Jointly Learning Author and Annotated Character N-gram Embeddings: A Case Study in Literary Text

Suraj Maharjan[⋆] Deepthi Mave[⋆] Prasha Shrestha[⋆] Manuel Montes-y-Gómez[†]
Fabio A. González[‡] Thamar Solorio[⋆]
[⋆]Department of Computer Science, University of Houston
[†]Instituto Nacional de Astrofisica Optica y Electronica, Puebla, Mexico
[‡]Systems and Computer Engineering Department, Universidad Nacional de Colombia
{smaharjan2, dmave, pshrestha3, tsolorio}@uh.edu
mmontesg@ccc.inoep.mx, fagonzalezo@unal.edu.co

Abstract

An author's way of presenting a story through his/her writing style has a great impact on whether the story will be liked by readers or not. In this paper, we learn representations for authors of literary texts together with representations for character n-grams annotated with their functional roles. We train a neural character n-gram based language model using an external corpus of literary texts and transfer learned representations for use in downstream tasks. We show that augmenting the knowledge from external works of authors produces results competitive with other style-based methods for book likability prediction, genre classification, and authorship attribution.

1 Introduction

Literary texts have been computationally modelled by extracting stylistic traits such as readability and writing density, flow of emotions, and even by cover images of books (Maharjan et al., 2018b,a, 2017; Ashok et al., 2013). However, modelling of authors through their work has not been explored until now. An author's style of presenting stories has a great influence on whether a book will be liked by readers or not.

We can find evidence of the effect that an author's style has on readers in book reviews and through readers' comments left on Goodreads[1] shown in Table 1. The readers talk about the impact of the author's writing style on their reading experience. In the first two examples, it left a positive impact on the readers, while in the last it had a negative impact. These examples provide further evidence for the need for modeling authors' writing style for the task of likability prediction of books.

This author's writing style is just perfect in every way, you will feel everything one should experience when you read a genre such as this.
The author's writing style is straightforward which made it easy to understand.
I think that the writing is very uneven. Overwhelmingly episodic, not terribly consistent, and largely as dimensionless as the characters.

Table 1: Readers comments showing the importance of authors' writing style

In this paper we propose a new approach to capture style in text by jointly learning author specific embeddings and character based n-gram embeddings. The idea of using author embeddings is motivated by reader comments as discussed above. The use of character n-gram embeddings comes from previous work on authorship attribution (AA) that has shown character n-grams to have strong prediction value for the task (Kešelj et al., 2003; Peng et al., 2003; Koppel et al., 2009; Stamatatos, 2009; Sapkota et al., 2015). Rather than using plain character-based n-grams, we first annotate them with their functional roles (*prefix*, *suffix*, and *whole-word*). This is necessary since, for example, *off* is semantically distinct from when it is used as a whole word and when it is used as a suffix (e.g. *trade-off*) or when it is used as a prefix (e.g. *offend*).

After obtaining representations for authors and annotated character n-grams using an external corpus of literary texts, we transfer them to tasks where author information is useful, namely book likability prediction and authorship attribution. Moreover, we provide quantitative and qualitative analyses of the author and annotated character n-gram embeddings.

[1]https://www.goodreads.com

Proceedings of Recent Advances in Natural Language Processing, pages 684–692,
Varna, Bulgaria, Sep 2–4, 2019.
https://doi.org/10.26615/978-954-452-056-4_080

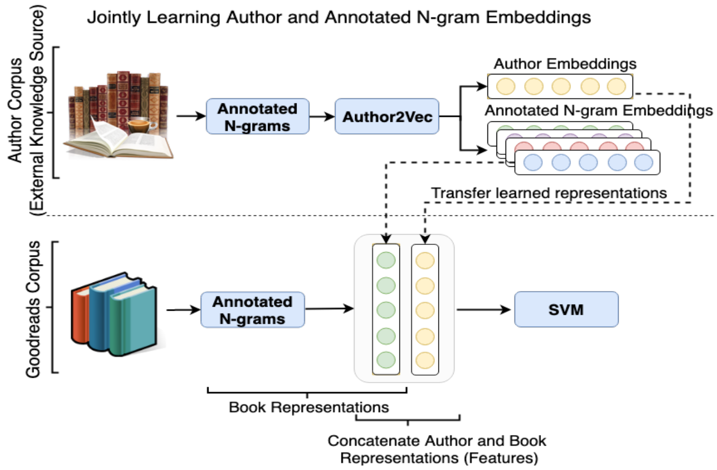

Figure 1: Learning and using author and annotated character n-gram embeddings

2 Methodology

Figure 1 shows the overall workflow of our proposed method. As shown in the figure, there are two phases. First, we jointly learn embeddings for authors and annotated character n-grams using an external corpus of books written by authors in the Goodreads corpus prepared by Maharjan et al. (2017). We refer to this corpus as Author Corpus. Next, we transfer this knowledge represented as the author and annotated character n-gram embeddings to build book representations. We describe these phases in detail below.

Phase I: Learning from an external corpus

We collected a new external corpus of books from Project Gutenberg to learn author and annotated character n-gram embeddings[2]. It consists of at most five books from each author in the Goodreads corpus (§3.1).

Annotated Character n-grams: We annotate character n-grams according to their position and function in a word as *prefix*, *suffix*, or *whole-word*. We follow the definitions by Sapkota et al. (2015) to decide which n-grams constitute each of these three types. *Prefix* and *suffix* are character n-grams that cover the first and last n-characters, respectively, of a word that is at least $n + 1$ characters. A *whole-word* n-gram is word that is exactly n characters long. This annotation helps to distinguish a single lexical entity as many different semantic entities. For instance, an n-gram like *the* could either be used as a *prefix* (***the****refore*), as a *suffix* (*wrea****the***), a standalone word (***the***), or occur within a word (*wi****the****r*). Note that although we do not explicitly annotate n-grams occurring mid-word, all of the remaining unannotated n-grams will fall under this category. These annotations will ensure that separate embeddings are learned according to the morphological and functional information carried by the n-grams.

Similar to Sapkota et al. (2015), we choose n as 3. While generating character 3-grams, we try various step sizes for sliding our window. With a step size of one, adjacent 3-grams will have two characters in common, one character in common with a step size of two and none with a step size of three. We name them *Overlap*, *Partial*, and *Non-Overlap*, respectively, based on the overlapping of characters in adjacent n-grams. We explore author and annotated character n-gram embeddings under these three settings.

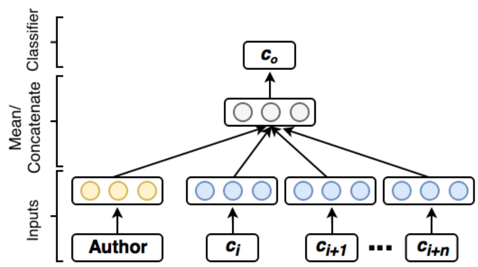

Figure 2: Author2Vec framework to learn author and character n-gram embeddings. $c_i, \ldots, c_{i+n}$ and $c_o = c_{i+n+1}$ are the input and output annotated character n-grams, respectively.

Author2Vec: Given a sequence of annotated character n-grams $c_i, \ldots, c_{i+n}$, and an author a, the objective of our Author2Vec model is to maximize

[2]The source code and data for this paper can be downloaded from https://github.com/sjmaharjan/author2vec.

the following conditional probability for the next-in-sequence output character n-gram c_o:

$$p(c_o|c_i,\ldots,c_{i+n},a) = \frac{exp(y_{c_o})}{\sum_{c\in V} exp(y_c)} \quad (1)$$

$$y_c = Wh(c_i,\ldots,c_{i+n};C,a;A) + b \quad (2)$$

where y_c is the unnormalized log probability for annotated character n-gram c in the vocabulary V, W and b are softmax parameters, and h is either the concatenation or mean function applied to character n-gram and author vectors from C and A, respectively. Similar to Le and Mikolov (2014), we call the concatenation method Distributed Memory Concatenation (DMC) and the mean method Distributed Memory Mean (DMM).

Phase II: Building book representations

We use the annotated character n-grams from Phase I to obtain the book's representations. We concatenate this with the book author's embedding and feed them as features to an SVM classifier. Similar to Maharjan et al. (2017), we consider the first 1k sentences from each book in the Goodreads corpus. We define the following three methods to obtain book representations:

Bag of annotated character n-grams (ACn): Similar to bag-of-word (BoW) approach, we generate the annotated character n-grams from books' content. We then represent each book by a sparse vector and weight each annotated character n-grams using their term frequency-inverse document frequency (TF-IDF) scores. The motivation behind using this representation is the success of stylistic analysis in the domain of books success prediction, author attribution, and author profiling.

Mean of Annotated character n-grams embeddings (Mean): Unlike the above method, here we use the annotated character n-gram embeddings to represent each book by a dense vector. We generate the annotated character n-grams from book content and look up their embeddings. A book is then represented by the mean of all annotated character n-gram embeddings generated from its content. Mathematically, the resulting book vector r is represented as $r = \frac{\sum_{i=1}^{N} emb(c_i)}{N}$, where N is the total number of annotated character n-grams for a book, and $emb(.)$ is the function that gets the embeddings for a given annotated character n-gram c.

Inverse Document Frequency (IDF) Weighted Average (Weighted): This method is similar to the above method of averaging the embeddings of annotated character n-grams, but weights each annotated n-grams embedding by their IDF scores before averaging. Mathematically, the resultant book vector r is represented as $r = \frac{\sum_{i=1}^{N} idf(c_i,B)*emb(c_i)}{N}$, where $idf(.)$ is the function that gets the IDF score for given annotated character n-gram c. The $idf(.)$ is learned from the training data and is defined as $idf(c,B) = \log \frac{|B|}{|\{d\in B:c\in d\}|}$, where B is the collection of books, and d is an instance of book.

3 Book Likability Prediction

Here we present results of using the author and character based n-gram embeddings for the task of predicting whether readers will like a book or not. We use likeability as a proxy to measure the success of a book. Narrowing down success as a measure of readers ratings is not ideal. But it gives us a practical starting approach to evaluate our models.

3.1 Dataset

We experiment with the publicly available book likability prediction dataset (Goodreads corpus) from Maharjan et al. (2017). They collected books from Project Gutenberg[3]. They then labeled the books into two categories: *Successful* and *Unsuccessful*, by using the average rating and the total number of reviews received by the books on Goodreads[4]. It consists of 1,003 books (654 *Successful* and 349 *Unsuccessful*) from 8 genres downloaded from Project Gutenberg: *Detective Mystery*, *Drama*, *Fiction*, *Historical Fiction*, *Love Stories*, *Poetry*, *Science Fiction*, and *Short Stories*.

3.2 Experimental Settings

We used the same stratified splits of 70:30 training to test as provided by Maharjan et al. (2017). We used the negative sampling (Mikolov et al., 2013) method to train 300-dimension embeddings for both authors and annotated character 3-grams. We filtered out 3-grams with frequency$<$2, set the window size to 5, configured the sample threshold to 1e-5 for randomly downsampling higher-frequency 3-grams, and trained for 100 epochs.

We predicted success separately (*Single task (ST)*) as well as simultaneously with genre (*Multi-task (MT)*). We used linear kernel SVM and tuned

[3] https://www.gutenberg.org/
[4] https://www.goodreads.com

Type	Overlap		Partial		Non-Overlap	
Features	**ST**	**MT**	**ST**	**MT**	**ST**	**MT**
Character 3-grams (Maharjan et al., 2017)	66.9	70.0	-	-	-	-
All Typed n-grams (Maharjan et al., 2017)	66.3	69.1	-	-	-	-
Annotated char-3gram(AC3)	66.8	69.8	**71.1**	67.8	68.9	70.5
Mean (DMM)	60.5	67.6	63.7	66.6	65.6	**68.3**
Mean (DMC)	62.8	70.0	63.5	**70.1**	65.1	67.7
Weighted (DMM)	56.5	65.6	65.6	**69.9**	66.7	69.0
Weighted (DMC)	65.3	66.4	64.8	**67.2**	60.0	63.5
AC3 + Author (DMM)	62.8	68.5	67.5	67.5	**70.6**	68.5
AC3 + Author (DMC)	69.3	68.7	67.7	68.3	**71.3**	70.0
Mean + Author (DMM)	62.6	70.3	68.2	66.6	**71.9**	66.8
Mean + Author (DMC)	69.0	69.2	68.3	67.0	**71.5**	71.1
Weighted + Author (DMM)	62.3	70.0	66.9	66.6	70.6	**70.7**
Weighted + Author (DMC)	71.1	**73.8***	69.8	70.1	71.7	70.5

Table 2: Weighted F1-scores (%) using book representations with and without author embeddings under three settings (*Overlap*, *Partial*, and *Non-Overlap*). *statistically significant at $p < 0.02$ (McNemar significant test with and without author embeddings).

the *C* hyper-parameter through a grid search over (1e{-4,...,4}), using three-fold cross validation on the training split.

3.3 Results

Table 2 shows the results for our methods under the *Overlap*, *Partial*, and *Non-Overlap* settings. Our first set of experiments tests book representation methods (*AC3*, *Mean*, and *Weighted*) without author embeddings. The sparse feature representation method *AC3* (71.1%) performs better than embedding aggregation methods, *Mean* (69.9%) and *Weighted* (70.1%), as the mean operation likely removes important information. Here, the *Partial* setting yields the best results.

Our next set of experiments combine book representations with author embeddings and this improves the results in most cases. We obtain the overall highest F1-score of 73.8% with *Weighted* setting under concatenation (DMC) method. This result is statistically significant ($p < 0.02$) over the same setup but without authors' embeddings. This result is also better than the results from the state-of-the-art methods by Maharjan et al. (2017): *Character 3-grams* (70.0%) and *All typed n-grams* (69.1%). We also see that the DMC method yields consistently better results than the DMM method.

Author embeddings and correctness: We group authors by the genre of their books (in case of multiple genres, we pick the one with the most books). We then obtain representations for successful and unsuccessful authors in that genre by averaging the author vectors for these two classes. Our intuition was that if the distance between these representations is small, there will be fewer correct predictions for that genre. We obtained a large negative Pearson correlation coefficient of -0.753 ($p < 0.03$) between distances and the number of incorrect predictions, supporting our intuition.

Annotated vs Plain Character n-grams: To validate the importance of annotating n-grams, we trained embeddings using unannotated n-grams and performed likability prediction using the best setup from before. This produced an F1-score of 69.9% ($<$ 73.8%) illustrating the usefulness of considering the functional behavior of character n-grams. The performance further decreased to 63.4% with the removal of author embeddings.

Authors as binary vectors: To further confirm the advantage of learning author embeddings from external data, we replace author embeddings with one-hot vectors indicating the book's author. Using the best setup in Table 2, this produced a score of 70.3% ($<$ 73.8%), strengthening our intuition that author embeddings capture style related information, which is relevant for likability prediction.

Author Embeddings and Genre: We experimentally verify that author embeddings capture genre-specific information by using them to perform genre classification. We used the best model, Author (DMC), to automatically infer author vectors for all books in the dataset and fed them to an SVM classifier. We obtained F1-scores of 64.6%, 66.8%, and 64.0% with the *Overlap*, *Partial*, and *Non-Overlap* settings respectively. These scores outperform a random baseline of 15.2%, showing that author embeddings are also capturing more general style traces related to the genre.

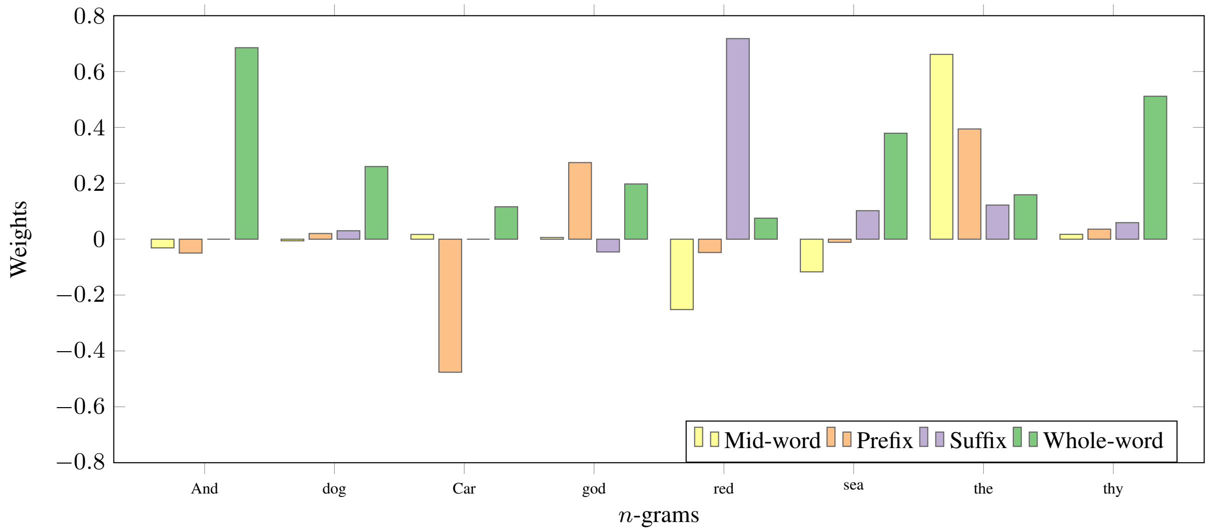

Figure 3: Feature importance assigned by SVM to different character n-grams for the likability prediction task.

4 Discriminative Annotated Character n-grams

Figure 3 shows some of the top positively and negatively weighted annotated n-grams by the classifier. We used the best performing *AC3* model from Table 2, *AC3* under *Partial* setting, to extract the weights for the annotated character n-grams. For each of the annotated n-grams, the figure also plots the weights for all four positional variants. The figure clearly shows that different forms of the same character n-gram have different contributions towards the likability prediction of books. This important piece of information would have been lost if we had treated these different forms of the n-grams as one. For instance, *sea* as a whole-word has a different meaning than when it is used as a prefix or a suffix. Accordingly, the classifier has also weighted them differently. The whole-word form of the n-gram *sea* is weighted higher than its other forms. This also holds for the case of *thy* and *dog*. During this analysis, we also found that quotation marks and male honorific titles were highly weighted by the classifier, similar to what Maharjan et al. (2017) found. This most likely points to the importance of dialogues and the preference of male characters in these books.

5 Analysis of Annotated Char n-grams

In Figure 4, we visualize the annotated character n-gram embeddings by projecting them using PCA. For some n-grams, the embeddings of different annotations are indeed distinct. For instance, the embeddings for *sub* (prefix, mid-word), *est* (whole-word, suffix), *ion* (suffix, whole-word), *mid* (prefix, suffix), and *the* (whole-word, suffix) lie far from each other. On the other hand, *ful* in suffix and mid-word form are close together, since the contexts for *ful* as a suffix (*beautiful*, *careful*) are similar to the contexts where it occurs mid-word (*beautifully*, *carefully*). In addition, we can also see a clear separation between *prefix* and *suffix* n-grams with the *mid-word* and *whole-word* n-grams occupying regions in between. The *suffix* and *prefix* n-grams mostly occupy the regions above and below the zero line respectively. This figure visually demonstrates that learning separate embeddings for n-grams with different functional roles is important to preserve their semantics.

Dataset	RG (65)	WordSim (353)	RW (2034)
Without Annotation	16.21	4.56	16.54
Annotated	30.75	12.27*	20.02**

Table 3: Results for word similarity task showing Spearman's rank correlation ($\rho \times 100$) with similarity scores assigned by human annotators $*p < 0.03$, $**p < 0.0001$

We also empirically show the advantage of these embeddings through the word similarity task using three standard datasets: *RG65* (Rubenstein and Goodenough, 1965), *WordSim353* (Agirre et al., 2009), and *RW* (Luong et al., 2013). Similar to FastText (Bojanowski et al., 2017), we represent a word as an average of the embeddings of its character n-grams. We create two representations

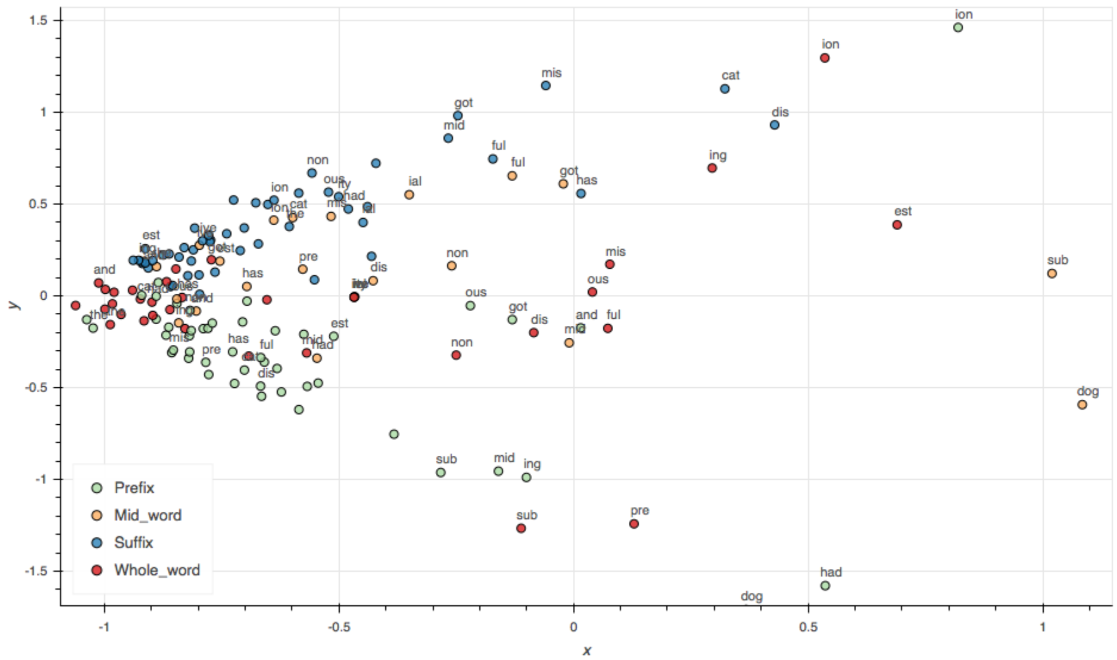

Figure 4: Projection of annotated char n-gram into 2D space using Principal Component Analysis (PCA)

for each word: one using plain character n-grams and another with annotated character n-grams. Table 3 shows the results for the word similarity task with these two different approaches. The Spearman's rank correlation coefficient between human annotations and word vectors composed of our annotated n-gram embeddings are higher than the same obtained from plain n-grams. The difference between the two proposed methods is statistically significant for WordSim353 ($p < 0.03$) and highly statistically significant for RW2034 ($p < 0.0001$). These results demonstrate that annotated character n-gram embeddings are also good at producing high-quality word representations and they might be capturing semantics at some level.

6 Authorship Attribution

Another task to test the effectiveness of our approach is authorship attribution (AA). To ensure that the model learns to discriminate authors and not the genre, we selected books from fiction genres for authors having at least five books that were not used to train our Author2Vec model. We have 12 such authors in our corpus.

We again used the first 1k sentences, used an SVM classifier in a stratified 5-fold cross-validation setup, and tuned the C hyper-parameter using grid search for each fold. Table 4 shows our results along with two baselines: word unigram and character 3-grams with tf-idf. Using only the book representations (Mean or Weighted), we obtained the highest mean accuracy of 86.67%. When we added in the inferred author embeddings (directly getting author embeddings would reveal the author, so we infer author embeddings similar to Le and Mikolov (2014) using only the book content without revealing the actual author of the book), the accuracy improved to 95% ($\sim$ 10% above *Char 3-gram*), showing that our approach not only works for likability prediction but also for AA.

7 Related Work

Sapkota et al. (2015) sub-grouped character n-grams according to grammatical classes, like affixes, lexical content, and stylistic classes, like beg-punct and mid-punct. With these sub-groups, they provided empirical evidence to support the importance of character n-grams features in the task of authorship attribution. Iacobacci et al. (2015) showed that learning separate word embeddings for polysemous words yields the state-of-the-art result in word similarity and relational similarity tasks. Separating the same tokens or n-grams helps to preserve their functional and morphological information which is important for all tasks. Learning embeddings for words (Mikolov et al., 2013), n-grams (Zhao et al., 2017), and documents (Le and Mikolov, 2014) and using them as input for various NLP tasks (Samih et al., 2016;

Methods	Overlap (%) $\mu \pm \sigma$	Partial (%) $\mu \pm \sigma$	Non-Overlap (%) $\mu \pm \sigma$
Word Unigrams	83.33 ± 5.27	-	-
Char 3-grams	85.00 ± 6.24	-	-
AC3	81.67 ± 6.24	83.33 ± 9.13	83.33 ± 10.54
Mean	85.00 ± 6.24	86.67 ± 8.50	83.33 ± 9.13
Weighted	81.67 ± 12.25	80.00 ± 8.50	83.33 ± 10.54
Mean + Inferred Author	83.33 ± 11.79	$\mathbf{95.00 \pm 4.08}$*	90.00 ± 6.24
Weighted + Inferred Author	83.33 ± 11.79	93.33 ± 6.24	90.00 ± 6.24

Table 4: Mean and standard deviation of accuracy for 5 fold cross validation AA experiments (*statistically significant at $p < 0.05$ from t-test with *Char* 3-*grams*)

Zhang et al., 2015; Kim, 2014) has shown improvement in performance. Again, Shrestha et al. (2017) showed that applying convolutional neural network (CNN) over character bigrams embeddings improves authorship attribution of short texts like tweets. They visually showed that character bigrams were capturing important stylistic markers to distinguish between bot-like authors and other normal authors. Song and Lee (2017) jointly learned the embedding for users (senders and receivers) and showed that these user embeddings capture the semantic relationship between users through an auto-foldering of emails task. Following these research findings, we also distinguish between the same character n-grams by annotating them with categories defined by Sapkota et al. (2015) and learn separate embeddings for each of them in addition to learning embeddings for authors.

Prior works in likability prediction of books have shown that style is an important aspect (Maharjan et al., 2017; van Cranenburgh and Bod, 2017; Ashok et al., 2013). They captured the style of successful and unsuccessful books using lexical, syntactic, readability, and writing density features, and deep learning methods with only first 1K sentences. Since style is evident even with first few fragments, they obtained competitive results using only first few fragments of books. Maharjan et al. (2017) even showed that around 200 sentences are enough to perform book success prediction with reasonable accuracy. Louis and Nenkova (2013) proposed features to capture different aspects of great writing (surprising, visual and emotional content) and used them in combination with genre-specific features to predict high quality writings in science articles. Iwana et al. (2016) extracted visual features from book covers for genre classification. Also, the potential of using a computer to plot the trajectory of emotion throughout the book and its correlation with success have been discussed (Vonnegut, 1981; Reagan et al., 2016). However, learning and using stylistically aware embeddings for authors in conjunction with other relevant stylistic features extracted from books for the problem has been overlooked. Our work fills this gap by adding author's general writing style learned using an external corpus of books.

8 Conclusions and Future Work

In this paper we explored a new dimension of modeling authors for literary texts by jointly learning annotated character n-gram embeddings and author embeddings using an external corpus. We showed that a book representation using our proposed embeddings significantly improves likability prediction results. Our approach was also able to obtain competitive accuracy for authorship attribution and genre classification, two tasks where style plays a prominent role. Moreover, we also demonstrated that annotated character n-gram embeddings yield higher quality word vectors. These results in likability prediction, authorship attribution, genre classification, and word similarity further demonstrate the usability of annotated character n-grams and author embeddings in varied tasks. In the future, we will extend our method to other domains where authors' information is important, such as author profiling.

Acknowledgments

This work was partially funded by National Science Foundation under award 1462141. We would like to thank the anonymous reviewers as well as Gayle McElvain and Brandon Burroughs for reviewing the paper and providing helpful comments and suggestions.

References

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Pasca, and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and wordnet-based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Boulder, Colorado, pages 19–27. http://www.aclweb.org/anthology/N/N09/N09-1003.

Vikas Ashok, Song Feng, and Yejin Choi. 2013. Success with style: Using writing style to predict the success of novels. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Seattle, Washington, USA, pages 1753–1764. http://www.aclweb.org/anthology/D13-1181.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics* 5:135–146.

Ignacio Iacobacci, Mohammad Taher Pilehvar, and Roberto Navigli. 2015. Sensembed: Learning sense embeddings for word and relational similarity. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Beijing, China, pages 95–105. http://www.aclweb.org/anthology/P15-1010.

Brian Kenji Iwana, Syed Tahseen Raza Rizvi, Sheraz Ahmed, Andreas Dengel, and Seiichi Uchida. 2016. Judging a book by its cover. *arXiv preprint arXiv:1610.09204* .

Vlado Kešelj, Fuchun Peng, Nick Cercone, and Calvin Thomas. 2003. N-gram-based author profiles for authorship attribution. In *In Proceedings of the Pacific Association for Computational Linguistics*. pages 255–264.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, pages 1746–1751. http://www.aclweb.org/anthology/D14-1181.

Moshe Koppel, Jonathan Schler, and Shlomo Argamon. 2009. Computational methods in authorship attribution. *Journal of the American Society for information Science and Technology* 60(1):9–26.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*. PMLR, Bejing, China, volume 32 of *Proceedings of Machine Learning Research*, pages 1188–1196. http://proceedings.mlr.press/v32/le14.html.

Annie Louis and Ani Nenkova. 2013. What makes writing great? first experiments on article quality prediction in the science journalism domain. *Transactions of the Association for Computational Linguistics* 1:341–352.

Thang Luong, Richard Socher, and Christopher Manning. 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*. Association for Computational Linguistics, Sofia, Bulgaria, pages 104–113. http://www.aclweb.org/anthology/W13-3512.

Suraj Maharjan, John Arevalo, Manuel Montes, Fabio A. González, and Thamar Solorio. 2017. A multi-task approach to predict likability of books. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Association for Computational Linguistics, Valencia, Spain, pages 1217–1227. http://www.aclweb.org/anthology/E17-1114.

Suraj Maharjan, Sudipta Kar, Manuel Montes, Fabio A. González, and Thamar Solorio. 2018a. Letting emotions flow: Success prediction by modeling the flow of emotions in books. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. Association for Computational Linguistics, New Orleans, Louisiana, pages 259–265. http://www.aclweb.org/anthology/N18-2042.

Suraj Maharjan, Manuel Montes, Fabio A. González, and Thamar Solorio. 2018b. A genre-aware attention model to improve the likability prediction of books. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, pages 3381–3391. https://doi.org/10.18653/v1/D18-1375.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *International Conference on Learning Representations (ICLR), Workshop* .

Fuchun Peng, Dale Schuurmans, Vlado Keselj, and Shaojun Wang. 2003. Language independent authorship attribution with character level n-grams. In *10th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 267–274.

Andrew J. Reagan, Lewis Mitchell, Dilan Kiley, Christopher M. Danforth, and Peter Sheridan Dodds.

2016. The emotional arcs of stories are dominated by six basic shapes. *CoRR* abs/1606.07772. http://arxiv.org/abs/1606.07772.

Herbert Rubenstein and John B. Goodenough. 1965. Contextual correlates of synonymy. *Commun. ACM* 8(10):627–633. https://doi.org/10.1145/365628.365657.

Younes Samih, Suraj Maharjan, Mohammed Attia, Laura Kallmeyer, and Thamar Solorio. 2016. Multilingual code-switching identification via lstm recurrent neural networks. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*. Association for Computational Linguistics, Austin, Texas, pages 50–59. http://aclweb.org/anthology/W16-5806.

Upendra Sapkota, Steven Bethard, Manuel Montes, and Thamar Solorio. 2015. Not all character n-grams are created equal: A study in authorship attribution. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Denver, Colorado, pages 93–102. http://www.aclweb.org/anthology/N15-1010.

Prasha Shrestha, Sebastian Sierra, Fabio A. González, Manuel Montes, Paolo Rosso, and Thamar Solorio. 2017. Convolutional neural networks for authorship attribution of short texts. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Association for Computational Linguistics, Valencia, Spain, pages 669–674. http://www.aclweb.org/anthology/E17-2106.

Yan Song and Chia-Jung Lee. 2017. Learning user embeddings from emails. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Association for Computational Linguistics, Valencia, Spain, pages 733–738. http://www.aclweb.org/anthology/E17-2116.

Efstathios Stamatatos. 2009. A survey of modern authorship attribution methods. *Journal of the American Society for Information Science and Technology* 60(3):538–556. https://doi.org/10.1002/asi.21001.

Andreas van Cranenburgh and Rens Bod. 2017. A data-oriented model of literary language. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Association for Computational Linguistics, Valencia, Spain, pages 1228–1238. http://www.aclweb.org/anthology/E17-1115.

Kurt Vonnegut. 1981. Palm sunday: An autobiographical collage.

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.

Zhe Zhao, Tao Liu, Shen Li, Bofang Li, and Xiaoyong Du. 2017. Ngram2vec: Learning improved word representations from ngram co-occurrence statistics. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, pages 244–253. https://www.aclweb.org/anthology/D17-1023.