Learning and Evaluating Character Representations in Novels

Naoya Inoue, Charuta Pethe, Allen Kim, Steven Skiena
Stony Brook University
{ninoue,cpethe,allekim,skiena}@cs.stonybrook.edu

Abstract

We address the problem of learning fixed-length vector representations of characters in novels. Recent advances in word embeddings have proven successful in learning entity representations from short texts, but fall short on longer documents because they do not capture full book-level information. To overcome the weakness of such text-based embeddings, we propose two novel methods for representing characters: (i) graph neural network-based embeddings from a full corpus-based character network; and (ii) low-dimensional embeddings constructed from the occurrence pattern of characters in each novel. We test the quality of these character embeddings using a new benchmark suite to evaluate character representations, encompassing 12 different tasks. We show that our representation techniques combined with text-based embeddings lead to the best character representations, outperforming text-based embeddings in four tasks. Our dataset is made publicly available to stimulate additional work in this area.

1 Introduction

High-quality distributed representations of characters (henceforth, *character embeddings*) play an important role for the computational analysis of narrative texts (Iyyer et al., 2016; Xanthos et al., 2016; Skorinkin, 2017; Azab et al., 2019; Labatut and Bost, 2019; Kubis, 2021; Brahman et al., 2021).

Ideally, characters who share similar properties such as job, gender and a relationship to other characters, should possess similar character embeddings even if they are in different stories (e.g. *Cinderella* and *Juliet*, both young women in forbidden romance situations). This paper aims for learning such fixed-length, distributed representations from novels.

The core problem of learning character embeddings is how to aggregate and embed the contextual information of characters into distributed representations. Conventionally, this has been extensively studied in word embeddings, including static word embeddings such as word2vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014), and in contextualized word embeddings such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019). All these methods follow the Distributional Hypothesis: "words that occur in the same context tend to have similar meanings" (Harris, 1954).

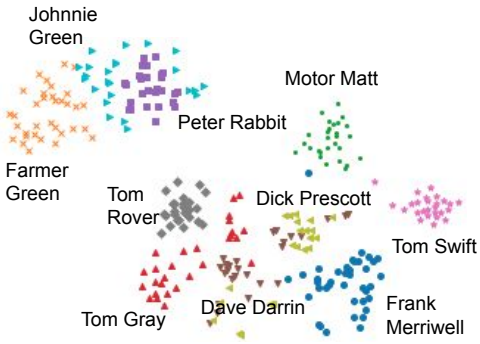

Figure 1: t-SNE visualization of our character embeddings for ten characters. Each character is sampled from more than 24 different books. The proposed method assigns similar representations to each character even though they exist in different books. The proposed method uses no surface form matching.

One limitation of these approaches is that they represent word embeddings by local context: they split documents into individual sentences or small chunks, ignoring the document information of each input. To learn character embeddings, however, it is desirable for an embedding algorithm to be aware of document-level information. This enables us to extend the Distributional Hypothesis to more global context: *characters that occur in the same books/authors tend to have similar or related properties* (e.g. the Sherlock Holmes series tend to have detectives, policemen, criminals, etc.).

To overcome the weakness of such text-based embeddings, we propose two novel methods to learn character embeddings using document-level

Findings of the Association for Computational Linguistics: ACL 2022, pages 1008 - 1019

information. First, we propose *graph-based embeddings*, where we build a full corpus-based character network accompanied with full book-level information and then use a graph neural network to learn character embeddings. Second, we propose *positional embeddings*, where we create low-dimensional embeddings from the occurrence pattern of characters in each novel.

To evaluate the quality of character embeddings, we construct a new character embedding benchmark (*CEB*) consisting of 12 different tasks. At training time, one is allowed to learn fixed-length character embeddings from novels. The learned embeddings are then tested if the important properties of characters such as gender can be recovered *solely* based on them, similar to recent work on probing pretrained language models (Hewitt and Manning, 2019; Voita and Titov, 2020, etc.).

The contribution of this paper can be summarized as follows:

- *New methods for character embeddings* – We propose two novel methods for learning character embeddings leveraging full book-level information (§4).
- *Evaluation of character embeddings* – We create a novel benchmark suite (CEB) for testing the quality of character embeddings, consisting of 12 different tasks (§5). The dataset and evaluation script are publicly available at `https://github.com/naoya-i/charembench`.

  Our experiments show that the proposed embedding methods combined with text-based embeddings leads to the best character embeddings, outperforming text-based embeddings in six CEB tasks (§6.3).
- *Corpus-level views of character embeddings* – We show that character embeddings cluster across large corpora by gender, protagonist status, profession/role, thus demonstrating the versatility of the techniques we employ (§7). Fig. 1 shows the key result, indicating that similar character representations are assigned to each cluster of character, even though they exist in different books.

2 Related work

There is a growing interest in computational narrative analysis, ranging from analyzing the structure of narratives (Kim et al., 2020, 2021; Pethe et al., 2020), identifying important events in stories (Wilmot and Keller, 2020, 2021; Papalampidi et al., 2020; Otake et al., 2020) to analyzing the relationship between characters in novels (Iyyer et al., 2016; Xanthos et al., 2016; Skorinkin, 2017; Azab et al., 2019; Labatut and Bost, 2019; Kubis, 2021; Brahman et al., 2021). The most relevant work to ours is Azab et al. (2019), who apply word2vec (Mikolov et al., 2013) to learn character embeddings from movie scripts. However, they do not use full document-level information such as the author of documents for learning character embeddings. They also experiment on a small-scale dataset–18 movie scripts, while we experiment on 17k novels. Brahman et al. (2021) propose two benchmark tasks for character-centric narrative understanding, namely character identification and character description generation. We extend their benchmark by introducing additional 12 character-related tasks.

Character embeddings are closely related to both static word embeddings such as word2vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014), and contextualized word embeddings such as dynamic entity embeddings (Kobayashi et al., 2016), ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019). As discussed in §1, these methods follow the Distributional Hypothesis (Harris, 1954), encoding the local context of words into distributed representations. We intend to complement this weakness by taking book-level context into account in the graph neural network-based embedding methods.

The task setting of CEB shares the similar spirit to a recent paradigm on probing pretrained language models (Hewitt and Manning, 2019; Petroni et al., 2019; Voita and Titov, 2020; Shin et al., 2020). The LAMA dataset (Petroni et al., 2019), for example, creates a sentence with blanks, e.g. ___ *was born in*, and ask language models to predict words in the blanks *solely based on* the learned model parameters. Our benchmark also follows this task setting, where one learns character embeddings on a particular corpus and is asked to recover information solely based on the learned embeddings in 12 different tasks.

3 Baseline text-based methods

3.1 Static embeddings

One simple way to learn character embeddings is to treat each character name as one unique token

at the document-level and apply standard word embedding algorithms. Given a corpus, we convert all character mentions including pronouns to special tokens consisting of its document ID and character name (e.g. *When* 113_Mary *was sent to...*). To identify character mentions and coreference relations between them, we use Stanford CoreNLP (Manning et al., 2014). See §5.1 for further details.

We then apply word2vec (Mikolov et al., 2013). Because a corpus of novels alone may not provide enough data to learn non-character word vectors, we initialize non-character word vectors with GloVe pretrained embeddings (Pennington et al., 2014).[1] Henceforth, we call this method `w2v`.

We also apply doc2vec (Le and Mikolov, 2014) to the preprocessed corpus, where we treat each character as one document and sentences that mention this character as the content of this document. Henceforth, we call this method `d2v`.

3.2 Context-aggregated embeddings

Another simple way to learn character embeddings is to aggregate contextual information of characters (Ethayarajh, 2019; Bommasani et al., 2020). Given a character c, we extract set $S(c)$ of sentences that mention c and generate a sentence representation $\mathbf{s}_i$ for each $s_i \in S(c)$. We then aggregate them via averaging: $\mathbf{c} = \frac{1}{|S(c)|} \sum_{s_i \in S(c)} \mathbf{s}_i$.

To generate $\mathbf{s}_i$, we explore two methods. The first method is `w_ag`, which simply averages word embeddings learned in Sec. 3.1: $\mathbf{s}_i = \frac{1}{|s_i|} \sum_{w_j \in s_i} \mathbf{w}_j$. We also make `gl_ag`, a variation of this model using vanilla GloVe pretrained embeddings (Pennington et al., 2014).

Another method is `rb_ag`, which uses contextualized word embeddings of characters generated by RoBERTa (Liu et al., 2019). Given $s_i \in S(c)$, we first replace character mentions of c with mask tokens. For example, suppose $c =$ *Mary* and $s_i =$ *Mary was most attracted by the mother and Dickon.* The sentence is then converted to *[MASK] was most attracted by the mother and Dickon.* To generate $\mathbf{s}_i$, we extract contextualized word embeddings of [MASK] tokens at the final layer.

3.3 Name embeddings (`nam`)

Ye et al. (2017) represent common first/last names using a vector representation that encodes gender, ethnicity, and nationality which is readily applicable to building classifiers and other systems. Name embeddings exploit the phenomenon of homophily in communication, specifically that people tend to associate with similar people or popularly that "birds of a feather flock together." These embeddings are constructed from email contact lists of email, rosters of friends on social media, or followers on Twitter. The homophily-induced coherence of these contact lists enables us to derive meaningful features using word embedding methods. We used 100 dimensional embeddings from (Ye and Skiena, 2019).

[1] CommonCrawl-840B-300d at `https://nlp.stanford.edu/projects/glove/`.

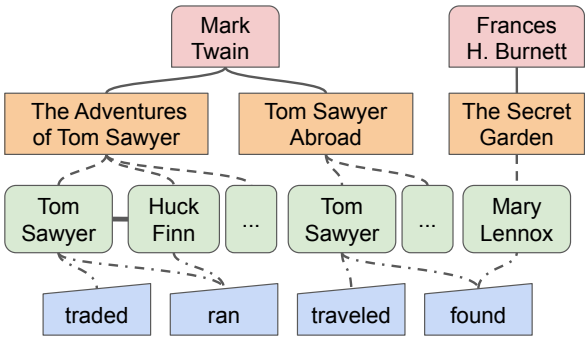

Figure 2: Example of character network. Characters (green) are connected through book-level information, i.e. books (orange) and authors (red). Context information (green) captures the attributes of characters.

4 Proposed methods

While text-based embeddings introduced in §3 can be expected to capture the local context of characters such as gender, they do not take into account full book-level information, such as the author. Intuitively, characters from the same book should have more relatively similar embeddings than those from different books, but the text-based embedding methods cannot use this kind of information. To address this weakness, we propose two methods for character embeddings: (i) `gr`: we build character network across books and then learn character embeddings using Graph Neural Networks (§4.1); and (ii) `pos`: we encode the occurrence pattern of characters into low-dimensional embeddings (§4.2).

4.1 Graph-based embeddings

4.1.1 Character network

Our character network is an undirected graph consisting of four types of nodes and four types of unlabeled edges as shown in Fig. 2.

Nodes. First, we introduce (i) *book nodes* (e.g. *The Adventures of Tom Sawyer*), (ii) *author nodes* (e.g. *Mark Twain*), and (iii) *character nodes* (e.g.

Node type	# nodes	Edge type	# edges
Book	17,275	Bk-Au	17,514
Character	718,324	Bk-Chr	712,332
Author	4,422	Chr-Con	30,934,451
Context	147,000	Chr-Chr	446,917

Table 1: Statistics of character network.

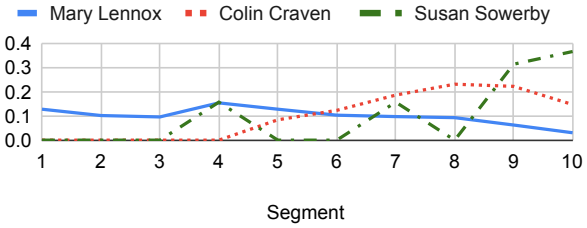

Figure 3: Positional embeddings for characters from *The Secret Garden*. *Mary* and *Colin*, the main characters, indicate continuous appearance throughout the book, while *Susan*, one of the minor characters, indicates discontiguous appearance.

Tom Sawyer), each of which represents individual book, author, and character in the corpus. Note that we keep characters with the same name as separate nodes in the network (e.g. *Tom Sawyer*) because it is not obvious if these characters are indeed the same character or not at this point. As described later, if characters are inferred to be the same from book-level information, these embeddings become similar given the network configuration.

Second, we introduce (iv) *context nodes* which represent the local context information of characters (e.g. *traded*). Following Bamman et al. (2014), we extract words that are connected with a character name in agent, patient, possessive, or predicative dependency relations as context.

Edges. We introduce (i) *book-author edges* connecting book node n_b with author node n_a if n_a is the author of n_b (e.g. *The Adventures of Tom Sawyer–Mark Twain*), and (ii) *book-character edges* connecting book node n_b with character node n_c if n_c appears in n_b (e.g. *The Adventures of Tom Sawyer–Tom Sawyer*). To associate context with characters, we have (iii) *character-context edges* connecting context nodes with character nodes if they have a dependency relation described above (e.g. *Tom Sawyer–traded*). To capture the interaction between characters, we introduce (iv) *character edges* connecting two character nodes n_{c_1}, n_{c_2} if c_1 and c_2 occur within 10 tokens of each other at least 10 times (e.g. *Tom Sawyer–Huck Finn*).

Table 1 shows the statistics of our character network constructed from 17,275 books from Project Gutenberg (see §5.1 for the details of dataset).

4.1.2 Learning embeddings

We use DeepWalk (Perozzi et al., 2014), which is a representation learning algorithm for graph-structured data. It samples graph paths by random walk and then applies word2vec algorithm (Mikolov et al., 2013) to the sampled paths, treating each node as one word.

The main advantage over the text-based methods is as follows. In the text-based methods, two characters from different novels never appear in the same sentence. In contrast, in the graph-based method, two characters may appear in the same sentence (or path) if they are connected via book nodes or author nodes, which makes two character embeddings closer (e.g. two *Tom Sawyer* via *Mark Twain* in Fig. 2). In other cases, two characters from different novels may appear in the same sentence (or path) if they share context nodes (e.g. *Tom Sawyer* and *Mary Lennox* via *found* in Fig. 2), which makes two characters with similar properties closer. This means that we inject document-level information into character embeddings.

4.2 Positional embeddings

The main character in novels is likely to always appear throughout the story, while a minor character may appear a few times in one chapter and disappear. Such document-level occurrence patterns are not captured by text-based methods, but they may encode useful information about characters.

We thus propose `pos` embeddings purely based on the pattern of mention positions of characters. We divide a novel into 10 segments and count the occurrences of each character i in each segment j (denoted $c_{i,j}$). As exemplified in Fig. 3, we then create two 10-dimensional embeddings by (i) normalizing $c_{i,j}$ across characters, i.e. $\mathbf{c}_i^c = \mathbf{c}_i / \sum_i c_{i,j}$, denoting how important the character is for the segment; (ii) normalizing $c_{i,j}$ across segments, i.e. $\mathbf{c}_i^s = \mathbf{c}_i / \sum_j c_{i,j}$, denoting how important the segment is for the character. Finally, we concatenate these, i.e. $[\mathbf{c}_i^c; \mathbf{c}_i^s]$, to form 20-dimensional embeddings. We repeat the same procedure with pronoun mentions, and concatenate these vectors to obtain final 40-dimensional positional embeddings for each character.

Task	Input	Output	Source	Size
Gender	One char	Male/Female	Heurstics (§5.2)	5,000
Role	One char, Four choices of roles	Role of a character (e.g. schoolmaster)	Reference books	484
Protagonist	One char	Protagonist/Other	Frequency	5,000
Identity	Two chars from different books	Yes/No (if two chars are same)	Metadata	5,000
Cloze	Sentence w/ blank (e.g. ___ *is born in India*), Four choices of chars	A character in the blank	Book content	5,000
Speaker	Quote, Four choices of chars	Speaker of the quote	Book content	2,879
Summary Cloze	Sentence w/ blank from chapter summary, Four choices of chars	A character in the blank	Literature websites	1,361
Desc	Description (e.g. *A simple , but honest and loyal black worker...*), Four choices of chars	A character that is best described by the given description	Literature websites	551
QA	Question (e.g. *Who does Mary Lennox accept an invitation from?*), Four choices of chars	Answer	Kočiský et al. (2017); Angelidis et al. (2019)	587
Author	Two chars	Yes/No (if two chars are from the same author's books)	Metadata	5,000
Book	Two chars	Yes/No (if two chars are from the same books)	Metadata	5,000
Genre	One char, Genre	Yes/No (if the character belongs to a book with the given genre)	Metadata	44,152

Table 2: Overview of CEB, a benchmark suite for character embeddings.

5 CEB: Character Embedding Benchmark

To test the quality of character embeddings, we construct a new benchmark suite of character embeddings, as summarized in Table 2. The benchmark probes what kind of character-related information, ranging from gender to authors, is embedded in character embeddings. It consists of 12 different tasks categorized into three levels: (i) *character-level tasks*: identifying character attributes (§5.2), (ii) *context-level tasks*: identifying the correct character that best describes a given context (§5.3), and (iii) *book-level tasks*: identifying the attributes of books where characters come from (§5.4).

5.1 Dataset

We extract 17,275 books from Project Gutenberg[2], a publicly available library of free eBooks. We use Stanford CoreNLP (Manning et al., 2014) for NER (Named Entity Recognition). We use the named entities of type `PERSON` as potential character mentions, and follow a rule-based approach similar to Vala et al. (2015) for clustering variants of the same name, and obtaining a final list of characters for each book. To ensure that tested character embeddings have sufficient information, we discarded characters with less than 100 mentions.

5.2 Character-level tasks

Gender Identify the gender of a given character c (female or male). To identify the gold-standard gender of a character, we count the number of male and female pronouns referring to each character (as annotated by CoreNLP), and take a majority vote. If the male pronoun count outnumbers the female pronoun count by at least 10%, we consider the character to be male, and vice versa for female.

Role Identify the role of a given character c. We extract gold-standard character roles from two reference books of English literature (Magill, 1968, 1952), where character roles are represented by simple natural language phrases such as *a French aristocrat*. We extract only head nouns by the dependency parse given by Spacy.[3]

Protagonist Identify whether a given character c is a protagonist or not. As approximation, we identify the most frequent characters as the gold-standard protagonist.

Identity Given two characters c_1, c_2 from different books, identify whether c_1 is the same character as c_2 or not. We use characters with the same full name and the same author as a positive instance.

[2] http://www.gutenberg.org/

[3] https://spacy.io/usage

5.3 Context-level tasks

Cloze Given a sentence S with a blank (e.g. ___ *stood up and tried to keep her eyes open while Mrs. Medlock collected her parcels.*) from book b and four candidate characters from b, choose the character c that best fits into the blank. To sample difficult wrong candidates, we sample characters with similar frequency in all the context-level tasks. Specifically, we use characters c' s.t. $r(c) - 2 \leq r(c') \leq r(c) + 1$, where r is the rank of frequency.

Speaker Given a quote Q (e.g. *"Well, it was this way. I was leaning on the stile..."*) from book b (≥ 50 words) and four candidate characters from b, choose the character that spoke this quote.

Summary Cloze Similar to Cloze, given a sentence S with a blank from a chapter summary of book b and four candidate characters from b, choose the character that best fits into the blank. We extract chapter summaries from LitCharts, an online guide for English literature.

Desc Given a character description snippet D (e.g. *A simple , but honest...*) and four candidate characters from the same book, choose the character that is best described by D. We extract character descriptions from five reliable web sources.[4]

QA Given a question about characters (e.g. *Who brings Mary Lennox the garden tools?*) and four candidate characters from the same book b, choose the character that best fits as the answer. We extract character-related questions (Angelidis et al., 2019) from NarrativeQA (Kočiský et al., 2017).

5.4 Book-level tasks

Author Given two characters from two different books b_1, b_2, identify whether the authors of b_1 and b_2 are the same or not.

Book Given two characters from two books b_1, b_2, identify whether b_1 and b_2 are the same.

Genre Identify the book genre of a given character c. Because one book can belong to more than one genre, we manually selected 11 frequent subjects from Project Gutenberg's metadata and turn them into 11 binary classification tasks[5] and report an average accuracy.

6 Evaluation

6.1 Setup

We follow recent work on probing word embeddings, which report that one should employ less expressive classifiers in order to prevent the classifier itself from learning to solve the probe tasks (Voita and Titov, 2020). At training time, one has access to all books and learns fixed-length character embeddings of each character. At test time, we freeze the learned character embeddings and train task-specific linear classifiers using the learned embeddings as a feature vector.

To solve classification tasks, we train a linear classifier that uses learned character embeddings as a feature vector. For pairwise classification, we merge two character embeddings by element-wise multiplication and absolute element-wise difference, i.e. $[\mathbf{c}_1 \odot \mathbf{c}_2; |\mathbf{c}_1 - \mathbf{c}_2|]$. In our experiments, we employ Support Vector Machines (Cortes and Vapnik, 1995). To solve multiple-choice tasks with context x and characters $\{c_i\}_{i=1}^4$, we train a scorer $f(x, c_i) = (W\mathbf{x} + \mathbf{b}) \cdot \mathbf{c}_i$ with a cross entropy loss, where $W, \mathbf{b}$ is a learned projection from the embedding space of context to characters. We use Sentence Transformers (Reimers and Gurevych, 2019)[6] to encode x into $\mathbf{x}$.[7]

The test instances with binary classification tasks are all balanced. Therefore, we use an accuracy as evaluation measure for all the tasks. To see overall picture, for each task category we calculate a *final score* by an average of task accuracies. We use 5-fold cross validation for evaluation and report an average accuracy. For the task with less than 2,000 instances (i.e. Role, Summary Cloze, Desc, QA), we use 10-fold cross validation to secure more training data.

6.2 Hyperparameters

For static embeddings, we use gensim implementation of word2vec (CBOW) and doc2vec.[8] We kept only top one million words in the vocabulary and trained 300-dimensional vectors with 5 epochs, 10 context words, and 10 negative examples.

[4] GradeSaver, LitCharts, CliffsNotes, Schmoop, SparkNotes.

[5] Selected subjects are: 19th century, adventure stories, detective and mystery stories, fiction, historical fiction, humorous stories, juvenile fiction, love stories, science fiction, short stories, western stories.

[6] We use all-MiniLM-L12-v2, a publicly available pretrained model of Sentence Transformers at https://www.sbert.net/docs/pretrained_models.html.

[7] For the role task, x is a character embedding, and c_i is a Sentence Transformer embedding of a role.

[8] https://radimrehurek.com/gensim/

	Character-level				Context-level					Book-level			Final score		
Model	**gen**	**role**	**prot**	**id**	**clz**	**spk**	**sclz**	**desc**	**QA**	**auth**	**book**	**genre**	**Ch**	**Co**	**Bk**
rand	50.0	25.0	50.0	50.0	25.0	25.0	25.0	25.0	25.0	50.0	50.0	50.0	43.8	25.0	50.0
w2v	88.6	41.9	75.4	92.7	32.9	38.8	37.7	40.7	39.7	70.8	92.1	76.4	74.7	38.0	79.8
d2v	87.2	40.1	71.1	95.3	32.5	32.0	29.3	43.6	33.7	79.1	92.3	78.9	73.4	34.2	83.4
nam	85.9	28.5	54.9	**99.9**	27.5	27.7	32.6	31.8	30.2	52.7	56.6	57.4	67.3	30.0	55.6
gl_ag	91.3	29.7	69.5	95.9	37.0	32.4	40.6	36.5	37.1	79.9	90.0	80.5	71.6	36.7	83.5
w_ag	91.8	31.8	73.1	96.3	37.3	35.3	40.8	45.9	39.4	79.5	89.2	**81.6**	73.3	39.7	83.4
rb_ag	96.6	40.5	86.7	96.7	**38.5**	43.5	**48.0**	**51.2**	41.6	75.3	84.8	79.9	80.1	44.6	80.0
gr	**98.6**	36.1	75.0	96.7	32.5	**49.5**	40.2	38.1	34.4	**85.6**	95.5	80.2	76.6	38.9	**87.1**
pos	52.2	30.8	86.2	74.9	26.0	45.5	40.1	27.6	37.1	54.9	60.5	55.7	61.0	35.3	57.0
rb_ag+ gr+pos	98.1	**43.2**	**92.4**	97.8	36.6	48.5	46.5	50.6	**42.7**	83.9	**95.6**	81.2	**82.9**	**45.0**	86.9

Table 3: Results on CEB. Text-based embeddings capture character-level information better, while graph-based methods capture book-level information better. Combining these two methods leads to the best embeddings.

For graph-based embeddings, we use the original implementation of DeepWalk[9] with 100-dimensional embeddings. We set the length of random walk path to 50 nodes and the number of random walks to start at each node to 20, and kept other hyperparameters as the default values.

We train the multiple-choice classifier for 10 epochs, using AdamW with batch size of 16, learning rate of 1e-3, and weight decay of 1e-2.

6.3 Results and discussion

The results are shown in Table 3. It shows that text-based methods perform better on character-level tasks and context-level tasks, while the graph-based method performs better on book-level tasks. This suggests that text-based methods can capture the local context of characters such as gender better, but it does not take into account document-level context discussed in §4.1. Name embeddings prove effective only at capturing gender.

Despite its simplicity, positional embeddings show surprisingly good performance on the character-level tasks (protagonist, identity) and context-level tasks (QA). This indicates that the occurrence patterns are deeply related to determining the importance of characters in books and that if the same character appears in different books, the occurrence patterns are also similar to each other. The good performance of QA indicates that the relationship between two characters are captured to some extent only by the occurrence patterns.

We then combined the best text-based embedding, `rb_ag`, with `gr` and `pos` (the last row).[10] The results indicate that they complement each other's strength and weakness. For example, `rb_ag`'s low performance on the author and book tasks and `gr`'s low performance on the protagonist and cloze tasks improved. Overall, the proposed methods using book-level information outperformed the text-based methods in four tasks, indicating the importance of book-level information in character representations.

In order to investigate the effect of introducing global edges, we ablate author-book edges (a,b) and character-character edges (c,c) from the proposed graph embedding method. The results are shown in Table 4. '-(c,c)' experiences more performance degradation in context-level tasks and book-level tasks than '-(a,b)', which indicates that character interaction provides useful information especially for these tasks. When both edges are removed, we observe performance drop in nine tasks, again indicating their need for character representations.

7 Qualitative analysis

To obtain further insights on the learned character embeddings, we visualize `rb_ag+gr+pos` by using t-SNE (van der Maaten and Hinton, 2008) with default hyperparameters.

7.1 Universality across books

In Fig. 1, we intend to check the universality of the learned character embeddings across books. We sampled characters with the same name and the same author from different books and plotted 281 samples of their character embeddings. This identifies characters that appear in a series of books, e.g.

[9] https://github.com/phanein/deepwalk

[10] We simply concatenated three embeddings, which yields 908-dimensional (768 + 100 + 40) embeddings.

	Character-level				Context-level					Book-level		
Model	**gen**	**role**	**prot**	**id**	**clz**	**spk**	**sclz**	**desc**	**QA**	**auth**	**book**	**genre**
graph	**98.6**	36.1	75.0	**96.7**	32.5	**49.5**	**40.2**	**38.1**	34.4	**85.6**	95.5	**80.2**
-(c,c)	**98.6**	**44.8**	74.7	95.5	32.2	46.8	37.0	35.6	**40.2**	81.4	89.4	79.1
-(a,b)	98.5	39.7	**75.3**	96.3	31.8	45.1	40.0	35.6	36.1	85.5	**95.6**	**80.2**
-(c,c)(a,b)	98.3	39.4	75.2	95.5	**33.0**	47.3	35.2	35.9	33.4	81.3	89.7	78.9

Table 4: Ablation study of character network embeddings.

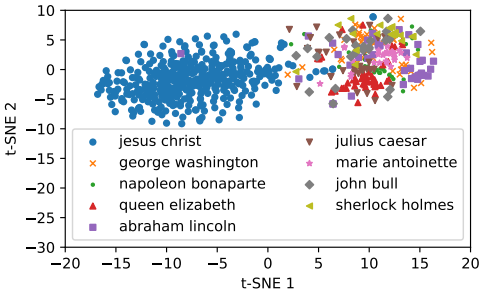

Figure 4: Character embeddings of historical figures.

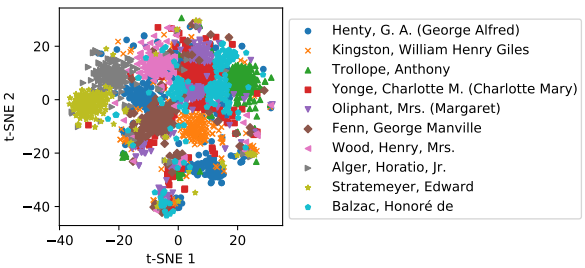

Figure 5: Character embeddings colored by author.

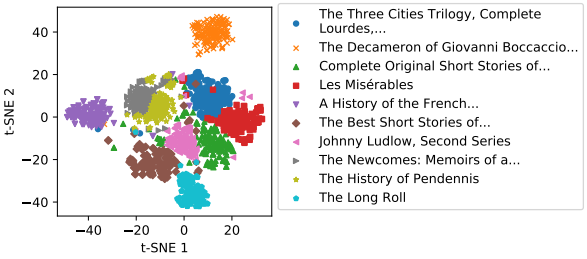

Figure 6: Plot of character embeddings colored by book.

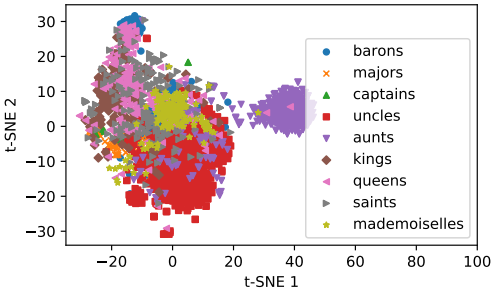

Figure 7: Character embeddings colored by titles.

Peter Rabbit in *The Tale of Peter Rabbit*. Interestingly, Fig. 1 shows that even though such characters appear in different books, the learned embeddings are close to each other. This suggests that the proposed method can capture the book-independent, universal property of characters.

To further confirm the universality of character embeddings, we manually identified 662 famous, historical figures such as *Jesus Christ* and *George Washington* in Project Gutenberg books and plotted character embeddings in Fig. 4. Similar to Fig. 1, it shows one big cluster for Jesus Christ and small clusters for the rest of historical figures, again indicating the universal property of our character embeddings.

While our goal is to learn book-independent universal character embeddings, we check to see if the character embeddings also preserve book-level information. Fig 5 shows character embeddings colored by the author of the book that each character came from. Fig. 6 visualize the learned character embeddings, where the datapoints are labeled by books. The results suggest that character embeddings also encode book-level information.

7.2 Character property

When characters have similar property (e.g. profession), it is desirable to have similar embeddings even though they exist in different books. This section studies the following three properties.

Profession/role Fig. 7 visualizes 2,232 characters that have manually specified titles (e.g. kings, aunts) across different books. We see a clear cluster for each title, and queens, kings and barons being close to each other (left). This indicates another book-independent, universal property of our embeddings from the profession/role's perspective. Note that our training methods do not exploit the titles for learning character embeddings: they convert the whole character name including the title as one unique special token (see §3).

Distance	Name	Gender	Book title	Book author	Juvenile?
0.00	Mary Lennox	Female	The Secret Garden	Burnett, Frances Hodgson	Y
1.44	Sibyl Ogilvie	Female	Daddy's Girl	Meade, L. T.	Y
1.56	Margaret Montfort	Female	Margaret Montfort	Richards, Laura Elizabeth Howe	Y
1.60	Betty Randall	Female	The Children on the Top Floor	Rhoades, Nina	Y
1.61	Carol	Female	Sunny Slopes	Hueston, Ethel	N
1.62	Matilda Laval	Female	Trading	Warner, Susan	Y

Table 5: Five nearest neighbors for *Mary Lennox* from *The Secret Garden.*

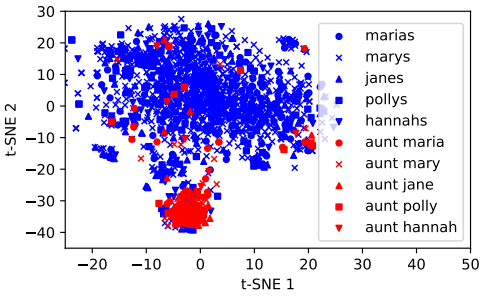

Figure 8: Character embeddings colored by aunts (red) and non-aunt characters (blue).

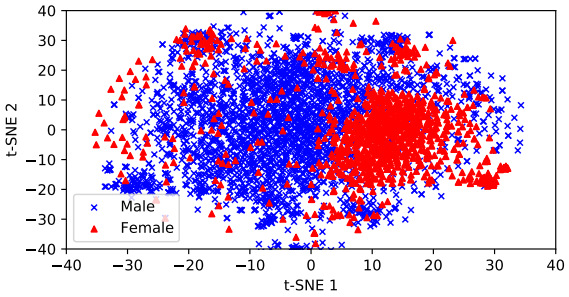

Figure 9: Character embeddings colored by gender.

To see if characters playing a specific role are separated from ordinary characters in our embedding space, we extracted 1,360 characters with the name *aunt X* and *(non-aunt) X* across books and plotted their character embeddings in Fig. 8. We see that aunts and non-aunts form separate clusters. This again supports that our character embeddings also capture the profession/role of characters.

Gender Fig. 9 visualizes 4,000 random samples of character embeddings across books, each of which is labeled with their gender. This clearly shows the clusters of female, indicating that the character embeddings have learned their gender.

Protagonist status Fig. 10 visualizes 4,000 protagonists and non-protagonists across books (4.9% of them are the protagonist). This clearly indicates that the character embeddings have learned protagonist status.

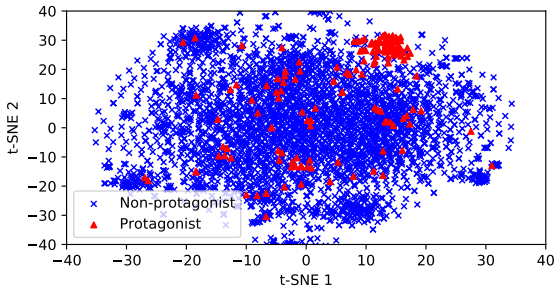

Figure 10: Character embeddings colored by protagonist status.

7.3 Nearest neighbors

To give a closer inspection, we show the list of nearest neighbor characters for *Mary Lennox*, the main female character from *The Secret Garden*, in Table 5. It successfully lists characters with similar attribute at a both character-level and book-level. For example, *Sibyl Ogilvie*, *Betty Randall* are female children of age similar to Mary from juvenile books.

8 Conclusions

We have addressed the problem of learning fixed-length, dense character representations from book-length narrative texts. To overcome the weakness of the text-based embeddings, we have proposed graph-based embeddings and positional embeddings. To test the quality of character embeddings, we have also constructed CEB, a novel benchmark suite for evaluating character embeddings, consisting of 12 different tasks. Our experiments have demonstrated that the proposed embeddings combined with text-based embeddings lead to the best character embeddings, outperforming text-based embeddings in four tasks. We also showed that character embeddings capture both character-level and book-level information across books, demonstrating the versatility of the techniques we employed.

Acknowledgements

We would like to thank anonymous reviewers for valuable and insightful feedback.

References

Stefanos Angelidis, Lea Frermann, Diego Marcheggiani, Roi Blanco, and Lluís Màrquez. 2019. Book QA: Stories of challenges and opportunities. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 78–85, Hong Kong, China. Association for Computational Linguistics.

Mahmoud Azab, Noriyuki Kojima, Jia Deng, and Rada Mihalcea. 2019. Representing movie characters in dialogues. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 99–109, Hong Kong, China. Association for Computational Linguistics.

David Bamman, Ted Underwood, and Noah A. Smith. 2014. A Bayesian mixed effects model of literary character. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 370–379, Baltimore, Maryland. Association for Computational Linguistics.

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting Pretrained Contextualized Representations via Reductions to Static Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4758–4781, Online. Association for Computational Linguistics.

Faeze Brahman, Meng Huang, Oyvind Tafjord, Chao Zhao, Mrinmaya Sachan, and Snigdha Chaturvedi. 2021. "let your characters tell their story": A dataset for character-centric narrative understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1734–1752, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273–297.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Kawin Ethayarajh. 2019. How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China. Association for Computational Linguistics.

Zellig Harris. 1954. Distributional structure. *Word*, 10(2-3):146–162.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Mohit Iyyer, Anupam Guha, Snigdha Chaturvedi, Jordan Boyd-Graber, and Hal Daumé III. 2016. Feuding families and former Friends: Unsupervised learning for dynamic fictional relationships. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1534–1544, San Diego, California. Association for Computational Linguistics.

Allen Kim, Charuta Pethe, Naoya Inoue, and Steve Skiena. 2021. Cleaning dirty books: Post-OCR processing for previously scanned texts. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4217–4226, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Allen Kim, Charuta Pethe, and Steve Skiena. 2020. What time is it? temporal analysis of novels. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9076–9086, Online. Association for Computational Linguistics.

Sosuke Kobayashi, Ran Tian, Naoaki Okazaki, and Kentaro Inui. 2016. Dynamic entity representation with max-pooling improves machine reading. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 850–855, San Diego, California. Association for Computational Linguistics.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2017. The narrativeqa reading comprehension challenge.

Marek Kubis. 2021. Quantitative analysis of character networks in Polish 19th- and 20th-century novels. *Digital Scholarship in the Humanities*, 36(Supplement_2):ii175–ii181.

Vincent Labatut and Xavier Bost. 2019. Extraction and analysis of fictional character networks. *ACM Computing Surveys*, 52(5):1–40.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1188–1196, Bejing, China. PMLR.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

F.N. Magill. 1952. *Masterpieces of World Literature in Digest Form*. Harper.

F.N. Magill. 1968. *Masterplots: 1510 Plot Stories & Essay Reviews from the World's Finest Literature*. Salem Press.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

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*, volume 26. Curran Associates, Inc.

Takaki Otake, Sho Yokoi, Naoya Inoue, Ryo Takahashi, Tatsuki Kuribayashi, and Kentaro Inui. 2020. Modeling event salience in narratives via barthes' cardinal functions. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1784–1794, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Pinelopi Papalampidi, Frank Keller, Lea Frermann, and Mirella Lapata. 2020. Screenplay summarization using latent narrative structure. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1920–1933, Online. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 701–710, New York, NY, USA. ACM.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Charuta Pethe, Allen Kim, and Steve Skiena. 2020. Chapter Captor: Text Segmentation in Novels. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8373–8383, Online. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

D. A. Skorinkin. 2017. Quantitative analysis of character networks in Polish 19th- and 20th-century novels. In *Proceedings of the International Conference Dialogue 2017*, page 14 pages.

Hardik Vala, David Jurgens, Andrew Piper, and Derek Ruths. 2015. Mr. bennet, his coachman, and the archbishop walk into a bar but only one of them gets recognized: On the difficulty of detecting characters in literary texts. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 769–774, Lisbon, Portugal. Association for Computational Linguistics.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

Elena Voita and Ivan Titov. 2020. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*,

pages 183–196, Online. Association for Computational Linguistics.

David Wilmot and Frank Keller. 2020. Modelling suspense in short stories as uncertainty reduction over neural representation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1763–1788, Online. Association for Computational Linguistics.

David Wilmot and Frank Keller. 2021. Memory and knowledge augmented language models for inferring salience in long-form stories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 851–865, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Aris Xanthos, Isaac Pante, Yannick Rochat, and Martin Grandjean. 2016. Visualising the dynamics of character networks. In *Digital Humanities*, pages 417–419.

Junting Ye, Shuchu Han, Yifan Hu, Baris Coskun, Meizhu Liu, Hong Qin, and Steven Skiena. 2017. Nationality classification using name embeddings. In *Proceedings of the ACM Conference on Information and Knowledge Management (CIKM '17)*, pages 1897–1906.

Junting Ye and Steven Skiena. 2019. The secret lives of names? name embeddings from social media. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3000–3008.