Story Fragment Stitching: The Case of the Story of Moses

Mohammed Aldawsari[1] , **Ehsaneddin Asgari**[2] , **Mark A. Finlayson**[3]
[1,3]Florida International University
[2]University of California, Berkeley
asgari@berkeley.edu, {malda021,markaf}@fiu.edu

Abstract

We introduce the task of *story fragment stitching*, which is the process of automatically aligning and merging event sequences of partial tellings of a story (i.e., *story fragments*). We assume that each fragment contains at least one event from the story of interest, and that every fragment shares at least one event with another fragment. We propose a graph-based unsupervised approach to solving this problem in which events mentions are represented as nodes in the graph, and the graph is compressed using a variant of model merging to combine nodes. The goal is for each node in the final graph to contain only coreferent event mentions. To find coreferent events, we use BERT contextualized embedding in conjunction with a *tf-idf* vector representation. Constraints on the merge compression preserve the overall timeline of the story, and the final graph represents the full story timeline. We evaluate our approach using a new annotated corpus of the partial tellings of the story of Moses found in the Quran, which we release for public use. Our approach achieves a performance of 0.63 F_1 score.

1 Introduction

Understanding stories is a long-held goal of both artificial intelligence and natural language processing [Charniak, 1972; Schank and Abelson, 1977; Wilensky, 1978; Dyer, 1983; Riloff, 1999; Frank *et al.*, 2003; Mueller, 2007; Winston, 2014]. Stories are found throughout our daily lives, e.g., in news, entertainment, education, religion, and many other domains. Automatically understanding stories implicates many interesting natural language processing tasks, and much information can be extracted from stories, including concrete facts about specific events, people, and things, commonsense knowledge about the world, and cultural knowledge about the societies in which we live. One interesting and challenging task which has not yet been solved is what we call here *story fragment stitching*. In this task we seek to merge partial tellings of a story—where each partial telling contains part of the sequence of events of a story, perhaps from different points of view, and may be found across different sources or media—into one coherent narrative which may then be used as the basis for further processing. Conceptually, this task is similar to both cross-document event coreference (CDEC) and event ordering in NLP. However, story fragment stitching, as we define it, presents a more challenging problem for at least two reasons. First, and unlike event coreference, the overall timeline of the story's events need to be preserved across all fragments. Second, and unlike event ordering which targets only events related to a single entity, this work considers all events across all fragments.

For the purposes of this work, we define a story as a sequence of events effected by characters and presented in a discourse. This is in accord with fairly standard definitions: for example, [Forster, 1927] said that "A story is a narrative of events arranged in their time sequence." As a simplifying assumption, we additionally assume that the events in the story are presented in the chronological order in which the events of a story take place (i.e., the *fabula* time order) [Bordwell, 2007]. We leave the problem of extracting the chronological ordering of events within a text for other work.

We present an approach to story fragment stitching problem inspired by [Finlayson, 2016] which in turn based on model merging, a regular grammar learning algorithm [Stolcke and Omohundro, 1993], using similarity measures based on BERT contextualized embedding and *tf-idf* weights of events and their arguments. We apply this approach to a concrete example of this problem, namely, the story of the prophet Moses as found in the Quran, the Islamic holy book. The story of Moses is not found in one single telling in the Quran; rather, it is found in eight fragments spread across six different chapters (the chapters of the Quran are called *suras*), with the story comprising 7,931 total words across 283 verses of anywhere from 2 to 94 words in length. In this work we demonstrated our approach using the seven fragments with coherent timelines.

The story of Moses is especially useful for this work because it has been subject to detailed event analysis, in particular, [Ghanbari and Ghanbari, 2008] identified a canoni-

In: A. Jorge, R. Campos, A. Jatowt, A. Aizawa (eds.): Proceedings of the first AI4Narratives Workshop, Yokohama, Japan, January 2021, published at http://ceur-ws.org

cal timeline of events for the story. Further, the Quran verse structure provides a natural unit of analysis, where nearly every verse is related to only a single event in the story timeline. We manually extracted 573 event mentions from 273 verses (omitting 11, as described later) and annotated all events corresponding to the Ghanbari's event categories. We used this data to test our approach, resulting in a proof of concept of story fragment stitching. We release both our code and data to enable reimplementations[1].

We begin by discussing prior work on cross-document event coreference, event ordering, as well as the description and analysis of story structure (§2). Then we introduce our method, including the task definition (§3.1) and specific aspects of our approach (§3.2–§3.3). We then describe our evaluation, including construction of the gold standard for the Moses story in the Quran (§4.1), the experiment setup (§4.2), the result of our model (§4.5), as well as an error analysis (§5). We conclude with a list of contributions (§6).

2 Related Work

The most closely related problems to story stitching are the problem of cross-document event coreference (CDEC) and cross-document event ordering. In CDEC systems, the goal is to group expressions that refer to the same event across multiple documents. [Bagga and Baldwin, 1999; Lee *et al.*, 2012; Goyal *et al.*, 2013; Saquete and Navarro-Colorado, 2017; Kenyon-Dean *et al.*, 2018; Barhom *et al.*, 2019]. In event ordering task, which was introduced in SemEval-2015 [Minard *et al.*, 2015], the goal is to order events cross-document in which a specific target entity is involved. That is, a system should produce a timeline for a specific target entity and that timeline consists of the ordered list of the events in which that entity participates. Similarly, within document event sequence detection task, which was introduced in TAC KBP 2017 event track [Mitamura *et al.*, 2017], aims to identify event sequence (i.e., after links) that occurs in a script [Schank and Abelson, 1977].

Despite this very interesting and useful prior work, these systems are not directly applicable to the task of story fragment stitching as we define it. In particular, CDEC systems ignore the timeline of the story's events (i.e., the overall timeline of the story's events is not guaranteed to be preserved across all fragments), while event ordering systems only order certain events related to a specific target.

Researchers have explored several ways of assessing similarity between stories [Schank and Abelson, 1975; Roth and Frank, 2012; Finlayson, 2012; Iyyer *et al.*, 2016; Nikolentzos *et al.*, 2017; Chaturvedi *et al.*, 2018]. These works provided valuable ways to capture similarity between stories. However, the story similarity task is not directly applicable to the task of stitching fragmented stories, where the goal is to order events across multiple stories (fragments), except in the simple baseline sequence alignment approach [Needleman and Wunsch, 1970; Reiter, 2014].

[1]The code and data are available at https://doi.org/10.34703/gzx1-9v95/28GC2M

3 Approach

We now discuss the precise definition of the story fragment stitching task (§3.1) and the details of the two main components of our approach: model formulation (§3.2), and the graph merge to align fragments events into a full, ordered, end-to-end list of story events (§3.3).

3.1 Task

We define the goal of story fragment stitching as: align a set of story fragments into a full, ordered, end-to-end list of story events. We assume that the story fragments are ordered lists of events, where the order is that of the fabula, namely the order of events as they happen in the story world. In many stories, the fabula order is different from the discourse order, but we do not consider this case here; we leave the problem of extracting the chronological order of events to other work. We also assume that each fragment shares at least one event with another fragment. The output of the system is an ordered list of nodes, where each node is a collection of event mentions (corefering events) that all describe one particular single event, and these nodes are in the same order as the overall fabula.

3.2 Model Formulation

The first step of the approach is model initialization which is shown Algorithm 1 lines 1–3. Using the function `constructLinearBranch`, we convert each fragment's list of events into a linear directed graph (linear branch) where each node contains only a single event. Each event is represented by a vector which is a concatenation of the event contextualized embedding from the BERT model and *tf-idf* weights of the event lemma and its semantic arguments. *BERT* [Devlin *et al.*, 2018] is a multi-layer bidirectional transformer trained on plain text for masked word prediction and next sentence prediction tasks, while *tf-idf* is the standard term weighting approach to reflect how important a word is in a document in comparison to the rest of documents [Salton and McGill, 1986]. Using the function `linkGraphs` we link all linear branches to a start and an end node, resulting in one directed graph of all the set of fragments, as shown in Figure 1.

This initial model will be used to generate possible solutions by merging different nodes on the basis of a similarity measure, discussed below. When two nodes A and B are merged, the new node C should contain an average vector of both A and B. In the next section, we introduce the merge approach.

3.3 Graph Merge

The second step of the approach is model merging, shown in Algorithm 1 lines 4–15. We first compute a threshold α using `computeTFIDFAvgSim` function, which takes the average of the highest and lowest cosine similarity values between all fragments using *tf-idf* weights. α sets the minimum similarity required to merge two nodes; for our data α was 0.39. Next, using a cosine similarity measure, the `computeNodesSim` function computes the full set of similarity scores between all pairs of nodes. Then the algorithm starts by searching for

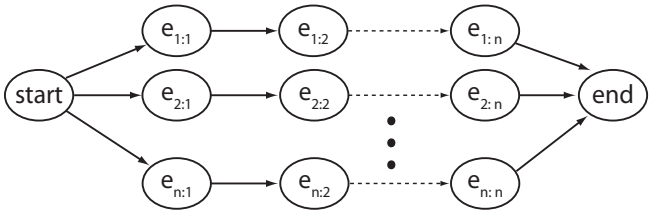

Figure 1: The initial model constructed using Moses story fragments. Each node represents an event's vector. Each fragment generates its own linear branch running from the start node to the end node where i in $e_{i:j}$ represents the fragment's number and j represents the event's number.

the most similar nodes using `findMostSim` function, lines 6 and 13, and merges the most similar nodes using `merge` function. Because the fragments are assumed to be already in fabula time order, the `pairsIntroduceCycle` boolean function disallows merges that would introduce cycles, ignoring (and removing) self-loops (the *no-cycles* constraint), and thereby preserves the overall order of the events. Note that disallowing cycles also prevents merges of non-neighboring nodes within the same fragment. The new merged node then contains a weighted average vector of the old nodes vectors and nodes similarity are updated using `updateNodesSim` function. The algorithm continues to merge nodes until the similarity measure drops below α. Because the final resulting graph is not guaranteed to contain only one path from start to end, by using `bestPath` function, the path with the maximum merged nodes (based on the number of events) is considered to be the final output of the model.

Algorithm 1

F : set of text fragments f
E : map of f to sets e_f of gold event annotations

/* **Create initial model** */
$G \leftarrow \emptyset$
foreach $f \in F$ **do**
1 | $g \leftarrow$ `constructLinearBranch` (*f, E.get(f)*)
| *G.add(g)*
2 **end**
3 *model* $\leftarrow$ `linkGraphs` (G)

/* **Merging process** */
4 $\alpha \leftarrow$ `computeTFIDFAvgSim` (F)
5 *nodesSim* $\leftarrow$ `computeNodesSim` (*model*)
6 *(maxPairSim,pairs)* $\leftarrow$ `findMostSim` (*nodesSim*)
repeat
7 | **if** $\neg$ `pairsIntroduceCycle` (*model, pairs*) **then**
8 | | *model* $\leftarrow$ `merge` (*pairs*)
9 | | *nodesSim* $\leftarrow$ `updateNodesSim` (*model,nodesSim*)
10 | **else**
11 | | *nodesSim* $\leftarrow$ `setSimToZero` (*pairs,nodesSim*)
12 | **end**
13 | *(maxPairSim,pairs)* $\leftarrow$ `findMostSim` (*nodesSim*)
14 **until** *maxPairSim* $< \alpha$;
15 bestPath $\leftarrow$ `findBestPath` (*model*)

4 Experiment

We evaluate our approach against a gold-standard annotation of Moses' from the Quran. We first describe how we collected and annotated the data (§4.1). After that we demonstrate the experiment setup (§4.2) and the evaluation (§4.3). Then we report the performance of our approach (§4.5). Finally, we do an error analysis of the performance of our system (§5).

4.1 Data

Moses was an important figure whose story is central to the major Abrahamic religions, including Judaism, Christianity, and Islam. Moses' story is found in fragmentary form throughout the holy books of these religions, with some parts repeated, but in different contexts and sometimes from different perspectives. In the Quran, the holy book of Islam, the story of Moses appears in eight different fragments across six different chapters (suras) comprising 283 verses. Thus the story of Moses serves as an excellent example for the evaluation of our approach to story fragment stitching. The relevant suras and verses are listed in Table 1, along with the number of events present in the fragments of each chapter.

We annotated verses based on a comparative analysis of Moses' story in the Old Testament and the Quran by [Ghanbari and Ghanbari, 2008]. The Ghanbari study breaks Moses' story 43 event categories, shown in Table 3 in chronological order. For the annotation, three annotators labeled each verse with its single relevant event. We measured a Fleiss' kappa of 0.76, which represents excellent agreement. The annotation was originally done on the Arabic version of the Quran, but we transferred the annotations to an English translation [Ali, 1973] for the remainder of the study.

We excluded one fragment (Sura 2 [Al-Baqarah], verses 50–60) from the analysis because its timeline is quite different from the fabula order. We manually extracted 708 total event mentions from the remaining seven fragments. Our annotation procedure followed the standards outlined for events in the TimeML standard [Saurı *et al.*, 2006]. We omitted 135 *Reporting* mentions (e.g., *say*, *reply*, etc.) because these usually are just indicators of direct speech, and do not correspond to plot events. This resulted in 573 event mentions relevant to the plot, which we labeled as to which specific event it referred in the Moses timeline (Table 3). 301 of the event mentions were labeled with an event described in the timeline, while 272 were not relevant.

4.2 Experimental Setup

We used the `netwrokx` library [Hagberg *et al.*, 2008] for graph operations. We extracted event contextualized embedding using the `flair` implementation [Akbik *et al.*, 2018] of the BERT model with the default parameters[2]. The *tf-idf* weights for the lemmas of all tokens excluding stop words are computed using `spaCy` [Honnibal and Montani, 2017] and `scikit-learn` libraries [Pedregosa *et al.*, 2011]. The event arguments are extracted and resolved using the `AllenNLP` semantic role labeling (SRL) and coref-

[2] `bert_base_uncased, layers=-1, pooling_operation=first`

Sura	Verses	# of Verses	Total # of Event Mentions	# of Moses Event Mentions	# of Event Categories
2. Al-Baqarah	~~50–60~~, 63–73, 92–93	13 (11+2)	36 (25+11)	9 (6+3)	6
7. Al-A'raf	103–161	59	149	99	23
10. Yunus	75–92	18	36	12	6
20. Ta-Ha	9–98	90	195	78	28
26. Ash-Shuara	10–67	58	63	37	8
28. Al-Qasas	7–40	34	94	66	14
Total	283	272	573	301	

Table 1: Number of verses (inclusive ranges), event mentions, and events of the Moses story in each fragment. Listed are the total number of non-Reporting Event mentions, the total number of event mentions labeled as an event from the Moses timeline, and the total number of distinct labels found in that fragment. The first fragment (Al-Baqarah verses 50–60) is omitted from the data because it violated the linear time order constraint.

erence systems [Gardner *et al.*, 2018; He *et al.*, 2017; Lee *et al.*, 2017].

4.3 Evaluation

For the evaluation, we used the temporal awareness measure [UzZaman *et al.*, 2013] used in both event ordering task SemEval-2015 [Minard *et al.*, 2015] and event sequence task TAC-KBP-2017 [Mitamura *et al.*, 2017]. The temporal awareness metric calculates precision and recall values based on the closure and reduction graphs. For a directed graph, a reduced graph is derived from the original graph by having the fewest possible edges that have the same reachability relation as the original graph. In this work, the final directed path of nodes in the final model represents the reduced graph. For example, consider the final directed path of nodes in the final model to be:

$$start \rightarrow n_1 \rightarrow n_2 \rightarrow n_3 \rightarrow end$$

where, for example, events $e_1, e_2 \in n_1$, $e_3 \in n_2$, and $e_4, e_5 \in n_3$. The reduced graph (G^-) is represented as the following edges: $\langle (e_1, e_3), (e_2, e_3), (e_3, e_4), (e_3, e_5) \rangle$ and the transitive closure graph (G^+) is represented as the following edges: $\langle (e_1, e_3), (e_2, e_3), (e_1, e_4), (e_1, e_5),$
$(e_2, e_4), (e_2, e_5), (e_3, e_4), (e_3, e_5) \rangle$,where the relation between (e_i, e_j) is defined as *before* relation. The temporal awareness metric calculates the precision and recall as follow:

$$precision = \frac{|System^- \cap Reference^+|}{|System^-|} \quad (1)$$

$$recall = \frac{|Reference^- \cap System^+|}{|Reference^-|} \quad (2)$$

,where *System* and *Reference* are the proposed approach and the gold standard, respectively. The final F_1 score is the harmonic mean of the precision and recall values.

4.4 Baseline

We used the Needleman-Wunsch algorithm [Needleman and Wunsch, 1970] as a baseline. Needleman-Wunsch is a well-known global alignment algorithm used in bioinformatics and the social sciences. Using dynamic programming, this algorithm searches for optimal alignment of an arbitrary number of items (the events lemma in our case) by using a scoring function that penalizes the dissimilarities and the insertion of gaps. We used the default implementation[3]developed by [Dekker and Middell, 2011] which follows the group of progressive alignment algorithms where two sequences are aligned and then the result is aligned to the next sequence. It repeats the procedure until all sequences are aligned.

Model	Prec.	Recall	F_1
Needleman-Wunsch	0.41	**0.70**	0.52
tf-idf	0.43	0.40	0.42
BERT	0.77	0.50	0.61
Concat	**0.81**	0.51	**0.63**

Table 2: Results on the Moses data using F_1 temporal awareness metric. **Concat** is the proposed model as described in 3.2, whereas *tf-idf* and BERT are variant models of the proposed model when tested alone.

4.5 Result

Table 2 shows our model results compared to the baseline. In the table, we compare three models: the proposed model **Concat**, *tf-idf* and BERT, which are sub-models of the proposed model when considered alone for graph's nodes vector representation as described in 3.2. As shown in bold in Table 2, the **Concat** approach achieves 0.63 F_1 which outperforms the baseline by 11 points and both *tf-idf* and BERT alone by 21 and 2 points, respectively. Also, the table shows that concatenating both *tf-idf* and BERT produced the best result even though *tf-idf* alone unperformed the baseline. It is also clear that BERT contextualized embeddings play a major role in the model merging approach for nodes' vector representation when assessing similarity between nodes.

5 Error Analysis

Inspection of the results revealed several sources of errors, aside from the usual noise introduced by the various subcomponents, such as the SRL or co-reference systems. Some

[3]https://github.com/interedition/collatex

peculiarities of Quranic language cause errors. For example, the word *We* is usually present as an event's argument when God is speaking of himself. This causes problems for the coreference resolution system, in that it does not pair *we* with such mentions *Lord* and *God*, thus introduces additional errors into the system. Also, some events have the same event mention and arguments but happen at different points in the timeline. Example 1 shows text from different parts of the story: the first is *when God shows Moses one of the signs* whereas the second is *when Moses shows the Pharaoh the sign*. Notably, the two events have the same event triggers (showed in bold) and the same arguments (underlined).

> *(20:19–20) "Throw it down, O Moses," said (the Voice). So he* ***threw*** *it down, and lo, it* ***became*** *a running serpent.*
>
> *(7:106–107) He said: "If you have brought a sign then display it, if what you say is true." At this Moses* ***threw*** *down his staff, and lo, it* ***became*** *a live serpent.*

Example 1: An example to show when two events happen at different points in the timeline.

Further, the approach is sensitive to the order of merges. If an incorrect merge is performed early, this can eliminate correct merges later on account of the *no-cycles* constraint. Therefore performing only the highest confidence merges first is critical, and errors in that process degrade other distance parts of the model.

6 Contributions

We introduced the story fragment stitching problem, the task of merging partial tellings of a story into a unified whole. We have introduced an approach that models the story's fragments in a graph and applies an adapted model merging approach to merge similar nodes and produce an ordered, end-to-end list of story events. Our approach achieves a performance of 0.63 F_1 using the temporal awareness metric.

7 Acknowledgements

Mr. Aldawsari was funded in part by a doctoral fellowship from Prince Sattam Bin Abdulaziz University, as well as NSF Grant IIS-1749917 to Dr. Finlayson. We thank Seyedeh Mohadeseh Taheri Mousavi and Zahra Ejei for their assistance in annotating the verses from the Quran in the original Arabic. The idea of combining distributional semantics with automatic story model merging was proposed and developed by Mr. Asgari in his Master's thesis at CSAIL MIT supervised by Dr. Finlayson in 2013–2014 academic year. The creation of the corpus was also among the contributions of that thesis.

Event	# of fragments	# of events	F_1
Moses's Birth			
1. Moses' Birth and left in the Nile.	2	6	0.33
Moses is Rescued from the Nile			
2. Moses is rescued from the Nile.	2	4	0.45
3. Moses' sister kept an eye on him.	2	4	0.55
4. Moses brought back to his mother.	2	2	0.34
5. Moses after infancy and through maturity.	1	2	0.68
Moses kills the Egyptian			
6. Moses beats and kills the Egyptian.	2	10	0.85
Moses flees to the Madyan			
7. Moses ran away to the Madyan.	1	1	0.74
Moses' Marriage			
8. Moses protected Shu'ayb's daughters.	1	20	0.53
9. Moses traveled with his family.	2	3	0.50
Moses is Chosen to be a Prophet			
10. Moses saw the fire from the distance.	2	12	0.60
11. Moses talked to God through the burning bush.	2	4	0.89
God Shows Moses the Miracles			
12. God changed the wand to the snake.	2	11	0.69
13. God illuminated Moses' hand.	2	5	0.63
God Send Moses to the Pharaoh			
14. God commanded Moses to meet the Pharaoh.	2	9	0.55
Moses Speaks with the Pharaoh			
15. Moses and Aaron went to the Pharaoh with miracles.	4	8	0.47
16. Moses showed Pharaoh the signs.	3	10	0.56
17. Pharaoh refused their message.	3	4	0.55
18. Pharaoh accused Moses.	4	5	0.80
19. Pharaoh requested a competition with Moses.	3	8	0.68
20. Competition between Moses and the magicians.	4	34	0.63
21. Magicians believed in Moses's message.	3	8	0.88
22. Magicians are threatened by the Pharaoh.	3	6	0.72
23. Pharaoh cruelty to the believers.	1	6	0.28
God Sends Calamities in Egypt			
24. Calamities are sent to the Egyptians and the Pharaoh.	1	6	0.58
25. God withdraw the punishment.	1	1	0.65
26. God commanded Moses to travel with his people.	3	6	0.31
27. Pharaoh and his army followed Moses and his people.	3	6	0.63
Parting of the Red Sea			
28. Separation of the Sea and drowning of the Pharaoh.	5	14	0.49
29. God saved Moses and his people.	3	5	0
Going to Mt. Sinai to Receive the Commandments			
30. Moses went to Sinai for 40 nights.	3	6	0.47
31. God sent down food and brings forth water.	1	4	0.73
32. Moses met God and appeared on mountain.	2	15	0.43
33. Moses delivered the commands and the stone tablets.	3	3	0
The People Betray God			
34. Worshipping the Calf in the Absence of Moses.	2	14	0.19
35. Moses returned to his people.	1	3	0.29
36. Samiri explained to Moses what he saw.	2	3	0
37. Moses blamed his brother.	1	10	0
38. Moses returned to God.	1	3	0
39. Moses stroked the stone.	1	9	0
Wandering in the Desert			
40. Israelites are commanded to take over the holy region.	1	5	0
41. The disobedient Israelites won't enter the holy region.	2	4	0
42. God punished them.	2	1	0
43. Sacrifice of a heifer.	1	1	0

Table 3: The 43 events in the Moses timeline. The second column refers the number of fragments in which the corresponding event appears. The third column refers to the number of events mentions for the event across all fragments. The last column is the standard F_1 measure for extraction of the corresponding event, compared to the gold standard.

References

[Akbik *et al.*, 2018] Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *COLING 2018, 27th International Conference on Computational Linguistics*, pages 1638–1649, 2018.

[Ali, 1973] Abdullah Yusuf Ali. *The Holy Qur'an: text, translation and commentary*. Islamic University of Al ima Mohammad ibn SAUD, 1973.

[Bagga and Baldwin, 1999] Amit Bagga and Breck Baldwin. Cross-document event coreference: Annotations, experiments, and observations. In *Proceedings of the Workshop on Coreference and its Applications*, pages 1–8. Association for Computational Linguistics, 1999.

[Barhom *et al.*, 2019] Shany Barhom, Vered Shwartz, Alon Eirew, Michael Bugert, Nils Reimers, and Ido Dagan. Revisiting joint modeling of cross-document entity and event coreference resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4179–4189, Florence, Italy, July 2019. Association for Computational Linguistics.

[Bordwell, 2007] David Bordwell. *Poetics of Cinema*. New York: Routledge, 2007.

[Charniak, 1972] Eugene Charniak. *Toward a model of children's story comprehension.* PhD thesis, Massachusetts Institute of Technology, 1972.

[Chaturvedi *et al.*, 2018] Snigdha Chaturvedi, Shashank Srivastava, and Dan Roth. Where have i heard this story before? identifying narrative similarity in movie remakes. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 673–678, 2018.

[Dekker and Middell, 2011] Ronald H Dekker and Gregor Middell. Computer-supported collation with collatex: managing textual variance in an environment with varying requirements. *Supporting Digital Humanities*, pages 17–18, 2011.

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

[Dyer, 1983] Michael George Dyer. *In-depth understanding: A computer model of integrated processing for narrative comprehension*. MIT press, 1983.

[Finlayson, 2012] Mark Mark Alan Finlayson. *Learning narrative structure from annotated folktales*. PhD thesis, Massachusetts Institute of Technology, 2012.

[Finlayson, 2016] Mark Alan Finlayson. Inferring propp's functions from semantically annotated text. *The Journal of American Folklore*, 129(511):55–77, 2016.

[Forster, 1927] Edward M. Forster. *Aspects of the Novel*. E. Arnold & Co., London, 1927.

[Frank *et al.*, 2003] Stefan L Frank, Mathieu Koppen, Leo GM Noordman, and Wietske Vonk. Modeling knowledge-based inferences in story comprehension. *Cognitive Science*, 27(6):875–910, 2003.

[Gardner *et al.*, 2018] Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. Allennlp: A deep semantic natural language processing platform. *arXiv preprint arXiv:1803.07640*, 2018.

[Ghanbari and Ghanbari, 2008] Bakhshali Ghanbari and Zohreh Ghanbari. Comparative study of moses' position in quran and torah. *Journal of Theology*, (5):73–90, 2008.

[Goyal *et al.*, 2013] Kartik Goyal, Sujay Kumar Jauhar, Huiying Li, Mrinmaya Sachan, Shashank Srivastava, and Eduard Hovy. A structured distributional semantic model for event co-reference. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 467–473, 2013.

[Hagberg *et al.*, 2008] Aric Hagberg, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

[He *et al.*, 2017] Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 473–483, 2017.

[Honnibal and Montani, 2017] Matthew Honnibal and Ines Montani. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing, 2017. https://github.com/explosion/spaCy; Last accessed on Nov 28 , 2019.

[Iyyer *et al.*, 2016] Mohit Iyyer, Anupam Guha, Snigdha Chaturvedi, Jordan Boyd-Graber, and Hal Daumé III. 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, 2016.

[Kenyon-Dean *et al.*, 2018] Kian Kenyon-Dean, Jackie Chi Kit Cheung, and Doina Precup. Resolving event coreference with supervised representation learning and clustering-oriented regularization. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 1–10, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[Lee *et al.*, 2012] Heeyoung Lee, Marta Recasens, Angel Chang, Mihai Surdeanu, and Dan Jurafsky. Joint entity and event coreference resolution across documents. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 489–500. Association for Computational Linguistics, 2012.

[Lee *et al.*, 2017] Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[Minard *et al.*, 2015] Anne-Lyse Minard, Manuela Speranza, Eneko Agirre, Itziar Aldabe, Marieke van Erp, Bernardo Magnini, German Rigau, and Rubén Urizar. SemEval-2015 task 4: TimeLine: Cross-document event ordering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 778–786, Denver, Colorado, June 2015. Association for Computational Linguistics.

[Mitamura *et al.*, 2017] Teruko Mitamura, Zhengzhong Liu, and Eduard H. Hovy. Events detection, coreference and sequencing: What's next? overview of the tac kbp 2017 event track. In *TAC*, 2017.

[Mueller, 2007] Erik T Mueller. Understanding goal-based stories through model finding and planning. In *Intelligent Narrative Technologies: Papers from the AAAI Fall Symposium*, pages 95–101. AAAI Press Menlo Park, CA, 2007.

[Needleman and Wunsch, 1970] Saul B Needleman and Christian D Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453, 1970.

[Nikolentzos *et al.*, 2017] Giannis Nikolentzos, Polykarpos Meladianos, François Rousseau, Yannis Stavrakas, and Michalis Vazirgiannis. Shortest-path graph kernels for document similarity. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1890–1900, 2017.

[Pedregosa *et al.*, 2011] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[Reiter, 2014] Nils Reiter. *Discovering Structural Similarities in Narrative Texts using Event Alignment Algorithms*. PhD thesis, Heidelberg University, 2014.

[Riloff, 1999] Ellen Riloff. Information extraction as a stepping stone toward story understanding. *Understanding language understanding: Computational models of reading*, pages 435–460, 1999.

[Roth and Frank, 2012] Michael Roth and Anette Frank. Aligning predicates across monolingual comparable texts using graph-based clustering. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 171–182. Association for Computational Linguistics, 2012.

[Salton and McGill, 1986] Gerard Salton and Michael J McGill. *Introduction to modern information retrieval*. McGraw-Hill, Inc., New York City, 1986.

[Saquete and Navarro-Colorado, 2017] Estela Saquete and Borja Navarro-Colorado. Cross-document event ordering through temporal relation inference and distributional semantic models. *Procesamiento del Lenguaje Natural*, 58:61–68, 2017.

[Saurı *et al.*, 2006] Roser Saurı, Jessica Littman, Bob Knippen, Robert Gaizauskas, Andrea Setzer, and James Pustejovsky. Timeml annotation guidelines version 1.2. 1, 2006.

[Schank and Abelson, 1975] Roger C Schank and Robert P Abelson. Scripts, plans, and knowledge. In *IJCAI*, volume 75, pages 151–157, 1975.

[Schank and Abelson, 1977] Roger C Schank and Robert P Abelson. *Scripts, plans, goals, and understanding: An inquiry into human knowledge structures*. Hillsdale, NJ: Lawrence Erlbaum, 1977.

[Stolcke and Omohundro, 1993] Andreas Stolcke and Stephen Omohundro. Hidden markov model induction by bayesian model merging. In *Advances in neural information processing systems*, pages 11–18, 1993.

[UzZaman *et al.*, 2013] Naushad UzZaman, Hector Llorens, Leon Derczynski, James Allen, Marc Verhagen, and James Pustejovsky. Semeval-2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 1–9, 2013.

[Wilensky, 1978] Robert Wilensky. Understanding goal-based stories. Technical report, YALE UNIV NEW HAVEN CONN DEPT OF COMPUTER SCIENCE, 1978.

[Winston, 2014] Patrick Henry Winston. The genesis story understanding and story telling system a 21st century step toward artificial intelligence. Technical report, Center for Brains, Minds and Machines (CBMM), 2014.