further insight into the study of literature, themes and genre. The problem of Genre Identification also works toward a greater topic of full length literature understanding and comprehension with complex and evolving themes throughout the course of an entire narrative.

Given the complicated and expansive nature of these documents, traditional approaches such as bag-of-word and bag-of-n-gram models are naturally unable to capture the enduring information which can persist across paragraphs and chapters. The themes which contribute to the assignment of genre to a book are not based purely on the frequencies of words used, but the abstract concepts presented over the course of the entire work. For example, it is not uncommon for Romance novels to dive deep into dark stories of murder and deceit, while still maintaining the overall theme of Romance. With this in mind, neural modeling approaches which learn fixed-length semantic representations of text are considered here, along with traditional bag-of-word approaches, as solutions when working on pieces of literature.

Most Neural Language Models which have been proposed are designed to capture text in a sequential fashion. These models are aimed at encoding the meanings of words or sentences based on previous items in a sequence. These works have begun to explore the application of successful architectures, such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Long Short-Term Memory Networks (LSTMs) or combination thereof to tasks such as sentiment classification and topic modeling (Kim, 2014; Tang et al., 2015).

However, when working with extremely long pieces of text, many of these Neural Networks quickly become impractical. Some of the models, such as the CNN, grow increasingly larger as the inputs grow, leading to unfeasibly large network implementations. The sequential models, such as the RNNs and LSTMs, are sensitive to extremely long sequences as gradients are likely to vanish during training and they require more and more time to train as the input sequence grows, eventually becoming too slow to train. Due to these limitations, it is evident that neural language models which are capable of processing an entire work of literature end to end would require very powerful systems and specialized machinery. In an effort to apply these neural network solutions to literature datasets, this work researches the effect of modeling smaller portions of text which are extracted from the greater work. These subtexts are used to train models, and results are presented which show the compositionality of the literature; that is, how the individual subtexts, such as chapters, contribute to the whole. The following questions on compositionality are addressed through this research: how do themes evolve over the course of a book and how do these evolving themes contribute to the overall genre of a book?

2 Related Work

This research builds on current deep learning approaches to text classification and sentiment analysis. CNN-Kim has been generally credited as one of the first neural network implementations for text classification (Kim, 2014). This model incorporated a simple CNN whose convolutions would capture neighborhoods of word embeddings. It was shown that on small text classification tasks, CNN-Kim could achieve competitive results when compared to other state-of-the-art models. This work laid a foundation for neural text classification, and while it focused on classifying sentences with an average length of 10-20 words, many researchers have built on this to work with longer documents (Glorot et al., 2011; Xiao and Cho, 2016; Zhang et al., 2015).

Since this time, there have been many more complex network architectures proposed for document modeling and classification. One such work proposes a novel architecture which models documents for sentiment classification using a hierarchy of LSTMs and bidirectional Gated Neural Networks (GNNs) (Tang et al., 2015). This hierarchy was designed to capture the natural structure of language where word representations are composed into sentence representations, which are in turn used to compose a document representation. This model produced competitive results on short text classification datasets with an average word count of 150. Other hierarchical approaches have been proposed that operate on a similar structure. Another proposed technique uses an autoencoder capable of capturing an unsupervised document representation with hierarchies of LSTMs (Li et al., 2015). The training goal is to reconstruct the text from the document representation using a hierarchical decoder. Two separate works have proposed alternative techniques to learning entire document representations in a hierarchical unsupervised fashion (Le and Mikolov, 2014; Kiros et al., 2015). All of these works, while exploring very different techniques, work on short documents where a document is composed of a list of sentences.

It has been shown that many NLP architectures can be significantly improved with the introduction of one or more Attention Mechanisms (Bahdanau et al., 2014). These Attention Mechanisms work to determine how informative individual elements of text are in regards to the task being trained over. These mechanisms employ a technique that learns a set of weights used to capture relevant meaning in a sequence through a weighted average. Using multiple levels of this mechanism, a Hierarchical Attention Network has been proposed for the task of document classification (Yang et al., 2016). This technique showed sweeping improvements over all other models on the same common text classification datasets.

In parallel to the research of deep learning models capable of performing complex text classification, significant efforts have been made towards finding optimal autoencoding solutions which are capable of compressing and representing variable-length text into fixed-length semantic vectors. The most notable of these is the word2vec embeddings which use a neural autoencoder in the skip-gram fashion to build fixed-length vectors which capture the contextual meaning of individual words (Mikolov et al., 2013b). Following this, Le and Mikolov (2014) and Kiros et al. (2015) both proposed the concept of embedding entire paragraphs and documents into fixed length vectors. While there has been reasonable discussion on the effectiveness of these embeddings, it is an important aspect to consider when building deep learning approaches to NLP.

Along with the extensive work that has been produced in applying deep learning techniques to text classification problems, there is also a wealth of solutions which apply traditional machine learning techniques and NLP representations such as bag of words (BOW). Liu et al. (2009) provide an in-depth study into models and weighted term representations to be leveraged when working with imbalanced text classification datasets. Their work compared BOW, term frequency - inverse document frequency (tf-idf) and custom weighting schemes while employing naive Bayes and support vector machines (SVMs) for their classification tasks. Other works have studied the impact that models such as random forests and k-nearest neighbors (kNN) have on text classification (Xu et al., 2012). While deep learning approaches have stolen the spotlight, these techniques are still highly relevant and are strong contenders for solving the Genre Identification problem.

3 Problem Definition and Hypothesis

This paper takes an approach where a literary genre is considered to be a writing style family where texts that contain similar themes are grouped together. The idea of genre used here is the same as the poetic genre defined by Miller and Greenberg (1981): "The term GENRE refers to a mode of writing that follows certain literary rules or conventions that have come down to the poet through custom and use... just as a word has connotations, a particular genre has a wealth of associations the poet may use." With this in mind, a classification task is envisioned that will learn these literary rules and conventions in a supervised environment.

The Genre Identification problem can formally be defined as: Given a book $d \in D$, referred to as a document from here out, with word length l_w where l_w has an order of magnitude of 4 or higher, determine the genre label $g \in G$ which represents the overall theme and categorization of the document. D is the set of documents in a corpus and G is the set of unique genre labels which the books may belong to. Both D and G are supplied by the literary dataset proposed in Section 3.2. In this work only one genre label is assigned to each document and $|G| = 6$. Given the extreme lengths and literary composition of these documents, the hierarchical nature which comprises them should be noted. Each document d is composed of l_c^d chapters or sections, and each chapter c in d is composed of $l_w^{d,c}$ terms. This notation will be used to reflect the hierarchical nature of the documents.

3.1 Text Composition Hypothesis

This work hypothesizes that genre, as an overarching theme of a piece of literature, is influenced over the course of the novel by individual subtexts, such as paragraphs and chapters. This hypothesis, if true, would mean that models which are used to assign a genre to a document could leverage an ensemble of subtexts or take a hierarchical approach in order to better classify the document. The experiments and evaluations presented here will evaluate how different subtexts of these long documents can be leveraged to improve the performance of the presented models.

3.2 Gutenberg Dataset

The Project Gutenberg is an extensive web catalog containing over 56,000 e-books. Along with providing the text for all of these books, Project Gutenberg also reports a detailed index for each book which contains the title, author, publication date and Library of Congress Subject Headers (LCSHs). All of

Genre	Count	Length	Mean Chapter Count	Mean Chapter Length
Science Fiction	$1,186$	$165,874$	8.2	$21,309.2$
Adventure stories	595	$459,556$	24.5	$22,288.5$
Love stories	508	$480,265$	24.2	$25,684.9$
Detective and mystery stories	485	$475,883$	25.1	$25,301.2$
Historical fiction	410	$558,343$	27.8	$26,046.5$
Western stories	393	$463,569$	23.1	$26,013.3$
Total	$3,577$	$379,101.5$	19.4	$23,694.6$

Table 1: Gutenberg Dataset Statistics

these e-books, along with the related index file, is publicly available on the Project Gutenberg website[1]. This work proposes the Gutenberg Dataset, a subset of the full Project Gutenberg catalog, which includes works with the following LCSHs: *Science fiction*, *Adventure stories*, *Historical fiction*, *Love stories*, *Detective and mystery stories* and *Western stories*. These 6 subject headers are the genres which we attempt to classify under the Genre Identification problem. Details of the Gutenberg Dataset can be found in Table 1. Scripts to download and assemble the Gutenberg Dataset as defined here can be found in the *ai-lit* project repository[2]. All of the authors' work is also available at this repository. 30% of the documents were held out in a test set which was used to evaluate performance.

4 Approaches and Architectures

Experiments are performed using both deep learning and standard text classification models. Additionally, multiple strategies are evaluated for representing the data during training in order to compensate for the extreme lengths of the documents. The objective is to establish a baseline evaluation to the Genre Identification task as well as to research how individual subtexts, such as chapters, contribute to the overall genre of a book.

4.1 Data Preprocessing

A minimal amount of preprocessing was performed on the dataset in order to prepare it for experimentation. First, the vocabulary of the entire dataset was extracted to build a distribution of all the unique terms found in the corpus. This vocabulary was captured and used to index the terms found in each book. In order to perform the indexing, the 5,000 most common terms from the vocabulary, along with the extra term $\langle$*unk*$\rangle$, used to indicate words which are outside of the 5,000 most common, were assembled to form the term index. This term index is used to convert each document, initially represented as a list of terms, into a list of integers which index words found in the term index. It is important to note that no additional text processing was performed on the data. Traditional NLP solutions continue to remove stop words and translate each word into its linguistic stem. However, in pursuit of a complete end-to-end deep learning architecture that does not require complicated preprocessing, these steps are omitted, which will force each model to learn the significance of stop word patterns and grammatical modifiers such as tense and plurality.

The neural network architectures proposed in this work utilize the concept of word-embeddings which represents each term found in a dataset as a N-dimensional vector. These vectors capture semantic and relational meaning of terms in regards to the goal on which they are trained (Mikolov et al., 2013b). These architectures can learn their own embeddings during training time, in which each vector is learned and optimized under the constraint of genre classification (Kim, 2014). This embedding layer works like a mapping where each term index corresponds to a meaningful vector, similar to the popular *word2vec* embeddings (Mikolov et al., 2013a).

[1]`https://www.gutenberg.org`
[2]`https://github.com/joeworsh/ai-lit`

4.2 Input Methodology

Due to the length of the records in the dataset, it is very prohibitive to design neural networks which can operate on an entire record all at once. Where traditional methods, such as bag of words, will sacrifice the structure and order of terms in a record in order to reduce dimensionality, here we propose alternative methods which allows the neural network models to capture the order and structure while still maintaining reasonable model sizes. These input methodologies follow the pattern proposed by Kolcz et al. (2001) in which smaller segments of a larger work are used for classification. Input methods which are utilized for both deep learning and traditional approaches are described below.

First 5,000: The first 5,000 input methodology both trains and evaluates on only the first 5,000 words in each record. In this case, the remainder of each book is discarded. There are no records in the dataset which are shorter than 5,000 terms, thus no padding is needed with this methodology.

Last 5,000: The last 5,000 input methodology, like the first 5,000, trains and evaluates on only the last 5,000 terms found in the dataset. Again, there is no need for padding as every document is longer than 5,000 terms.

Random 5,000: The random 5,000 method will randomly extract a 5,000 term window from a record on every training batch and evaluation batch. This method effectively expands the dataset as each epoch will see a different subtext for each book. While there is no guarantee that a book will be entirely processed, this random sampling process significantly grows the content which is trained and evaluated.

All Chapters: This input methodology applies a simple pattern matching algorithm to each book in order to split the books up into chapters. In this case, entire books, both in the training set and testing set, will be processed. However, chapter lengths vary significantly from 158 terms up to 169,588 terms, and we must select a fixed length for each (given the nature of the applied models). For this method, 2,500 terms are kept from each chapter. For the chapters which are shorter than this length, a padding token, $\langle pad \rangle$, is used to bring it up to the appropriate length. When using this method, results are computed across the individual chapters, and across full books by applying a voting scheme across all the chapters in each book.

Bag-of-words: BOW is a simple vector representation of an entire document where each dimension is representative of a unique word in the corpus. Thus, the input dimensionality is equivalent to the number of terms which are maintained when building the BOW representation. This method is a very simple and useful technique for representing documents as the frequency of terms which it is made of, while sacrificing the order of the words. In these experiments, all BOW representations are normalized so that the length of a document is not favored by any learning model.

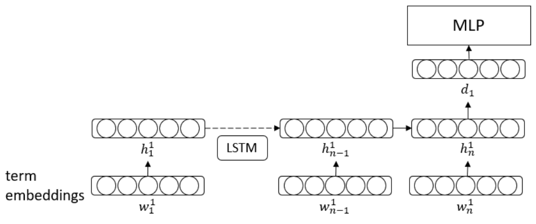
The models which are employed herein require both flat representations of the data and hierarchical representations. Hierarchical inputs will be labeled with an "*h2*" indicator to show that it is broken up into two hierarchical levels. In this case, each record, either a chapter or 5,000 term slice, is further parsed into paragraphs. Then a fixed size representation of 50 paragraphs each made up of 50 terms are captured in order to structure the records in the dataset. Thus, for all $d \in D$, $l_c^d = 50$ and $l_w^{d,c} = 50$. These numbers were selected in an effort to reduce the amount of padding included in the hierarchical models.

4.3 Models

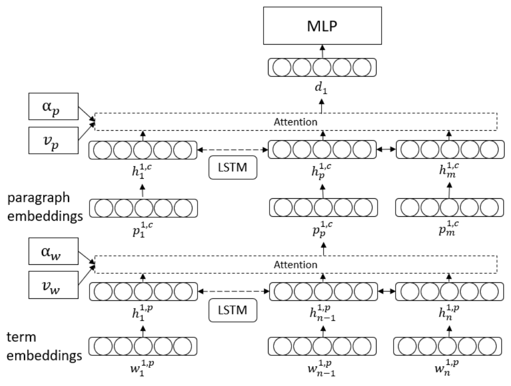
The models which are applied to the Genre Identification problem are detailed in this section. This work presents results with the CNN-Kim architecture (Kim, 2014), a flat LSTM model (Hochreiter and Schmidhuber, 1997), the Hierarchical Attention Network (HAN) (Yang et al., 2016) and a host of traditional machine learning algorithms. The authors also experimented with LSTM models for classification (Hochreiter and Schmidhuber, 1997), however due to the size of the data, they were prohibitively slow to train. Future research is needed to apply more optimized and hierarchical LSTMs to this data given the length of each text.

4.3.1 CNN-Kim

The deep learning model applied to the Genre Identification problem is the convolutional neural network (CNN) proposed by Kim (2014). This model begins with an embedding layer that learns a dense k-



(a) LSTM Architecture



(b) HAN Architecture

dimensional vector representation for each word during the training process. These experiments set $k = 100$ dimensions. Then, these word embeddings are processed in a convolutional layer by a set of filters $w \in W$ where $w \in \mathbb{R}^{hk}$. Here h is the size of the neighborhood around each word which is processed in the convolutional filter. This model employs four different h values, [3, 4, 5, 6], and 128 filters per filter shape. Following the convolutional layers, a max pooling layer is employed in order reduce the dimensionality of the previous layer and select the most informative dimensions. Finally, a fully connected layer, referred to as a multi-layer perceptron (MLP), is used to produce the network outputs. During training a dropout probability of 0.5 is applied following the convolutional layer as regularizer.

The problem is a multi-class classification task with high dependencies across very long sequences of terms. As such, this model will produce a softmax distribution of the confidence that a record belongs to each of the available classes. The softmax unit is defined as:

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \tag{1}$$

for network output vector z and unit j. In order to optimize each model for solving the classification problem, the cross-entropy between the softmax distribution p and the true genre distribution q of document x is minimized:

$$H(p,q) = -\sum_{x} p(x) \log q(x). \tag{2}$$

After the the loss of the CNN is computed, the Adam optimizer is used to train all of the network weights (Kingma and Ba, 2014) in order to minimize the loss with a learning rate of 10^{-4}.

4.3.2 LSTM

The next model applied to the Genre Identification problem is a text based Long Short-term Memory (LSTM) network. This model follows the standard LSTM architecture with a single layer where a document is represented as a sequence of term vectors (Hochreiter and Schmidhuber, 1997). Preceding the LSTM, the term index is embedded into a vector space through the embedding layer. As each term-embedded vector t is passed into the network, an LSTM cell c_t produces output h_t. The architecture of this model can be found in Figure 1a. While multiple values for the term and document embeddings were experimented with, the highest performing values were 100 and 500 respectively. The output of the LSTM is the hidden unit which represents the network output at timestep t. At each LSTM cell a dropout probability of 0.5 is applied as a regularizer.

The final LSTM hidden vector of a document sequence of length d, h_d, is used as the input to a softmax layer for performing classification over the set of possible genres and the loss is the same log loss. This loss value is minimized using the RMSProp optimzier (Tieleman and Hinton, 2012) with a learning rate of 10^{-3}.

4.4 HAN

The hierarchical nature of the HAN, plus the addition of the popular attention mechanisms make HAN a good fit for the Genre Identification problem. The encoding layers of the HAN will be made up of LSTM layers designed to capture the forward context of the text elements while focusing primarily on the element encoded at time step t (Li et al., 2015; Yang et al., 2016). The HAN architecture is detailed in Figure 1b. The attention layers of the HAN are designed to learn which encoded elements from the previous layer contribute to the classification task and which elements do not. This layer is made up of three components. The first is a simple MLP trained to build annotations over each element. The annotations learn to represent the contribution made by the element to the classification task. The measurement of the importance of the element is then computed using a softmax function over all elements at this layer. Finally a weighted sum of the encoding layer using the importance measurement is produced as a representation for the element at the next hierarchical level (Bahdanau et al., 2014). The attention mechanism at each layer is represented as:

$$u_t = \tanh(Wh_t + b) \tag{3}$$

$$\alpha_t = \frac{\exp(u_t^T u)}{\sum_t \exp(u_t^T u)} \tag{4}$$

$$s_t = \sum_t \alpha_t h_t \tag{5}$$

where h_t is the hidden state of cell t. W and b are the trainable components of the attention MLP and s_t is the weighted average output of the attention mechanism. The hierarchical nature of this model leads to three different embedding sizes, term, paragraph and document. After experimentation, the highest performing values were found to be 50, 100 and 150 respectively.

Again the architecture is finished off with a softmax layer and the loss is computed. This model employs a dropout probability of 0.5 at both the term layer and the sentence layer. Again, the Adam optimizer is employed (Kingma and Ba, 2014) with a learning rate of 10^{-3}.

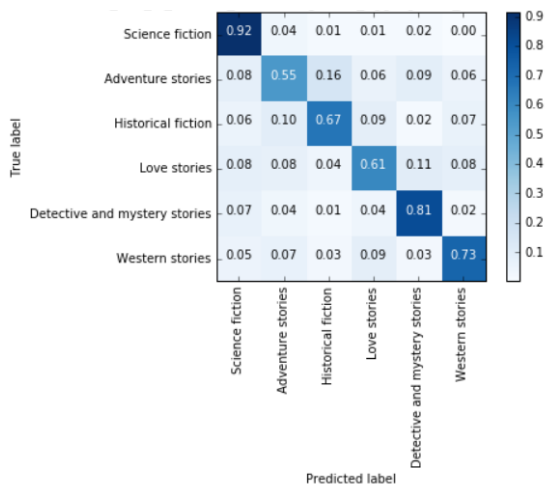
4.4.1 Comparison Models

Along with the detailed deep learning models, we employed traditional machine learning models for classification in order to establish a performance baseline for the Genre Identification task. Naive Bayes (Rish, 2001), k-Nearest Neighbor (*kNN*) (Cover and Hart, 1967), Random Forest (Breiman, 2001) and XGBoost (Chen and Guestrin, 2016) are all evaluated with the BOW input method. *kNN* is different than the other models in that it simply employs a distance metric in order to identify which other documents in the dataset a new record is found to be close to. The set of nearest neighbors then vote on what the label of the new record should be. In these experiments, *k* is set to 1, 5 and 10 and the standard Euclidean distance is used (Huang, 2008).

The Random Forest model is made up of an asynchronous collection of small decision trees which are trained on random selections of features from the dataset. In this case the features are the unique tokens within the BOW input method. It has been shown that Random Forest is an effective model for classification and regression and is not prone to overfitting (Breiman, 2001). XGBoost is a highly optimized, award winning Gradient Boosting solution which is made up of a boosted set of sequential trees learned from the gradients of some differentiable loss function (Chen and Guestrin, 2016). This model often produces state-of-the-art results and is trained on the BOW form of the Gutenberg Dataset in this work.

5 Results

After training, each model was used to assign genres to the records reserved in the test set. Accuracy and the F1-Measure are used as a measure of a model's performance. The F1-Measure is defined in terms of precision and recall (Powers, 2011). The results of each experiment can be found in Table 2. Our results show that each of the models were able to generalize and capture elements that pertain



True label \ Predicted label	Science fiction	Adventure stories	Historical fiction	Love stories	Detective and mystery stories	Western stories
Science fiction	0.92	0.04	0.01	0.01	0.02	0.00
Adventure stories	0.08	0.55	0.16	0.06	0.09	0.06
Historical fiction	0.06	0.10	0.67	0.09	0.02	0.07
Love stories	0.08	0.08	0.04	0.61	0.11	0.08
Detective and mystery stories	0.07	0.04	0.01	0.04	0.81	0.02
Western stories	0.05	0.07	0.03	0.09	0.03	0.73

Figure 2: All Chapters CNN Confusion Matrix

Input	Model	Acc.	F1
First 5,000	CNN	0.62	0.55
Last 5,000	CNN	0.56	0.42
Random 100	LSTM	0.42	0.35
Random 5,000	LSTM	0.34	0.20
Random 5,000	CNN	0.65	0.61
All Chapters	CNN	0.75	0.71
All Chap. (h2)	HAN	0.52	0.53
BOW	Naive Bayes	0.63	0.59
BOW	kNN k=1	0.64	0.58
BOW	kNN k=5	0.61	0.55
BOW	kNN k=10	0.61	0.55
BOW	Rand Forest	0.82	0.79
BOW	XGBoost	**0.84**	**0.81**

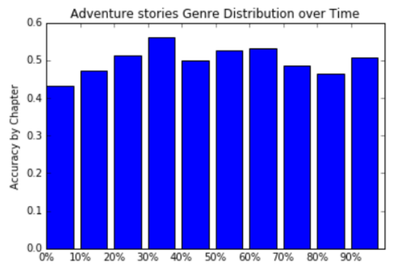
Table 2: Genre Identification Results

to genre across the different works in literature. However each model had weaknesses when working with this dataset. CNN-Kim had the most reliable performance of the deep learning models, scoring a total of 75% accuracy on the "All Chapters" input. Surprisingly, the HAN model had a difficult time converging on this dataset, frequently being fooled by the more represented classes such as *Adventure Stories* and *Science Fiction*. The LSTM, the lowest scoring neural network architecture, collapsed and was unable to learn relevant features over the long sequences. Further experiments showed that an LSTM trained on random 100 term slices from each document actually outperformed the 5,000 term model by 8% accuracy. XGBoost outperformed all models to have the highest accuracy at 84% and F1-Measure at 81%. This shows that while the deep learning models have garnered a lot of attention, tree-based methods still come out on top, and took minutes, instead of days, to train.

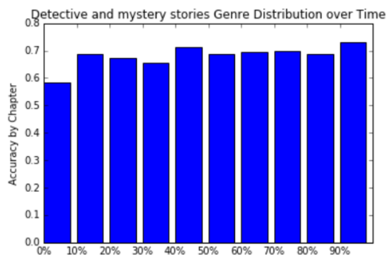
These results fall short of similar experiments performed by Kim (2014) using CNNs, however, the difference in data can be credited for this deviation. On a simple question classification task CNN-Kim received an accuracy rating between 91.2% and 93.6%. Lai et al. (2015) report results of 47.47% for another multi-label classification task using a CNN. Similar results have also been published using RNNs and LSTMs for sentiment classification tasks with evaluations ranging between 40% and 71% (Tang et al., 2015; Yang et al., 2016). Additionally, Liu et al. (2009) have shown the common distribution of text classification results given traditional BOW and tf-idf methods. The recorded Naive Bayes F1-Measures from this research range between 45% and 80% across many different experiments and datasets (Liu et al., 2009).

Additionally, this research has shown that across the corpus, there was a 6% gain in accuracy when training and evaluating on the first 5,000 words of each book over training and evaluating on the last 5,000. However, there was an additional 2% increase in accuracy when training and evaluating on the random 5,000 word subtexts extracted from each book. This gain in accuracy provided the insight that training our model on every chapter, and then setting up a voting algorithm where all the chapters of a book get to vote on the genre of the book could prove beneficial. Intuitively, each subtext is unique and can be leveraged in a way that expands the dataset during training allowing the models to train on new subtexts each epoch. As such, the All Chapters CNN-Kim experiment received a large 10% gain in accuracy over the best performing deep learning model with an accuracy of 75% and an F1-Measure of 71%. Figure 2 shows the confusion matrix over all classes in the test set.

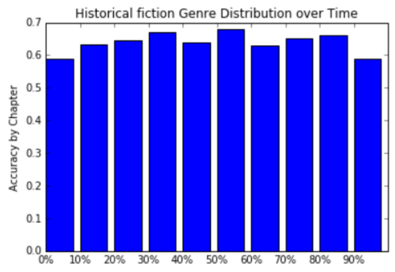
Following the voting scheme method employed in the "All Chapters" technique, which yielded a high increase in performance, we digitized each chapter within their respective books into one of ten bins based on the chapter index relative to the number of chapters. That is, chapter $c \in C$ was placed in bin $b = \lfloor \frac{c}{|C|} * 10 \rfloor$. Following this, the accuracy for each bin was computed, relative to the actual genre label, to show the accuracy over the course of each book in the genre. These accuracy over time compositions



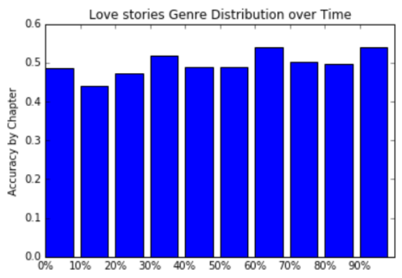
(a) Adventure stories



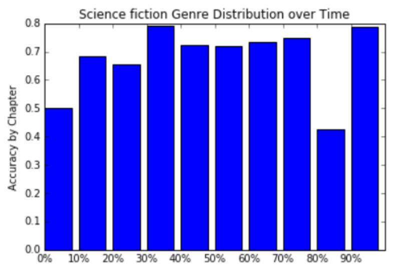
(b) Detective and mystery stories



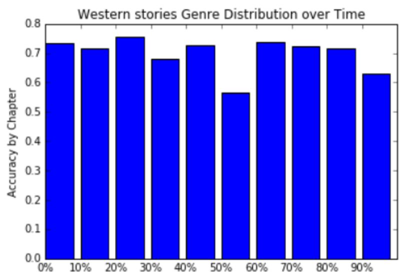
(c) Historical fiction



(d) Love stories



(e) Science fiction



(f) Western stories

Figure 3: Genre composition over the course of the narrative - All Chapters CNN-Kim

for the CNN-Kim experiment are shown in Figure 3. While it is interesting to note the slight pattern variations that have emerged across the books in each genre, such as"Adventure story" novels being more difficult to classify in the first third, it is important to show that accuracy is fairly consistent across all chapters. There is no strong emerging patterns which indicate that some genres would consistently look like others at certain points in the narrative. However, this composition also provides certainty to the voting method as all chapters can be considered relatively reliable when assembling a genre classifier.

6 Future Work

Following the proposal of the Genre Identification problem and the research presented within, which shows the benefits of taking a compositional approach when working with text of this magnitude, future research will focus on building and refining hierarchical models for gains in performance. Applying a hierarchical model, like HAN, on the chapter encodings to learn classification from chapters could yield improved results over the voting scheme employed here, as a model will have the capability to learn patterns over the courses of each chapter. Given the limitations of traditional LSTMs in this context, there is a wealth of future research opportunities to apply specialized models to this domain.

Additionally, we propose future research into the compositionality of literature in the forms of comparing chapter and book similarity across the genres. Future research will study how similar chapters are to each other, and how books of the same genre may vary given additional parameters such as the title, region of origin and publication date. A study on how the title of a book contributes to the assigned genre is the next step in working with the Gutenberg dataset. There is still a lot to be learned when analyzing literature.

7 Conclusion

This work has presented the Genre Identification problem, which is a very long text classification task that requires both syntactic and thematic analysis in order to assign a literary genre to a book from a corpus. Not only is genre a challenge because it is comprised of long running themes which can span paragraphs and chapters, but these documents are significantly longer than the usual text classification datasets.

Along with the Genre Identification problem, different machine learning approaches are presented and evaluated as solutions for assigning genre. Convolutional Neural Networks, LSTMs, HAN, Naive

Bayes, k-Nearest Neighbors, Random Forests and XGBoost are considered along with different data representation schemes to help manage the extreme lengths of the documents. We showed that results were improved when training on random slices of 5,000 words from each document. Following this intuition, an ensemble method was assembled in which each chapter can vote towards the total genre of the book. This ensemble yielded a 10% increase in accuracy. Finally, the genre composition of chapters belonging to each genre has been evaluated to show how chapters contribute to the genre as a whole. There will continue to remain a significant amount of work to be done in researching new deep learning models and gaining more insight into genre and literature.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Leo Breiman. 2001. Random forests. *Machine learning*, 45(1):5–32.

Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA. ACM.

Thomas Cover and Peter Hart. 1967. Nearest neighbor pattern classification. *IEEE transactions on information theory*, 13(1):21–27.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 513–520.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November.

Anna Huang. 2008. Similarity measures for text document clustering. In *Proceedings of the sixth new zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand*, pages 49–56.

Brett Kessler, Geoffrey Numberg, and Hinrich Schütze. 1997. Automatic detection of text genre. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, pages 32–38. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302.

Aleksander Kolcz, Vidya Prabakarmurthi, and Jugal Kalita. 2001. Summarization as feature selection for text categorization. In *Proceedings of the tenth international conference on Information and knowledge management*, pages 365–370. ACM.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *AAAI*, volume 333, pages 2267–2273.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1188–1196.

Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. 2015. A hierarchical neural autoencoder for paragraphs and documents. *arXiv preprint arXiv:1506.01057*.

Ying Liu, Han Tong Loh, and Aixin Sun. 2009. Imbalanced text classification: A term weighting approach. *Expert systems with Applications*, 36(1):690–701.

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

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

Ruth Miller and Robert A Greenberg. 1981. Genre. In *Poetry*, pages 158–202. Springer.

David Martin Powers. 2011. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation.

Irina Rish. 2001. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46. IBM.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *EMNLP*, pages 1422–1432.

Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31.

Yijun Xiao and Kyunghyun Cho. 2016. Efficient character-level document classification by combining convolution and recurrent layers. *arXiv preprint arXiv:1602.00367*.

Baoxun Xu, Xiufeng Guo, Yunming Ye, and Jiefeng Cheng. 2012. An improved random forest classifier for text categorization. *JCP*, 7(12):2913–2920.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J Smola, and Eduard H Hovy. 2016. Hierarchical attention networks for document classification. In *HLT-NAACL*, pages 1480–1489.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.