

A Computational Expedition into the Undiscovered Country - Evaluating Neural Networks for the Identification of Hamlet Text Reuse

Maximilian Bryan^a, Manuel Burghardt^a and Johannes Molz^a

^aLeipzig University, Augustusplatz 10, 04109 Leipzig, Germany

Abstract

In this article, we describe a two-step processing pipeline for identifying text reuse of Shakespeare's Hamlet in a corpus of postmodern fiction by comparing n-grams from both sources. A key feature of our approach lies in a pre-filtering step, in which we select target sentences in the fiction corpus that are potential candidates for Hamlet text reuse. Without pre-filtering, the amount of text reuse pairs (that are no actual quotes) would be extremely high. In a second filtering step, we compare potential text reuse pairs by their vector representation using a neural network trained in an unsupervised manner. We found that using the vector similarity produces a problematic amount of false positives. The created vector representations are created using an unsupervised training approach, resulting in similarity aspects that are unfavorable for our use case.

Keywords

text reuse, intertextuality, Shakespeare, neural networks

1. Introduction

Intertextuality is an approach to literary studies that assumes that works of literature are never independent but are instead part of a bigger network of textual relations, i.e., literary texts are, to some degree, always influenced and informed by the texts that preceded them [1]. The most objective way of studying intertextuality is to look at the *recognizable re-occurrence* [11] of an earlier text in a later text. The detection of such recognizable re-occurrences, i.e. more or less verbatim repetitions of parts of one text in another, is very well suited for a computational approach, as the area of *text reuse*, which typically has applications in plagiarism detection and information retrieval, provides a well-equipped toolbox of methods and algorithms [2, 3, 4, 18, 10]. More concretely, we are interested in using computational methods for the detection of Shakespearean intertextuality (see [15]) in postmodern fiction, as Shakespeare's words, topics, characters, and plots are present in some of the most successful writers of the genre, like for instance Neil Gaiman & Terry Pratchett [21].

In a previous pilot study, we experimented with a local alignment algorithm (Smith-Waterman). As we aligned all the n-grams of Shakespeare's work with the contemporary corpus, this was a computationally intensive approach that returned a high number of false positives [7]. In this work, we present the first results of an alternative approach to the detection of


CHR 2020: Workshop on Computational Humanities Research, November 18–20, 2020, Amsterdam, The Netherlands

✉ bryan@informatik.uni-leipzig.de (M. Bryan); burghardt@informatik.uni-leipzig.de (M. Burghardt); johannes.molz@googlemail.com (J. Molz)

📄 0000-0003-1354-9089 (M. Burghardt)

© 2020 Copyright for this paper by its authors.

Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

intertextuality, which relies on a two-step NLP pipeline. First, we use a classifier to extract sentences in our corpus that are potential candidates for quoting Shakespeare. Second, we compare these candidate sentences to the actual Shakespearean texts. This comparison is implemented using a sequence vector representation created by a siamese neural network trained on a large collection of news articles¹.

The rest of the paper is structured as follows: We first give an overview of text reuse approaches that rely on neural networks and related techniques. Next, we describe our NLP pipeline and focus on the two neural networks that were used before explaining our process in more detail. After that, we use our pipeline to discover intertextual references to Hamlet in a data set of postmodern fictional literature and discuss our findings.

2. Related work

Text reuse has been a popular use case in computational linguistics and information retrieval for many years. However, most of the existing approaches so far have relied on statistical approaches rather than on alternatives that come to mind when looking at neural networks. We claim that neural networks are well suited for the computer-aided investigation of text reuse and present an innovative approach to do so in this paper.

This section gives an overview of the related work that uses statistical methods for text reuse detection. We also provide related work for the usage of neural networks for tasks very similar to text reuse and intertextuality detection, making a case for an existing research gap to use a neural network architecture for the identification of text reuse.

Statistical approaches. [14] use tri-grams to represent text passages. When comparing these tri-grams to other text passages, the Jaccard index is used to measure overlap. A major downside of this simplistic approach is that synonyms or more complex rephrasing also will result in a low tri-gram overlap. A similar approach is used by [22]. Here, longer n-grams with lengths 4-7 are used to represent documents, while frequent n-grams are filtered out. The remaining n-grams are hashed into a set of predefined buckets, with overlapping words being put into the same bucket. Another approach that hashes n-grams is presented by [25]. They calculate a one-sided Jaccard index, which allows them to find out how much of a text passage is included in a bigger passage. An approach using alignments (similar to [7]) is presented by [24]. Here, the BLAST algorithm is used to find reused text given query words.

When looking for a reused sentence, it is interesting to dissect the identified, possibly reused, text parts by their type of similarity. A threefold way for looking at such similarities is presented by [3], who distinguish *content*, *structure* and *style* as potential levels of textual similarity.

Neural Networks. One major disadvantage of using purely statistical approaches lies in the binary representation of text similarity, i.e. a word that is spelled differently is also treated as a different word, no matter the semantic closeness. When using neural networks for finding text reuse, approaches can be used that give continuous similarity values. Our research did hardly find any previous work regarding neural networks explicitly for intertextuality, whereas several approaches are aiming at similar problems while not being applied in the broader field of intertextuality in digital humanities.

¹<https://corpora.uni-leipzig.de>

An approach that uses alignments is presented by [6]. The authors use a pre-processed corpus and try to find text reuse for different input queries (=the actual quotes) using a modified Needleman-Wunsch operating on word embeddings. An approach that also uses word vectors to find similar sentences for a given query sentence is presented by [12]. When comparing not just single words, but rather full sentences, it is necessary to respect the order of words in a sentence. Recurrent neural networks (RNN) can be used for this, as has been demonstrated by [13].

In general, when trying to find similarities between objects, neural networks are trained in an unsupervised manner. In image similarity scenarios, a siamese architecture can be used to create a semantic vector representation of data, meaning that data with similar features will lead to a similar vector representation. To our knowledge, the earliest usage of a siamese architecture is documented in [5]: In their project, the authors wanted to calculate the similarity between signatures. The signatures have been given into the neural network by a set of different predefined features. In this case, a positive pair contained two signatures from the same person. Negative signatures were two signatures from two different people. The last layer's output was linear, and the vector representation of the two signatures was compared using cosine similarity. [8] created another siamese approach: In their work, they present a convolutional neural network with a linear output. During training, the network is trained to give face images from the same person a cosine similarity of 1. When the images came from two different people, the cosine similarity should be 0. Another siamese approach was created by [17]. Here, a sentence was put token-wise into an LSTM, the last hidden state of that layer was then used to calculate the Manhattan distance to another sentence's vector on a pre-labeled data set. [20] created a similar approach: They used the output sequence of a pre-trained BERT model [9] and added a pooling layer to get a single vector representation of that sequence. The network then was trained by giving positive and negative sentence pairs into the network, similar to the previously mentioned approaches.

3. Method

Building on the above related work, we suggest a novel two-step filtering approach to find quotations of Shakespeare's Hamlet in postmodern fictional literature. We use a neural network to create vector representations of sentence parts, both from Hamlet and our target corpus. These are then compared to each other to find the most similar sentence parts. Since the number of comparisons would be extremely high if we were to compare all Hamlet sentences to all the sentences in our target corpus of 31 books, a major challenge was to lower the number of actual comparisons, which ultimately will also result in a lower number of false positives: While each of Shakespeare's plays and poems has an undisputed literary quality, not every line is equally suitable for quoting. For example, a sentence like *Beware the ides of March* is rather famous, and it would be interesting to know where this quote is reused. However, a sentence like *I am a man* is a common idiomatic phrase that is neither uniquely Shakespearean nor will it be read as a quotation of Shakespeare. *I am a man* in a contemporary text does not evoke a connection of an older text, as it is just a prevalent and frequent phrase that occurs in many different texts (including Shakespeare's works and earlier or later texts) and would thus result in a false positive.

This section describes the steps and methods used to find quotes of Shakespeare's Hamlet in a small corpus of postmodern fictional literature. First, we describe the materials used.

After that, we explain the two neural networks we used in the two different steps of our NLP pipeline. The order in which the two networks are described resembles the order in which they are used in practice: First, we describe the training of a classifier that allows us to pre-filter the corpus of novels, i.e. to filter out sentences that are likely to not contain any traces of Hamlet. Second, we describe an approach to train a neural network in an unsupervised manner in order to find semantically similar sentences for the remaining sentences of the corpus. The individual steps of the proposed pipeline can be found in Figure 1.

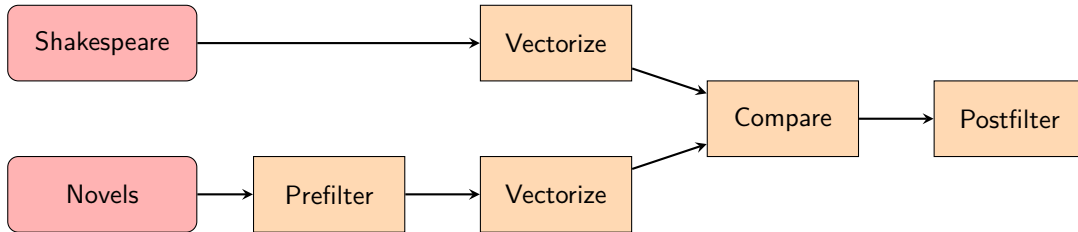


Figure 1: NLP pipeline for finding Shakespeare quotes.

3.1. Language resources

For this paper, we decided to focus on Shakespeare’s most popular tragedy: *Hamlet - The Tragedy of Hamlet, Prince of Denmark*. Our comparison corpus is a collection of 31 novels of postmodern fiction (see Table A.1), with a total size of 286,165 sentences. We decided to use this corpus because of the availability of a ground truth of 129 Hamlet quotes for these books, which were gathered by a Shakespeare expert by a mixture of close and distant reading [16]. This ground truth allows us to evaluate our approach (for more details see the *results* section).

3.2. Filtering candidate quotes

When we are looking for quotes from Shakespeare’s Hamlet in the presented corpus, we assume that a potential quote uses register, vocabulary and style that resembles Shakespeare’s way of writing. In order to pre-emptively discard sentences that – with a high probability – do not contain a Shakespeare quote, we use a neural network that classifies sentences or sentence parts by their quote potential. For this, we are using a list of predefined Shakespeare quotes. We used a quote list created by Oxford scholars [19] and combined it with a crowdsourced list of Wikiquotes². Since our classifier also needs to be trained with negative examples, we chose to also include data from the same time era by using an early modern drama data set³. Since Hamlet is a drama, we also included a data set of movie subtitles [23], which are also based on dialogue and consequently more similar to stage plays than contemporary prose. To widen the classifier’s knowledge about text, we also included a data set of contemporary books⁴ and news articles⁵ into our data set.

The classifier’s neural network architecture is straight-forward, as it comprises an embedding layer for word input, a recurrent layer that receives the token sequences, and a final feed-forward

²https://en.wikiquote.org/wiki/William_Shakespeare

³<https://graphics.cs.wisc.edu/WP/vep/vep-early-modern-drama-collection/>

⁴<https://www.gutenberg.org/>

⁵<https://corpora.uni-leipzig.de/>

layer with a single output with sigmoid activation. Some preliminary tests have shown that the classifier works best when receiving full sentences as input, although this leads to undesired side effects: The classifier can quickly learn that a sentence is non-Shakespearean if it contains at least one word that does never appear in the Shakespeare quote list. Thus, when using the classifier on sentences to find potential quotes, many sentences would be discarded, although parts of it may contain a quote. To circumvent this problem, we decided to train the classifier on n-grams of lengths 5, 7, 9, 11, and 13. If a longer sentence contains at least a smaller part that is a potential quote, the sentence will not be discarded. Given the rather small number of lines (8,855) that make up Shakespeare’s Hamlet, the classifier quickly learned to identify all the quotes we used in the training phase. The problem was that the classifier only found exact verbatim quotes and all variations were discarded, even if it was only a minor change compared to the original quote. To counter this problem, we added input noise during the training by randomly masking words from the input sequence, i.e. words have been replaced randomly by an empty placeholder. During training, the classifier now learns that a given sequence can be a valid quote, even if specific keywords are not present. The effect is that when using the classifier on non-Shakespearean data, not only verbatim quotes but also variations of quotes are classified as quote candidates.

The number of ngrams for all the 31 novels can be found in Table 1. For each n-gram, our classifier computes a probability that a given n-gram is likely to be an actual Shakespeare quote. For n-grams with length 5, the threshold was set to 0.8, since the shorter n-grams often are included in the longer n-grams. We also noticed that n-grams of that length amounted to a large number of false positives. For the other lengths, we decided to only keep those n-grams with a probability higher than 0.3. We decided to only use such a low threshold to keep true positive matches while at the same time discarding roughly 90% of all of the other ngrams.

3.3. Finding similar sentences

To find reused parts of Shakespeare’s Hamlet in postmodern fiction, we had to find a sentence representation that makes actual Hamlet quotes and candidate quote sentences comparable. We decided to use a vector representation created by a neural network. We did this because there may be slightly different words used, compared to a quote in a source by a different author. An example would be *Or that the Everlasting had not ...* (Shakespeare: Hamlet) as compared to *Or that God had not ...* (Fforde, Jasper: Thursday Next Book 4: Something Rotten). We assume that a neural network will be able to learn that these words are semantically very closely related. Another reason against using a purely statistical approach is that modern texts contain many words that have not been in used in Shakespeare’s time so that the number of exactly overlapping n-grams [14] is lower compared to the comparison of texts from the same era or even the same topic.

The neural network to be used will find sentence pairs between two corpora. The previously mentioned encoder-decoder approach by [13] is able to do exactly that. However, we think that having a decoder structure for textual data is too complex memory-wise, since generating text needs a weight matrix with one axis being the size of the used dictionary. We circumvent this problem by only using an encoder structure and by proposing a siamese neural network to create vector representations of both the known quotes and the candidate quotes. Such an architecture is used to create a semantic vector representation of data, meaning that data with similar features will lead to a similar vector representation. We decided to use a custom model rather than relying on existing, pre-trained models such as BERT [9] for several reasons. On

Table 1

The number of snippets for each snippet length before and after filtering.

ngram length	number of ngrams	number of ngrams after filtering
5	1,729,519	237,690
7	1,480,019	359,216
9	1,262,438	101,875
11	1,078,270	38,872
13	921,433	5,284
Sum	6,471,679	742,937

the one hand, the BERT model has to be extended since it is not explicitly trained on creating vector representations of whole sentences. Instead, the output of the BERT model is a sequence of vector representations for all tokens of the input sequence. A vector representation for the whole sentence is created by calculating the mean of all the tokens vectors. On the other hand, the BERT model consists of 12 layers and creates a vector output of 768 features, and is trained on full sentences. We feel a model of that size is unnecessarily complex for our use case since our data consists only of short token sequences with a small variety of used topics. Similar to the necessary extension of the BERT model [20], we created a neural network that is giving a sentence token-wise into a bidirectional recurrent layer. The recurrent layer returns a vector for each time step of the sequence of shape $length \times features$. We chose to use a feature size of 256, which is three times smaller than the number of features the BERT model uses. We sum that output on the length axis, resulting in a single vector representation of the sequence.

When training the network, sentences are inserted into the network as positive and negative pairs. A positive pair of sentences consists of two coherent sentences from the same document. A negative sentence pair, on the other hand, contains two randomly chosen sentences. For each of the pair’s sentences, the model had to create a vector representation. The vector similarity of that pair should be 1 for positive pairs and 0 for negative pairs. The similarity was calculated using the cosine similarity. To get training data, we used the Leipzig Corpora Collection⁶, which provides large corpora in various languages. We used an English news corpus, which contains news articles about various topics with an average length of 20 tokens per sentence. A positive training pair for our siamese architecture was a sentence pair from the same news article; a negative training pair was a sentence pair from different articles. The idea is similar to the one presented by [13], where training pairs are coherent book sentences. To train our network also on Shakespearean words, we extended the corpus by Shakespearean texts. The fact that the network is trained with (weak) labels makes the training of the network supervised. However, creating vector representations that result in close or distant similarity scores for the shown positive and negative pairs is an auxiliary task. The resulting vector representation of input sequences is not pre-defined, thus this part of the training is unsupervised. The neural network was trained with a batch size of 16. Each batch contained two input sequences of sentences. The maximum sequence length was 32. If a sentence’s sequence was longer than 32, a random subset was chosen. The sequence pairs were either coherent sentences from the same news article or random sentences from the same dataset, i.e. negative and positive pairs have been shown with the same frequency. For positive pairs, the target similarity value was 1, for

⁶<https://corpora.uni-leipzig.de/>

negative pairs 0. Mean squared average was used as loss function. The model was trained for 12 hours and was created using the following layers from the *Keras* library:

```
model = Sequential()  
model.add(Embedding(input_dim=50000, output_dim=256))  
model.add(Masking())  
model.add(Bidirectional(GRU(256, return_sequences=True)))
```

Using these vector representations, we can pair each Hamlet n-gram with each candidate n-gram, whereas only parts of the same length are compared. In a final step, we compared the results by the tokens that appear in each sentence part. We created a set of tokens appearing in each part as well as in both parts and the share of each part set with the combined set using intersection over union. Thus, if a pair has received a high vector similarity but hardly shares common tokens, we discarded it. The result list was sorted by the average of the two similarity scores. We have cut the list after the last occurrence of a found known quote. To reduce the number of entries in the list, we removed duplicate entries: Since each sentence in our postmodern fictional literature dataset was split into several n-grams, sometimes more than one n-gram was paired with the same Hamlet quote. In those cases, we only kept the pair with the highest vector similarity score.

4. Results and discussion

This section presents the results we obtained with our approach on a sample corpus of 31 books (see section 3.1). After executing our two-step pipeline, we retrieved a list with pairs of Hamlet quotes and potential candidate quotes from our corpus of postmodern fictional literature. The pairs are sorted by the average of their vector and token similarity, with a score that ranges from 0 - 100. Of the 129 known quotes from the ground truth, we found a total of 92 in our results list. Among the 92 true positives, our approach also found a number of false positives. We cut off the list of results after the last true positive match, which results in a list with 1897 items. In order to be able to better discuss the list of results, we split it into three parts (see Figure 2). Also, for each index for which we find a true positive, we calculate the precision up to that index position. After that, we calculate the *average precision* of all the precision values. Using that evaluation measure, ranking true-positives high in the list of results correlates with a high average precision.

Part 1 – The first part from index 1 to 500 contains only verbatim or near-verbatim quotes. It contains 75 true positives, resulting in an average precision of 0.55. At the top of the list, there are quotes that are full or nearly full-verbatim quotes. An example for a true positive would be *how the wheel becomes it* (Stephen Fry: *Paperweight*). The lower the similarity score, the more the pairs differ. For example, the pair *what a piece of work is a man* (Terry Pratchett: *The Amazing Maurice and his Educated Rodents*) and *what a wonderful piece of work is man* (*Hamlet*) has received similarity of 0.99. In this part of the list, there were also true-positive quotes that have not been part of the ground truth but have been added to it subsequently. For example, the phrase *To sleep, perchance to dream* in *Hamlet* is used, which has been rephrased to *To Die, Perchance to Sleep?* by Jasper Fforde in *Early Riser: The new standalone*. Another newly found reference is a slight variation of *The time is out of joint*. In *Wyrd Sisters* by Terry Pratchett it is used: *time was out of joint*.

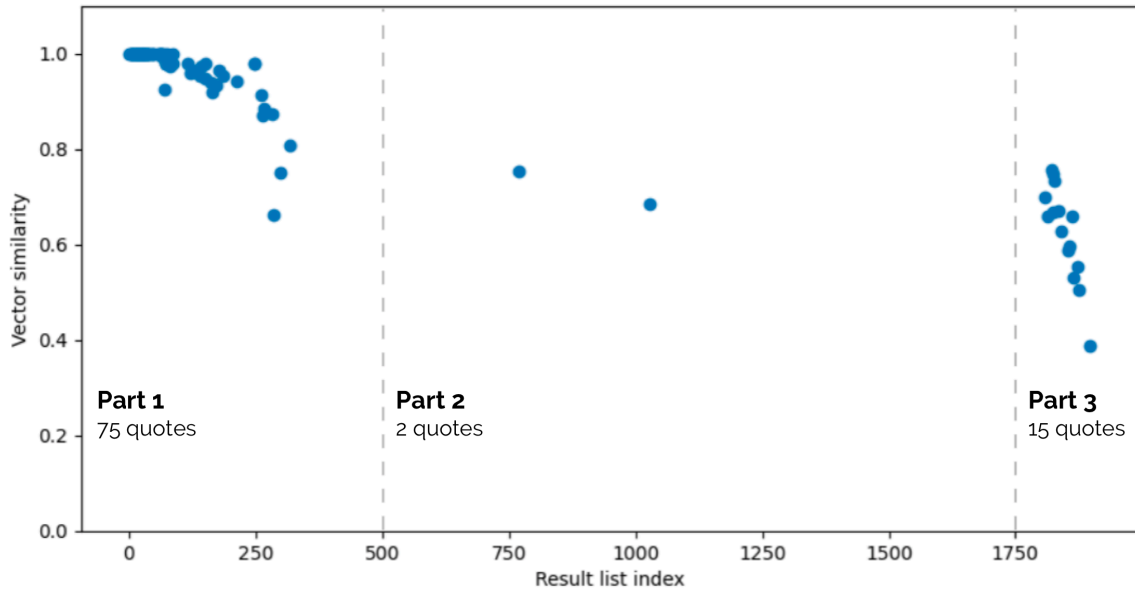


Figure 2: Vector similarity and result list index of the 92 found known quotes.

Part 2 – The second part of the list from index 501 to 1750 contains pairs that differ in some stop words and have a high vector similarity. Although this part is considerably longer, it only contains 2 true positives, the average precision from index 1 to 1750 is 0.54. This part of our result list also contains many false positive pairs of two kinds. The one kind of pairs contains many overlapping words, but these are in a different order. The other kind contains words in the same position of the n-grams, but the not-overlapping words create a very different meaning. An exemplary pair would be *catch the conscience of the* and *with the blood of the*: Although the second, fourth, and fifth words are identical, the meaning of the two n-grams differs significantly. A true-positive example from that part with a vector similarity of 0.75 is *i ll catch the conscience of the king* paired with *William Shatner in Conscience of the King*. The two n-grams contain a significant overlap but both start with different words, which leads to a low vector similarity. This shows that the vector similarity is not based on the aspects we are interested in.

Part 3 – The more interesting pairs are found in the third part of our result list starting from index 1751 to 1897. In this last part, there are 15 true positives, resulting in an average precision of the whole list of 0.46. The pairs in this part overall have lower similarity scores, among which we find quite interesting pairs such as *to be or not to be that is the question* and *to espresso or to latte that is the question*, which is a true positive match. Comparing this pair to the pairs of the second part in our result list shows that the vector similarity at times focuses on aspects we are not interested in from the perspective of intertextuality. For example, n-grams pairs containing word word groups like *of the* or *in the* account for a large part of our false-positive findings.

37 of the 129 ground truth quotes were either not found by our approach or ranked with a score so low that they were lost among the other low-ranked false positives. Apart from quotes that have been filtered out in the pre-filtering step of our pipeline, other quotes we failed to

identify fall into the following problem categories:

- A One-word quotes, which are too short to be detected automatically and that demand a lot of context (for examples see Table A.2, lines 1-3)
- B Quotes with long insertions or acutely different word order (lines 4-6)
- C Quotes that are just too implicit, too altered or non-specific (lines 7-9)
- D Quotes that contain spelling errors or deviations. One line contained intra-word punctuation (line 10), one was a play on homophones (line 11), another contained a typographical error in the contemporary text (line 12)
- E Miscellaneous quotes, where we see no obvious reason why they have not been found or why they were scored too low (lines 13-15)

5. Conclusion

The search for reused passages of Shakespeare’s texts at first sight might look like a standard task for existing text similarity detection approaches. However, finding pairs that are reused in the sense of an intertextual reference can be rather challenging, as automated searches for these are bound to produce many false positives. In this article, we described a two-step process to address this problem partially. We found that a critical element is the pre-filtering of potential quote candidate sentences, to reduce the number of false positives (as compared to previous studies by [7] and [16]). Without this pre-filtering, the number of pairs that need to be compared would have been extremely high. In the second filtering step, we compared potential pairs by their vector representation using a neural network which was trained without supervision.

We found that using the vector similarity of two different n-grams leads to undesired results. Our list of results contains pairs of n-grams that are lexically similar, but not all repetitions of words are actual intertextual references. This shows that the learned vector representation focuses on features that result in unfavorable similarity scores for our use case. For example, if an n-gram pair contains stopwords in a different order, we would classify this as a false positive and would like this pair to get a lower similarity score. Using our existing list of results, we plan to use the true positive pairs to identify changes in the vector space for future work. We also work with the false positives we found to identify undesired changes in the vector space. With that knowledge, we plan to modify the neural networks to deliver less false positives.

Although our approach did not find all the quotes that were defined as the ground truth by a Shakespeare expert, it did manage to identify 5 new quotes, that were not part of the initial ground truth, but that were verified ex post. While this means that we have to critically reflect on the function and requirements for ground truth data in the context of intertextuality, it also shows that neither man nor machine are reliably able to detect all possible intertextual references in a text, but rather a combined approach is most viable. Besides, a human expert is needed in any case, to interpret the automatically generated results lists, which will always contain false positives obvious to the expert but indistinguishable for the automated method. With this in mind, we will also look into providing a more interactive tool that allows scholars to effectively sieve through the results. We plan to realize a tool that is similar to a search engine (see [6] for an example interface), i.e. all potential results will be displayed in a graphical user interface that supports the verification of any of the results by close reading the underlying full texts.

References

- [1] G. Allen. *Intertextuality*. Psychology Press, 2000.
- [2] D. Bamman and G. Crane. “The logic and discovery of textual allusion”. In: *In Proceedings of the 2008 LREC Workshop on Language Technology for Cultural Heritage Data*. 2008.
- [3] D. Bär, T. Zesch, and I. Gurevych. “Text Reuse Detection using a Composition of Text Similarity Measures”. In: *Proceedings of COLING 2012*. Mumbai, India: The COLING 2012 Organizing Committee, Dec. 2012, pp. 167–184. URL: <https://www.aclweb.org/anthology/C12-1011>.
- [4] M. Berti et al. “Measuring the Influence of a Work by Text Reuse”. In: *The Digital Classicist 2013* (2013), pp. 63–79.
- [5] J. Bromley et al. “Signature Verification Using a ”Siamese” Time Delay Neural Network”. In: *Proceedings of the 6th International Conference on Neural Information Processing Systems*. NIPS’93. Denver, Colorado: Morgan Kaufmann Publishers Inc., 1993, pp. 737–744. URL: <http://dl.acm.org/citation.cfm?id=2987189.2987282>.
- [6] M. Burghardt and B. Liebl. “”The Vectorian - Eine parametrisierbare Suchmaschine für intertextuelle Referenzen””. In: Book of Abstracts, DHd, 2020.
- [7] M. Burghardt et al. *“The Bard meets the Doctor” – Computergestützte Identifikation intertextueller Shakespearebezüge in der Science Fiction-Serie Dr. Who*. Book of Abstracts, DHd, 2019.
- [8] S. Chopra, R. Hadsell, and Y. Lecun. “Learning a similarity metric discriminatively, with application to face verification”. In: *Proc. Computer Vision and Pattern Recognition*. Vol. 1. July 2005, 539–546 vol. 1. ISBN: 0-7695-2372-2. DOI: 10.1109/CVPR.2005.202.
- [9] J. Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *CoRR* abs/1810.04805 (2018). arXiv: 1810.04805. URL: <http://arxiv.org/abs/1810.04805>.
- [10] C. Forstall et al. “Modeling the scholars: Detecting intertextuality through enhanced word-level n-gram matching”. In: vol. 30(4). 2015, pp. 503–515.
- [11] R. Hohl Trillini. *Casual Shakespeare: Three centuries of verbal echoes*. Routledge, 2018, pp. 1–199. DOI: 10.4324/9781351120944.
- [12] T. Kenter and M. de Rijke. “Short Text Similarity with Word Embeddings”. In: *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. CIKM ’15. Melbourne, Australia: ACM, 2015, pp. 1411–1420. ISBN: 978-1-4503-3794-6. DOI: 10.1145/2806416.2806475. URL: <http://doi.acm.org/10.1145/2806416.2806475>.
- [13] R. Kiros et al. “Skip-Thought Vectors”. In: *CoRR* abs/1506.06726 (2015). arXiv: 1506.06726. URL: <http://arxiv.org/abs/1506.06726>.
- [14] C. Lyon, J. Malcolm, and B. Dickerson. “Detecting Short Passages of Similar Text in Large Document”. In: (Dec. 2001).
- [15] J. Maxwell and K. Rumbold. *Shakespeare and Quotation*. Cambridge University Press, 2018. DOI: 10.1017/9781316460795.

- [16] J. Molz. “A close and distant reading of Shakespearean intertextuality”. July 2019. URL: <http://nbn-resolving.de/urn:nbn:de:bvb:19-261274>.
- [17] J. Mueller and A. Thyagarajan. “Siamese Recurrent Architectures for Learning Sentence Similarity.” In: *AAAI*. Ed. by D. Schuurmans and M. P. Wellman. AAAI Press, 2016, pp. 2786–2792. URL: <http://dblp.uni-trier.de/db/conf/aaai/aaai2016.html#MuellerT16>.
- [18] M. Potthast et al. “Overview of the 5th International Competition on Plagiarism Detection”. In: *Working Notes Papers of the CLEF 2013 Evaluation Labs*. Ed. by P. Forner, R. Navigli, and D. Tufis. Sept. 2013. ISBN: 978-88-904810-3-1. URL: <http://www.clef-initiative.eu/publication/working-notes>.
- [19] S. Ratcliffe. *Oxford Essential Quotations*. Oxford University Press, 2012.
- [20] N. Reimers and I. Gurevych. *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks*. 2019. arXiv: 1908.10084 [cs.CL].
- [21] A. Rzyman. *The Intertextuality of Terry Pratchett’s Discworld as a Major Challenge for the Translator*. Cambridge Scholars Publishing, 2017.
- [22] D. Smith, R. Cordell, and E. Dillon. “Infectious texts: Modeling text reuse in nineteenth-century newspapers”. In: *2013 IEEE International Conference on Big Data (2013)*, pp. 86–94.
- [23] J. Tiedemann. “Parallel Data, Tools and Interfaces in OPUS.” In: *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC 2012)*. Vol. 30(4). 2012.
- [24] P. Vierthaler and M. Gelein. “A BLAST-based, Language-agnostic Text Reuse Algorithm with a MARKUS Implementation and Sequence Alignment Optimized for Large Chinese Corpora”. In: *Journal of Cultural Analytics* (Mar. 22, 2019). DOI: 10.22148/16.034.
- [25] Q. Zhang et al. “Continuous Word Embeddings for Detecting Local Text Reuses at the Semantic Level”. In: *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*. SIGIR ’14. Gold Coast, Queensland, Australia: Association for Computing Machinery, 2014, pp. 797–806. ISBN: 9781450322577. DOI: 10.1145/2600428.2609597. URL: <https://doi.org/10.1145/2600428.2609597>.

Table A.1

Corpus of postmodern fiction novels that will be used to test our automatic approach for text reuse detection of Shakespeare's Hamlet.

Title	Author	#Sentences
Adams, Douglas	H2G2 The Ultimate Hitchhiker's Guide	21,289
Barnes, Julian	The Noise of Time	3,208
Carter, Angela	Nights At The Circus	5,883
Carter, Angela	Shaking A Leg	15,433
Carter, Angela	Wise Children	5,927
Fforde, Jasper	Early Riser: The new standalone	9,654
Fforde, Jasper	Lost in a Good books	9,258
Fforde, Jasper	One of our Thursdays is Missing	7,640
Fforde, Jasper	Something Rotten	9,201
Fforde, Jasper	The Eyre Affair	8,997
Fforde, Jasper	The Well Of Lost Plots	9,305
Fforde, Jasper	The Woman Who Died a Lot	7,746
Fry, Stephen	Making History	12,608
Fry, Stephen	Moab Is My Washpot	6,951
Fry, Stephen	Paperweight	7,631
Fry, Stephen	The Hippopotamus	7,741
Fry, Stephen	The Liar	8,280
Gaiman, Neil	Anansi Boys	10,778
Gaiman, Neil	Trigger Warning	9,772
Pratchett, Terry	Guards! Guards!	9,649
Pratchett, Terry	Nation	9,084
Pratchett, Terry	Night Watch	11,317
Pratchett, Terry	The Amazing Maurice and His Educated Rodents	7,412
Pratchett, Terry	Wyrd Sisters	8,315
Pratchett, Terry; et al.	The Science of Discworld 2	8,009
Rushdie, Salman	East, West	2,597
Rushdie, Salman	Imaginary Homelands	6,078
Rushdie, Salman	Joseph Anton	12,551
Rushdie, Salman	The Ground Beneath Her Feet	13,006
Rushdie, Salman	The Moor's Last Sigh	8,214
Smith, Zadie	On Beauty	12,631

Table A.2

	Novel	Intertextual reference	Original Shakespeare passage	Problem category
1	Fforde, Jasper - <i>Thursday Next Book 4: Something Rotten</i>	Does " pirate " have one "t" or two?	Ere we were two days old at sea, a pirate of	A
2	Fforde, Jasper - <i>Thursday Next Book 4: Something Rotten</i>	So he then screamed abuse at her for five minutes, told her she was a whore and marched out, muttering something about being a chameleon .	Excellent, i' faith; of the chameleon's dish. I eat the air, promise-cramm'd. You cannot feed capons so.	A
3	Pratchett, Terry, Ian Stewart and Jack Cohen - <i>The Science of Discworld II: The Globe</i>	'A fishmonger was involved.	You are a fishmonger .	A
4	Adams, Douglas - <i>The Hitchhiker's Guide to the Galaxy</i>	And the rest , after a sudden wet thud, was silence .	So tell him, with th' occurments, more and less, which have solicited- the rest is silence . Dies.	B
5	Fforde, Jasper - <i>Thursday Next Book 4: Something Rotten</i>	'— in the ear? ' said Joffy as I walked into the kitchen. 'Does that work?'	Anon comes in a fellow, takes off his crown, kisses it, pours poison in the sleeper's ears , and leaves him.	B
6	Fforde, Jasper - <i>Thursday Next Book 4: Something Rotten</i>	' Whether 'tis tastier on the palette to choose white mocha over plain,' he continued in a rapid garble, ' or to take a cup to go.	Whether 'tis nobler in the mind to suffer The slings and arrows of outrageous fortune Or to take arms against a sea of troubles, And by opposing end them.	B
7	Rushdie, Salman - <i>East, West</i>	Thus haste, enforced by our inevitable end, makes Yoricks of us all)	Thus conscience does make cowards of us all ,	C
8	Pratchett, Terry - <i>Wyrd Sisters</i>	avenge the terror of [his] father's death . P222	The collocation 'father's death' appears 8 times in Hamlet: "my father's death" is used by Laertes once, thrice by Hamlet himself; "his father's death," "your father's death," and "your dear father's death," is used by Claudius and Gertrude once speaks of "her father's death."	C
9	Fforde, Jasper - <i>Thursday Next Book 4: Something Rotten</i>	' Oh God, oh God!	O God! God!	C
10	Rushdie, Salman - <i>East, West</i>	Methought there was nothing meet .	sings. In youth when I did love, did love, Methought it was very sweet; To contract-O- the time for- a- my behove, O, methought therea- was nothing- a- meet .	D
11	Carter, Angela - <i>Wise Children</i>	We sported bellhop costumes for our Hamlet skit; should, we pondered in unison and song, the package be delivered to, I kid you not, ' 2b or not 2b '	To be, or not to be- that is the question:	D
12	Carter, Angela - <i>Shaking a Leg</i>	The notion of sickness is related to that indefinable ' something rotten in the state '.	Something is rotten in the state of Denmark.	D
13	Carter, Angela - <i>Nights at the Circus</i>	As the baboushka slept, her too, too solid kitchen fell into pieces under the blows of their disorder as if it had been, all the time, an ingenious prop, and the purple Petersburg night inserted jagged wedges into the walls around the table on which these comedians cavorted with such little pleasure, in a dance which could have invoked the end of the world.	O that this too too solid flesh would melt, thaw, and resolve itself into a dew!	E
14	Pratchett, Terry - <i>Guards, Guards</i>	Sometimes you have to be cruel to be kind	I must be cruel , only to be kind	E
15	Smith, Zadie - <i>On Beauty</i>	There was only this one high note – the rest was silence . 127	So tell him, with th' occurments, more and less, Which have solicited- the rest is silence . Dies.	E