Placing (Historical) Events on a Timeline: A Classification cum Co-ref Resolution Approach

Anonymous ACL submission

Abstract

The event timeline provides one of the most effective ways to visualize the important historical events that occurred over a period of time, presenting the insights that may not be so apparent from reading the equivalent information in textual form. By leveraging generative adversarial learning for important event classification and by assimilating knowledge based tags for improving the performance of event coreference resolution we introduce a two staged system for event timeline generation from multiple (historical) text documents. We demonstrate our results on two manually annotated historical text documents. Our results can be extremely helpful for historians, in advancing research in history and in understanding the socio-political landscape of a country as reflected in the writings of famous personas.

1 Introduction

Timeline serves as one of the most effective and easiest means to contextualize and visualize a complex situation ranging from grasping spatio-temporal events in historical studies to critical decision making in businesses. With the stupendous increase of textual resources for many historical contents in several online platforms it has become imperative for the history researchers to understand the chronological orderings of the incessant historical phenomenon. The event timeline can be an extremely useful aid to highlight the temporal and causal relationships among several events and the interactions of the characters over time, that results in identifying common themes that arise over the period of interest in a historical document (see Figure 1 in Appendix A.1).

In this paper we present a full pipeline to build a chronology of events extracted from historical text. Our contributions are as follows.

- We curate a first of its kind dataset from two different historical texts – the *Collected Works of Mahatma Gandhi* (CWMG) and the *Collected Works of Abraham Lincoln* (CWAL) for our experiments. For each of these datasets we manually annotate sentences that correspond to important events. Next for each of these annotated sentences we also further annotate the coreferences to the same event; we call these event coreferences. Upon acceptance we shall release this data for future research.
- We introduce a novel divide-and-conquer based approach to generate event timeline from timestamped historical texts. In the first step, we classify sentences as containing events or not using a generative adversarial learning setup. In the subsequent step we compute event coreferences using both unsupervised and supervised methods. The main novelty here is that inclusion of world knowledge in the form of tag embeddings results in higher performance gains.
- We present a rigorous evaluation of both the steps as well as the full system which was absent in previous literature (Bedi et al., 2017). Further we compare our results to the closely related event timeline summarization tasks by suitably adapting them so that the comparison is fair.
- In order to determine the readability and usefulness of the timeline, we conducted an online crowd-sourced survey. 93% survey participants found it to be effective in summarizing historical timeline of events.
- We also show that our method is generic by evaluating it against a COVID-19 news related dataset which is not a historical text per se.

2 Related work

Important event classification: Zhang and Wallace (2016) used CNN to analyse sensitivity for text classification. Miyato et al. (2017) and Zhang et al. (2020) introduced virtual adversarial training methods for robust text classification from a small number of training data points.

Event coreference resolution: Recent works like Choubey and Huang (2017), Kenyon-Dean et al. (2018) have used neural network based architecture to train their model on benchmark coreference dataset (ECB+ Cybulska and Vossen (2014)). Lu et al. (2020) attempted to create an end-to-end event coreference resolution system based on the standard KBP dataset[1].

Timeline of historical events: Bamman and Smith (2014) proposed an unsupervised generative model to construct the timeline of biographical life-events leveraging encyclopaedic resources such as Wikipedia. Aprosio and Tonelli (2015) also uses Wikipedia for timeline construction of historical events. Bedi et al. (2017) attempted to construct an event timeline from history textbooks considering the sentences having temporal expressions. Palshikar et al. (2019) proposed an automatic approach to capture and visualize temporal ordering of interactions between multiple actors. Adak et al. (2020) created an AI-enabled web portal based on CWMG dataset.

Timeline summarization (TLS): The timeline summarization task aims to summarize time evolving documents.Gholipour Ghalandari and Ifrim (2020) evaluated existing state-of-the-art methods for news timeline summarization and proposed *datewise* and *clustering* based approaches on the TLS datasets. Born et al. (2020) demonstrated the potential of employing several IR methods on TLS tasks based on a large news dataset. La Quatra et al. (2021) proposes a new approach by generating date level summaries, and then selecting the most relevant dates for the timeline summarization.

The present work: Our paper is closest in spirit to the work done by Bedi et al. (2017). In this paper the authors outlined the challenges related to event coreference for timeline generation; however, they did not suggest ways to effectively tackle these challenges and, thereby, solve the problem. We close this gap in our paper by proposing an efficient approach to resolve event coreference. Our work has also close parallels with the event timeline summarization (TLS) task. Nevertheless, previous TLS researchers mostly worked on the documents containing multiple news articles, which are rich in events. These works have not focused much on prior event detection and have not addressed how they can be effectively generalized in historical text documents such as biographies. Our work for the first time shows that event detection could largely benefit TLS tasks in the context of historical texts.

[1] https://www.ldc.upenn.edu/collaborations/past-projects/tac-kbp

3 Data preparation

In this section we present the details of the datasets that we prepare for our experiments. We also outline the overall annotation process of these datasets.

3.1 Datasets

Collected works of Mahatma Gandhi: We leverage the Collected Works of Mahatma Gandhi (CWMG) available at (Preservation and Trust, 2013), an assortment of 100 volumes consisting of the books, letters, telegrams written by Mahatma Gandhi and also the compiled writings of the speeches, interviews engaging Gandhi. This data covers many important historical events within the time period of 1884-1948 in British colonised India.

Collected works of Abraham Lincoln: The second dataset we have use to demonstrate our system is based on the life-long writings of the 16^{th} president of the United States, Abraham Lincoln, formally known as the Collected Works of Abraham Lincoln (CWAL)[2] comprising a total of 8 volumes.

COVID-19 event dataset: In addition, to establish the generalizability of the approach, we collect 140 major events, that happened in India during the COVID-19 pandemic from different sources such as *Wikipedia*[3], *Who.int*[4] to be placed on a timeline for elegant visualisation using our system.

3.2 Pre-processing

From the 100 volumes of text files from CWMG we first extract all the letters containing the publication dates and recipients name. There were a total of 28531 letters in the entire CWMG. We primarily use the letters for our experiments as we observe that they contain the best temporal account of the events. From the overall set of letters, we select the year range 1930–1935 since this range has the largest collection of letters. In order to further choose the right data sample, we categorize the letters into *formal* and *informal* types based on

[2] https://quod.lib.umich.edu/l/lincoln/

[3] https://en.wikipedia.org/wiki/COVID-19_pandemic_in_India

[4] https://www.who.int/india/emergencies/coronavirus-disease-(covid-19)/india-situation-report

Doc creation time (Initial reference time)	Important sentences	Updated reference time
May 4, 1930	He was arrested at 12.45 a.m. on May 5.	May 5, 1930
May 4, 1930	In Karachi, Peshawar and Madras the firing would appear to have been unprovoked and unnecessary.	May 4, 1930

Table 1: Sample list of sentences from CWMG after the sentence classification. The explicit temporal expression inside the sentence is highlighted.

the recipients of the letters. A simple heuristic that we follow is – the letters written to government officials and famous historic personalities can be categorized as formal while those written to the family members can be classified as informal ones. We collect the list of Mahatma Gandhi's family member names from Gandhian experts for identifying the informal letters. We manually notice that the formal letters contain much more useful historic information than the informal ones. We therefore only consider the formal letters for manually annotating the useful sentences. In addition, we only consider the letters which have more than 1000 words in its content. This results in 41 letters with substantial content.

3.3 Annotation

In this section we outline the data annotation procedure for the two phases. Recall that our method has two important steps – event classification and coreference resolution. While the event classification phase is supervised (Level I annotations), the coreference resolution is done using both unsupervised and supervised techniques. The annotations for the coreference resolution (Level II annotations) are therefore required to (a) train the supervised approach and (b) test the efficacy of both the unsupervised and the supervised approaches.

Level I – Important sentences: Finally, out of these filtered letters we manually annotate all the sentences of 18 letters (i.e., 979 sentences in all). The remaining sentences (i.e., 1689 in total) from the rest of the letters were left unlabelled. Both of these labelled and unlabelled sentences were used for training the classifier. The classes in which the sentences were classified were based on their historical importance. In specific, we identify three such important classes – (a) the *events/facts*, which typically represent that something happened or took place (Pustejovsky et al., 2003), e.g., '*A vegetable market in Gujarat has been raided because the dealers would not sell vegetables to officials*'[5]; (b) the *demands*, which represent the demands Mahatma Gandhi had made to the British government through his writings, e.g., '*The terrific pressure of land revenue, which furnishes a large part of the total, must undergo considerable modification in an independent India.*' and (c) others (i.e., not important). As the examples suggest, each individual sentence is annotated as important (i.e., containing an event) or not. In order to further enrich the dataset we collect gold standard events related to Mahatma Gandhi from an additional reliable and well maintained resource[6]. We obtain 86 additional sentences thus making a total of 1065 (i.e., 979 + 86) important sentences (see Table 8 in Appendix A.4 for the category distribution.).

For the CWAL we simply extract all the sentences from volume 2 and follow similar approaches to annotate important sentences as in the case of CWMG. Without considering any filtering criteria we consider all the 111 articles of volume 2 including his letters and propositions which consist of a total of 1386 sentences. Out of these 720 sentences were manually annotated (see Table 8 in Appendix A.4 for the category distribution.).

Annotator details and annotation guidelines: For both the datasets three annotators annotated the sentences. The annotation process was led by one PhD student along with two undergraduate students. The PhD student had substantial experience in historical text analysis and will be referred to as the expert annotator henceforth. The first level of annotation was carried out for each of the sentences and based on the assumption that a full sentence corresponds to an event/demand. All the annotators annotated the sentences independently. For the training of the two undergraduate annotators, they were provided with the examples of 25 gold standard events and demands each. The gold standard events were collected from the reliable resource mentioned in the earlier paragraph and the gold standard demands were collected from the formal letters of Mahatma Gandhi which were first annotated by the expert annotator and verified by a Gandhian scholar (see Table 6 in Appendix A.3 for example annotations). The inter-annotator agreements, i.e., Cohen's κ were 0.66 and 0.58 for the former and the latter datasets respectively. Table 8 shows the category distribution for both the

[5]Such sentences would typically consist of participants and locations.

[6]https://www.gandhiheritageportal.org/

datasets. The Level I annotation was not carried out for the COVID-19 dataset because, each sentence collected were presented as events in the mentioned portals and thus we considered all the sentences as important events.

Level II – Coreference resolution: The second round of annotation was carried out for evaluating the event coreference detection task on the same dataset. For this case we only annotate the texts which were marked important during the Level I annotation. In addition, the Level II annotation was also carried out for the COVID-19 event dataset.

Annotator details and annotation guidelines: The same annotators annotated for the Level II phase. The annotators were provided with sentences, the reference documents (letters) from which the sentences were extracted and the reference time (document publication date). Based on the perception of the annotators, the sentences that potentially referred to the same event were placed in the same cluster. The coreferences have been placed by the annotators in different clusters based on different factors like the commonness of the mentioned times, entities and the event name/composition. Consider these two sentences - '*The crowd that demanded restoration of the flag thus illegally seized is reported to have been mercilessly beaten back.*' and '*Bones have been broken, private parts have been squeezed for the purpose of making volunteers give up, to the Government valueless, to the volunteers precious salt*'. Although there is no explicit mention of time in either of the sentences, both of them are from the same document and thus their reference dates would be the same as the publication date of the document. Also both of them refer to similar types of atrocities. So these two sentences should be placed in the same cluster. We first carried out a trial round for the two undergraduate annotators by using 100 randomly chosen important sentences from the Level I phase and the trial annotations were verified by the expert annotator. Finally for the complete Level II annotations, the inter-annotator agreements were 0.74, 0.61, and 0.78 for the CWMG, the CWAL and the COVID-19 dataset respectively using MUC (Vilain et al., 1995) based F1-score (Ghaddar and Langlais, 2016) (see Table 7 in Appendix A.3 for example annotations and Appendix A.5 for other agreement metrics.).

4 Methodology

Our method consists of three major components (see Figure 2 in Appendix A.2.): (i) important sentence extraction, (ii) event coreference resolution, and (iii) timeline visualization. The arrows represent the direction of data flow. In this section we describe in detail the methods used for each of these components.

4.1 Important event extraction

Baselines: As baselines, we use *SVM* (Hearst, 1998) and *Multinomial Naïve Bayes* (Kibriya et al., 2004) on simple bag-of-words feature. For *SVM* we use linear kernel. For the evaluation of the classifiers we use a 70:30 train-test split of the annotated data.

Fine-tuned BERT: Apart from the above two baselines, we try BERT (Devlin et al., 2019) neural network based framework for the classification. We train the model using the PyTorch (Paszke et al., 2019) library, and apply *bert-base-uncased* pre-trained model for text encoding. We use a batch size of 32, sequence length of 80 and learning rate of $2e-5$ as the optimal hyper-parameters for training the model.

GAN-BERT text classifier: In search for further enhancement of the performance based on our limited sets of labelled data, we employ the *GAN-BERT* (Croce et al., 2020) deep learning framework for classifying the important sentences. It uses generative adversarial learning to generate augmented labelled data for semi-supervised training of the transformer based BERT model. It improves the performance of BERT when training data is scarce and is therefore highly suited for our case. Here we also feed the unlabeled data sample, as discussed in section 3.3, to help the network to generalize the representation of input texts for the final classification (Croce et al., 2020).

4.2 Event coreference resolution

Once the classification was done we end up with 'eventful' sentences linked to its corresponding document creation time in the format noted in Table 1.

Time within sentences: For generating the accurate event timeline we need to assign a valid date to a particular sentence (or event). For example, in the first sentence in Table 1, although the document publication time is mentioned to be `May 4, 1930`, the sentence clearly has embedded in it the exact event date `May 5, 1930` apparent from the snippet '*arrested on May 5*'. Therefore, if the explicit time is present in the sentence we use it directly, else we use the creation/publication date of the document. We extract the explicit mention of time in the text using the *HeidelTime* (Strötgen and Gertz, 2010) tool. This tool is capable of identifying embedded mentions of temporal expressions such as *'yesterday', 'next day' etc.*

Tag generation from world knowledge: An individual sentence does not always contain much information about the event which it is getting referred to. So we attempt to incorporate world knowledge for each individual sentence. By using each sentence as a query we gather the top five *Google* search results using the *googlsearch* api[7] and also consider the document from which the sentence was being extracted. Next we analyse the search result using *TextRank*[8], *Rake*[9] and *pointwise mutual information*[10] to generate top keywords present in the search result. Although these methods produce reasonably good results, in many cases we needed to manually filter out certain noisy tags. For each sentence we therefore land up with one or more tags. We retain the top ten tags for every sentence which means that the number of tags for a sentence could vary between one and ten. The details of the tag generation procedure mentioned in Appendix A.6. We do not use encyclopaedic resources such as Wikipedia to get the search results because the datasets we are using, are only available in a few very specific websites. We fed the list of keyword(s) or tag(s) obtained for a sentence to the pre-trained *sentence-bert* model for obtaining a 768 dimensional embedding representation of the keywords.

Unsupervised event clustering: We employ several unsupervised approaches for sentence coreference resolution. As baselines, we choose two commonly used approaches for coreference resolution – (a) *Lemma*: It attempts to put the sentence pairs in same coreference chain which share the same head lemma, (b) *Lemma-*δ: In addition to same head lemma as a feature, it also computes the cosine similarity (δ) between the sentence pair based on *tf-idf* features, and only places the sentence pairs in the same coreference chain if δ exceeds some threshold. Then the sentence clusters were created using agglomerative clustering method. To extract the head lemma of a sentence, we use the *SpaCy* dependency parser.

Apart from these two common baselines, we vectorize the sentences using *tf-idf* vectorization technique and then apply different clustering techniques such as *Gaussian-Mixture*[11] model, *agglomerative clustering* to cluster the sentences corresponding to similar events. We also use the pre-trained *sentence-bert* (Reimers and Gurevych, 2019) model to encode the sentences and apply similar clustering techniques. Finally, we concatenate the sentence embedding with the tag embedding generated from that particular sentence. We again cluster the sentences based on this new representation. This, as we shall later see, significantly improves the performance of the clustering phase. We evaluate the clustering results on the basis of the annotated data which had been obtained in the second phase of data annotation. We used the *elbow* method to find the optimal number of clusters in case of Gaussian-Mixture and used *dendogram* to select the optimal distance threshold for the suitable number of clusters in case of agglomerative clustering. The distance threshold we selected were 0.25, 0.6 and 0.6 for CWMG, CWAL and COVID-19 data respectively.

Supervised event mention-pair model: An *event mention* is a sentence or phrase that defines an event and one event may contain multiple *event mentions* (Chen et al., 2009). We first create a dataset containing all the possible pairs of *eventful* (i.e., event/fact or demand) sentences from the ground-truth annotations. We set the coreference label to 1 if the sentence pair is contained in the same cluster as per the Level II annotation and 0 otherwise. Here we again use a 70:30 split to generate training and test instances. The overall architecture is inspired from Barhom et al. (2019) (see Appendix A.7). The inputs to the model are the two sentences (i.e. S_1 and S_2) and their corresponding *actions* (i.e., A_1 and A_2), *time* (i.e., T_1 and T_2) and *tags* (i.e., K_1 and K_2). We extract *actions* (i.e., A_i) for each of the sentences (fact or demand might not contain any *action*) using *SpaCy* dependency parser.

Mention pair construction: We used *Tensorflow* (Abadi et al., 2015) tokenizer to vectorize each fea-

[7] https://github.com/MarioVilas/googlesearch

[8] https://github.com/DerwenAI/pytextrank

[9] https://pypi.org/project/rake-nltk/

[10] https://www.nltk.org/howto/collocations.html

[11] https://scikit-learn.org/stable/modules/mixture.html

Dataset	Model	Evaluation Metric	
		Accuracy	F1
CWMG	MNB	0.74	0.45
	SVM	0.79	0.5
	Fine-tuned BERT	0.8	0.57
	GAN-BERT	**0.9**	**0.69**
CWAL	MNB	0.6	0.3
	SVM	0.6	0.34
	Fine-tuned BERT	0.61	0.56
	GAN-BERT	**0.7**	**0.65**

Table 2: Results (accuracy and macro F1-score) for the important event classification using our approaches on the two datasets. MNB: Multinomial Naïve Bayes. Best results are marked in boldface and highlighted in green cells.

ture (i.e., sentences, actions, time and tags) to convert it into sequence of integers after restricting the tokenizer to use only the top most common 5000 words. For the sentences we limit the sequence length to 64. For the other features – actions, time and tags – we limit the sequence length to 10. We always use zero padding for smaller sequences. We next encode the words present in each of these sequences using a pre-trained *GloVe* (Pennington et al., 2014) embedding (100 dimensions). Thus each sentence comes out as a $64 * 100$ size vector representation while each of the other features come out as a $10 * 100$ size vector representation. Now each of these vectors are separately passed through a LSTM (Hochreiter and Schmidhuber, 1997) layer with default hyperparameters to transform them into 128 size vectors each. Next each of these 128 size vectors are passed through separate dense layers to obtain 32 size vectors. Finally, these 32 size vectors are concatenated using a concatenation layer. The output of the concatenation layer is what we term as a *mention representation*. Two mention representations are concatenated to get a pairwise representation (i.e., an *event mention pair*) and passed through a feed forward network to return a score denoting the likelihood that two mentions are coreferent (see Figure 3 in Appendix A.7). Based on the predicted pairwise score on the test instances we used a threshold (0.5 in our case) to generate a similarity matrix of the mentions, and then applied agglomerative clustering to partition the similar mentions into the same clusters.

4.3 Timeline visualization

Once the event coreference resolution phase was successfully executed, we generated visualization for the given event sequence using *vis-timeline*[12], a dynamic, browser based visualization library.

[12] https://visjs.github.io/vis-timeline/docs/timeline/

5 Experiments

5.1 Evaluation metrics

We have used separate evaluation metrics for the two phases.

Important sentence classification: In this case we use the standard *accuracy* and *F1-score* values.

Event coreference resolution: Here we conduct the evaluation based on the widely used coreference resolution metrics – (a) *MUC* (Vilain et al., 1995), (b) *B^3* (Bagga and Baldwin, 2000), (c) *CEAF* (Luo, 2005), and (d) *BLANC* (Recasens and Hovy, 2011). Due to the inconsistency of each of these evaluation metrics (Moosavi and Strube, 2016) we shall also report the average outcomes of all the metrics.

5.2 Results

We evaluate the two different phases separately. Ground-truth data was used from each phase for respective evaluations.

Important event classification: The key results for the two datasets (CWMG and CWAL) are summarised in Table 2. Our approach based on GAN-BERT by far outperforms the standard baselines. For the CWMG dataset, the macro F1-score shoots from 0.50 (SVM) to 0.69 on the three class classification task. Likewise for the CWAL dataset, the macro F1-score shoots from 0.34 (Naïve Bayes) to 0.65.

Evaluation of coreference resolution: For the evaluation of event coreference resolution we use several coreference resolution metrics to analyse the model performance. It is apparent from Table 3 that the approach based on clustering with *sentence-bert* embeddings by far outperforms the baselines *lemma* and *lemma-δ*. For the CWMG dataset, *sentence-bert* + agglomerative clustering is the best overall; for the other two datasets no single method is a clear winner. However, the primary point that we wish to emphasize in the table is the result after incorporating tag embedding. It can be clearly observed that this intuitive, albeit hitherto unreported, technique almost always produces better results (see Appendix A.6 and the Table 10 therein describing the tag generation process in more details). In fact, the assimilation of the tag embeddings with the *sentence-bert* embeddings boosted the overall F1-score by 13%, and 16% for the CWMG and the CWAL datasets respectively. Note that these results hold even if the manual filtering step in the tag generation is completely omitted (see Table 13 in Appendix A.10). An interesting observation is that

Dataset	System	MUC F1	B^3 F1	CEAF_E F1	BLANC F1	Avg (overall) Recall	Avg (overall) Precision	Avg (overall) F1	Time taken
CWMG	Lemma	0.45	0.38	0.20	0.49	0.39	0.38	0.38	45 sec
	Lemma-δ	0.53	0.41	0.19	0.48	0.48	0.40	0.41	7 min 22 sec
	tf-idf + GM	0.53	0.53	0.36	0.60	0.49	0.52	0.50	26 min 14 sec
	tf-idf + AC	0.55	0.50	<u>0.42</u>	0.57	0.50	0.53	0.51	5 min 13 sec
	s-bert + GM	0.61	0.54	0.41	0.60	0.54	0.54	0.54	29 min 34 sec
	s-bert + AC	<u>0.63</u>	<u>0.57</u>	0.40	<u>0.61</u>	<u>0.55</u>	<u>0.56</u>	<u>0.55</u>	7 min 42 sec
	+ tag embedding								
	tf-idf + GM	0.64	0.57	0.45	0.64	0.57	0.60	0.58	28 min 19 sec
	tf-idf + AC	0.62	0.61	0.51	0.66	0.58	0.63	0.60	6 min 57 sec
	s-bert + GM	0.65	0.62	0.48	0.66	0.60	0.60	0.60	30 min 28 sec
	s-bert + AC	0.75	**0.70**	0.52	**0.73**	0.65	**0.71**	0.68	8 min 36 sec
	mention-pair model	**0.91**	0.59	**0.83**	0.53	**0.83**	0.69	**0.72**	2 hr 10 min 32 sec
CWAL	Lemma	0.28	0.11	0.17	0.49	0.26	0.27	0.27	58 sec
	Lemma-δ	0.31	0.15	0.14	0.48	0.28	0.27	0.18	9 min 41 sec
	tf-idf + GM	0.53	0.37	0.35	0.49	0.42	0.45	0.43	41 min 25 sec
	tf-idf + AC	<u>0.57</u>	<u>0.42</u>	0.38	0.49	0.45	<u>0.49</u>	0.46	8 min 5 sec
	s-bert + GM	0.43	0.39	<u>0.40</u>	<u>0.54</u>	0.43	0.46	0.44	46 min 18 sec
	s-bert + AC	0.51	0.42	<u>0.40</u>	<u>0.54</u>	<u>0.46</u>	0.48	<u>0.47</u>	11 min 15 sec
	+ tag embedding								
	tf-idf + GM	0.74	0.52	0.40	0.63	0.56	0.59	0.57	43 min 23 sec
	tf-idf + AC	0.72	0.51	0.48	0.64	0.57	0.61	0.59	9 min 27 sec
	S-bert+ GM	0.74	0.41	0.34	0.67	0.51	0.57	0.54	47 min 12 sec
	s-bert + AC	0.82	**0.53**	0.44	**0.72**	0.60	**0.66**	0.63	11 min 42 sec
	mention-pair model	**0.96**	0.42	**0.78**	0.35	**0.82**	0.65	**0.64**	2 hr 11 min 40 sec
COVID-19	Lemma	0.55	0.39	0.28	0.55	0.51	0.42	0.44	9 sec
	Lemma-δ	0.34	0.29	0.25	0.51	0.35	0.34	0.35	1 min 8 sec
	tf-idf + GM	0.56	0.41	<u>0.36</u>	0.60	0.47	0.50	0.48	6 min 37 sec
	tf-idf + AC	0.59	<u>0.45</u>	<u>0.36</u>	<u>0.62</u>	<u>0.49</u>	<u>0.54</u>	<u>0.51</u>	1 min 44 sec
	s-bert + GM	<u>0.63</u>	<u>0.45</u>	0.32	0.57	0.47	0.51	0.49	8 min 41 sec
	s-bert + AC	0.61	0.44	0.35	0.57	0.48	0.50	0.49	2 min 25 sec
	+ tag embedding								
	tf-idf + GM	0.44	0.33	0.28	0.54	0.39	0.40	0.39	7 min 31 sec
	tf-idf + AC	0.44	0.34	0.32	0.44	0.4	0.42	0.41	2 min 38 sec
	s-bert + GM	0.57	0.41	0.35	0.59	0.47	0.49	0.48	9 min 35 sec
	s-bert + AC	0.63	0.46	0.39	0.59	0.51	0.52	0.52	3 min 19 sec
	mention-pair model	**0.95**	**0.94**	**0.93**	**0.94**	**0.943**	**0.942**	**0.94**	29 min 18 sec

Table 3: Event coreference results before and after tag embedding. GM: Gaussian Mixture based clustering; AC: Agglomerative Clustering; s-bert: sentence-bert. Best results including the tag embedding are marked in boldface and highlighted in green cells. Best results excluding the tag embedding are marked by underline and highlighted in blue cells.

the benefit of the tag embedding is best leveraged by the sentence-bert + agglomerative clustering which is a clear winner for all the three datasets. For the COVID-19 dataset, since search results are generic, the benefit of tag embedding is less. Note that the tag generation is done only once and therefore takes a fixed amount of time. It took 3.26 seconds, 3.47 seconds, and 1.96 seconds per sentence on average to generate knowledge-based tags for CWMG, CWAL, and COVID-19 datasets respectively. The time that the model takes to inference in presence of the tag embeddings is negligible as compared to the model without these embeddings (see the last column of Table 3). For the supervised models though, the major chunk of time is required for the mention pair generation.

Full system evaluation: So far, the assessment for the two components was carried out separately, i.e., the evaluation for the important sentence extraction was based on Level I annotated data while the evaluation for event coreference resolution was on the basis of Level II annotations independently. We also conduct the full system evaluation for CWMG and CWAL datasets, i.e., the complete evaluation was only dependent on Level II annotated data. For this case we trained the GAN-BERT classifier with 30% of the labeled data along with the unlabeled data (discussed in section 3.3), and had predictions for the rest of 70% data. Now, we consider only the *true positives* (labeled as important, and also predicted important), before performing the coreference resolution. This task is evaluated based on the Level II annotated data. The primary reasons for considering only true positive samples are - (1) we do not have ground-truth Level II annotated data for the non-important sentences (i.e., the false positives), (2) for all practical purposes we are only interested in the coreferences present in the positive predictions (i.e., in the predicted important sentences). Table 4 shows the comparison between the full system evaluation result and the standard result (see Appendix A.11 for results w/o tags). The results shown here are the average value of the four different standard metrics (MUC, B^3, CEAF_E and BLANC) corresponding to the best performing unsupervised model as well as the mention-pair based supervised model.

Comparison with TLS: Since our method has some parallels with TLS, in this section we perform a thorough comparison with state-of-the-art TLS systems. Note that the output of our system is not similar to that of the standard TLS output. In order to make the comparison possible and fair we added a simple summarization step at the end of our pipeline. We used the BERT extractive summarizer (Miller, 2019) to extract the two most important sentences as the summary for each of the event clusters generated by our method. We evaluated the summaries using the alignment-based ROUGE (AR) F-Score (Martschat and Markert, 2017). Unlike (Gholipour Ghalandari and Ifrim, 2020), we did not use any date ranking method to rank the dates of the predicted timeline and compared the ground-truth with the top-k predicted timeline. We tested all the approaches using our Level I annotated data as the ground-truth reference. Table 5 shows the detailed comparison of our approach with few of the existing state-of-the-art TLS approaches on two of our datasets. In order to perform these experiments we considered preselected 41 formal letters from CWMG in the time period 1930-1935 with more than 1000 words and all the documents of volume 2 from CWAL (from which the Level I annotations were performed) and directly passed through the TLS pipeline using the codes provided by the respective authors. In order to make the comparison further fair, we also performed an experiment by first carrying out important sentence classification using our method and then feeding the filtered data into the TLS pipeline provided by the authors. In order to benefit the TLS models the event detection for this pre-filtering was performed using the model fine-tuned on our dataset. This modification results in superior performance of the TLS. In fact, event detection prior to summarization always helps – our method as well as one of the baseline methods (Gholipour Ghalandari and Ifrim, 2020) where event detection can be easily incorporated show significantly[13] improved performance. In Table 11 of Appendix A.8 we also show that this event detection step brings benefits to a standard TLS dataset which has not been built from historical text. The reason for this inferior performance could be that the summary in the standard TLS approaches are highly sensitive to the keywords used for the particular dataset and generating quality keywords for a dataset consisting of diverse events like ours requires domain-expertise (see Table 12 in Appendix A.9).

6 Timeline visualization

Generating a timeline would not be that impactful unless it is visualized in an interpretable and convenient way. We incorporate an elegant visualization for the generated event timelines using *vis-timeline* javascript library (Appendix A.12 shows an example timeline).

Survey: In order to understand the effectiveness of the interface we ran an online crowd-sourced survey. Out of 33 participants with different educational backgrounds, overall 93% agreed that the interface was very useful for summarization of historical timeline of events. 88% participants found some information which would have been hard for them to fathom just by reading the CWMG plaintext (more results in Appendix A.13).

Dataset	Coref-resolution type	methods	R	P	F1
CWMG	Supervised	MA	0.83	0.69	0.72
		MP	0.74	0.63	0.64
	Unsupervised	MA	0.65	0.71	0.68
		MP	0.62	0.65	0.63
CWAL	Supervised	MA	0.82	0.65	0.64
		MP	0.74	0.59	0.60
	Unsupervised	MA	0.60	0.66	0.63
		MP	0.55	0.59	0.57

Table 4: Full system evaluation result. MA: Important sentences obtained through manual annotation, MP: Important sentences obtained from model prediction. Appendix A.11 shows the same results without using tag embeddings.

System	CWMG Dataset		CWAL Dataset	
	AR1-F	AR2-F	AR1-F	AR2-F
MM	0.023	0.001	0.052	0.024
DT	0.008	0.001	0.022	0.002
ED (our) + DT	0.015*	0.006*	0.026*	0.002
CLUST	0.028	0.02	0.055	0.040
ED (our) + CLUST	0.034•	0.025•	0.086•	0.071•
Our method	**0.062†*•**	**0.043†*•**	**0.069†*•**	**0.042†*•**

Table 5: Comparison of our method for the with the existing state-of-the-art TLS methods - (1) MM (submodularity based method): Martschat and Markert (2018) and (2) DT: datewise and (3) CLUST: clustering based TLS by Gholipour Ghalandari and Ifrim (2020), ED: Event detection. †, *, • show that our results are significantly different from MM, ED + DT, ED + CLUST respectively. In turn, any method with ED (*, •) is significantly better than MM.

7 Conclusion

In this work we presented a framework to generate event timeline from any timestamped document. The entire pipeline has two parts – important event detection and event coreference resolution. We achieve very encouraging results for both these tasks. While it is true that our evaluations are based on two historical texts, our methods are generic and can be easily extended to other datasets. The system that we developed is not limited to any actor specific event (human or location) which, in fact, made the coreference resolution task even more challenging. We believe that our work will open up new and exciting opportunities in history research and education.

[13] Statistical significance were performed using Mann–Whitney U test (Mann and Whitney, 1947)

8 Ethical considerations

We have framed our datasets by collecting textual information from publicly available online resources and these do not contain any individual private information. The two historical datasets, i.e., the CWMG and the CWAL have been constructed by using the two specific online sources mentioned in 3.1, while the privacy rights have been acknowledged. The contents in the COVID-19 event dataset are collected from freely accessible Wikipedia and publicly available information from `https://who.int`. Further, the datasets have been annotated by the research scholars and university undergraduate students voluntarily. Finally, in order to avoid concerns of bias in the survey we had 5 expert historians out of the 33 participants. Three among these participants found the information on the timeline fully correct and the other two found it mostly correct. Further four of them agreed that the sentences appeared in the timeline are important for summarizing the life events. Since the observations of the experts align very well with nontechnical audience, we are confident that the accuracy and factuality of the information gathered and shown on the timeline are not misleading.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Sayantan Adak, Atharva Vyas, Animesh Mukherjee, Heer Ambavi, Pritam Kadasi, Mayank Singh, and Shivam Patel. 2020. Gandhipedia: A one-stop ai-enabled portal for browsing gandhian literature, life-events and his social network. In *Proceedings of the ACM/IEEE Joint Conference on Digital Libraries in 2020*, JCDL '20, page 539–540, New York, NY, USA. Association for Computing Machinery.

Alessio Aprosio and Sara Tonelli. 2015. Recognizing biographical sections in wikipedia. pages 811–816.

Amit Bagga and Breck Baldwin. 2000. Entity-based cross-document coreferencing using the vector space model. *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, 1.

David Bamman and Noah A. Smith. 2014. Unsupervised discovery of biographical structure from text. *Transactions of the Association for Computational Linguistics*, 2:363–376.

Shany Barhom, Vered Shwartz, Alon Eirew, Michael Bugert, Nils Reimers, and Ido Dagan. 2019. Revisiting joint modeling of cross-document entity and event coreference resolution.

Harsimran Bedi, Sangameshwar Patil, Swapnil Hingmire, and Girish Palshikar. 2017. Event timeline generation from history textbooks. In *Proceedings of the 4th Workshop on Natural Language Processing Techniques for Educational Applications (NLPTEA 2017)*, pages 69–77, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Leo Born, Maximilian Bacher, and Katja Markert. 2020. Dataset Reproducibility and IR Methods in Timeline Summarization. In *LREC 2020*.

Zheng Chen, Heng Ji, and Robert Haralick. 2009. A pairwise event coreference model, feature impact and evaluation for event coreference resolution. In *Proceedings of the Workshop on Events in Emerging Text Types*, pages 17–22, Borovets, Bulgaria. Association for Computational Linguistics.

Prafulla Kumar Choubey and Ruihong Huang. 2017. Event coreference resolution by iteratively unfolding inter-dependencies among events. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2124–2133, Copenhagen, Denmark. Association for Computational Linguistics.

Danilo Croce, Giuseppe Castellucci, and Roberto Basili. 2020. GAN-BERT: Generative adversarial learning for robust text classification with a bunch of labeled examples. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2114–2119, Online. Association for Computational Linguistics.

Agata Cybulska and Piek Vossen. 2014. Using a sledgehammer to crack a nut? lexical diversity and event coreference resolution. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4545–4552, Reykjavik, Iceland. European Language Resources Association (ELRA).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Abbas Ghaddar and Phillippe Langlais. 2016. Wikicoref: An english coreference-annotated corpus of wikipedia articles. In *Proceedings of the Tenth International Conference on Language Resources and*

Evaluation (LREC 2016), Paris, France. European Language Resources Association (ELRA).

Demian Gholipour Ghalandari and Georgiana Ifrim. 2020. Examining the state-of-the-art in news timeline summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1322–1334, Online. Association for Computational Linguistics.

Marti A. Hearst. 1998. Support vector machines. *IEEE Intelligent Systems*, 13(4):18–28.

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

Kian Kenyon-Dean, Jackie Chi Kit Cheung, and Doina Precup. 2018. Resolving event coreference with supervised representation learning and clustering-oriented regularization.

Ashraf M. Kibriya, Eibe Frank, Bernhard Pfahringer, and Geoffrey Holmes. 2004. Multinomial naive bayes for text categorization revisited. In *Proceedings of the 17th Australian Joint Conference on Advances in Artificial Intelligence*, AI'04, page 488–499, Berlin, Heidelberg. Springer-Verlag.

Moreno La Quatra, Luca Cagliero, Elena Baralis, Alberto Messina, and Maurizio Montagnuolo. 2021. *Summarize Dates First: A Paradigm Shift in Timeline Summarization*, page 418–427. Association for Computing Machinery, New York, NY, USA.

Yaojie Lu, Hongyu Lin, Jialong Tang, Xianpei Han, and Le Sun. 2020. End-to-end neural event coreference resolution.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics.

H. B. Mann and D. R. Whitney. 1947. On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *The Annals of Mathematical Statistics*, 18(1):50 – 60.

Sebastian Martschat and Katja Markert. 2017. Improving ROUGE for timeline summarization. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 285–290, Valencia, Spain. Association for Computational Linguistics.

Sebastian Martschat and Katja Markert. 2018. A temporally sensitive submodularity framework for timeline summarization. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 230–240, Brussels, Belgium. Association for Computational Linguistics.

Derek Miller. 2019. Leveraging bert for extractive text summarization on lectures.

Takeru Miyato, Andrew M. Dai, and Ian Goodfellow. 2017. Adversarial training methods for semi-supervised text classification.

Nafise Sadat Moosavi and Michael Strube. 2016. Which coreference evaluation metric do you trust? a proposal for a link-based entity aware metric. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 632–642, Berlin, Germany. Association for Computational Linguistics.

Girish Palshikar, Sachin Pawar, Sangameshwar Patil, Swapnil Hingmire, Nitin Ramrakhiyani, Harsimran Bedi, Pushpak Bhattacharyya, and Vasudeva Varma. 2019. Extraction of message sequence charts from narrative history text. In *Proceedings of the First Workshop on Narrative Understanding*, pages 28–36, Minneapolis, Minnesota. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Sabarmati Ashram Preservation and Memorial Trust. 2013. The Collected Works of Mahatma Gandhi. `https://www.gandhiheritageportal.org/the-collected-works-of-mahatma-gandhi`. [Online; accessed 22-February-2020].

James Pustejovsky, José Castaño, Robert Ingria, Roser Saurí, Rob Gaizauskas, Andrea Setzer, Graham Katz, and Dragomir Radev. 2003. Timeml: Robust specification of event and temporal expressions in text. pages 28–34.

M. Recasens and Eduard Hovy. 2011. Blanc: Implementing the rand index for coreference evaluation. *Natural Language Engineering*, 17:485 – 510.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks.

Jannik Strötgen and Michael Gertz. 2010. HeidelTime: High quality rule-based extraction and normalization of temporal expressions. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 321–324, Uppsala, Sweden. Association for Computational Linguistics.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. pages 45–52.

W. Zhang, Q. Chen, and Y. Chen. 2020. Deep learning based robust text classification method via virtual adversarial training. *IEEE Access*, 8:61174–61182.

Ye Zhang and Byron Wallace. 2016. A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification.

A Appendices

A.1 Example timeline of events

The method that the we propose can generate a timeline as shown in Figure 1. This can be remarkably helpful to recognize the context and the actors of a particular event in a certain period.

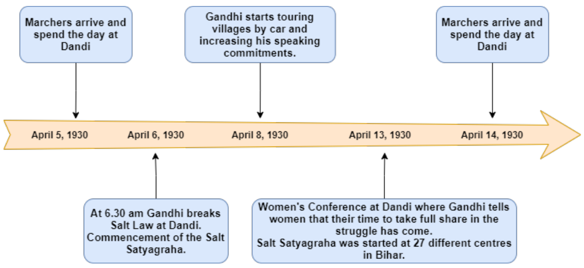

Figure 1: Sample event timeline example extracted from documents.

A.2 Schematic of our method

Figure 2 shows the different steps constituting our over methodology.

A.3 Sample annotations

Table 6 shows the examples of Level I annotated data (sentence classification) and Table 7 illustrates Level II annotated data (coreference resolution) for some portions in the CWMG dataset.

doc_id	publication date	time	sentence	importance	type
volume43_book_393	1930-05-04T00:00:00+00:00	1930-05-04T00:00:00+00:00	The public have been told that Dharasana is private property .	1	fact
volume43_book_393	1930-05-04T00:00:00+00:00	1930-05-04T00:00:00+00:00	This is mere camouflage .	1	fact
volume43_book_393	1930-05-04T00:00:00+00:00	1930-05-04T00:00:00+00:00	It is as effectively under Government control as the Viceroy 's House .	1	fact
volume43_book_393	1930-05-04T00:00:00+00:00	1930-05-04T00:00:00+00:00	Not a pinch of salt can be removed without the previous sanction of the authorities .	1	fact
volume43_book_393	1930-05-04T00:00:00+00:00	1930-05-04T00:00:00+00:00	It is possible for you to prevent this raid , as it has been play- fully and mischievously called , in three ways :	0	None
volume43_book_393	1930-05-04T00:00:00+00:00	1930-05-04T00:00:00+00:00	by removing the salt tax ; 1 The letter was drafted on the eve of Gandhiji 's arrest .	0	None
volume43_book_393	1930-05-04T00:00:00+00:00	1930-05-05T00:00:00	He was arrested at 12.45 a.m. on May 5 .	1	event
volume43_book_393	1930-05-04T00:00:00+00:00	1930-05-04T00:00:00+00:00	THE COLLECTED WORKS OF MAHATMA GANDHI 2 . by arresting me and my party unless the country can ,	0	None

Table 6: Sample Level I annotation of CWMG dataset.

sentence	importance	type	cluster
The public have been told that Dharasana is private property .	1	fact	1
This is mere camouflage .	1	fact	1
It is as effectively under Government control as the Viceroy 's House .	1	fact	1
Not a pinch of salt can be removed without the previous sanction of the authorities .	1	fact	1
It is possible for you to prevent this raid , as it has been play- fully and mischievously called , in three ways :	0	None	None
by removing the salt tax ; 1 The letter was drafted on the eve of Gandhiji 's arrest .	0	None	None
He was arrested at 12.45 a.m. on May 5 .	1	event	2

Table 7: Sample Level II annotation of CWMG dataset. We only marked the cluster value for the sentences which are marked as important by at least 2 annotators during the level I annotation.

A.4 Category distribution

Classes	Count	
	CWMG	CWAL
event/fact	716	242
demand	81	96
other	268	382

Table 8: Category distribution for the two datasets.

A.5 Annotator agreement using different metrics for Level II annotated samples

Dataset	Metric			
	MUC	B^3	CEAF_E	BLANC
CWMG	0.74	0.72	0.65	0.77
CWAL	0.61	0.54	0.55	0.59
COVID-19	0.78	0.81	0.71	0.74

Table 9: Annotator agreement (F1 score) for Level II annotated data using different metrics.

A.6 Details of tag creation method

The generation of tags from world knowledge for a particular sentence is an important part of our pipeline, which contain the manual filtering part. We take the sentence as query, and by using *google-search* api we obtain the top 5 retrieved urls and scrape the texts from these. We also consider the original document from where the sentence is being extracted (for COVID-19 data document this is not present) to gather additional context. Based on the internet connectivity, server response time, number of results per page it can take from 1 second to up to a maximum of 30 seconds for scraping the texts from web for each of the sentences. Then we use three methods (*TextRank*, *Rake*, and *pointwise mutual information*) to collect top 5 bigrams (we

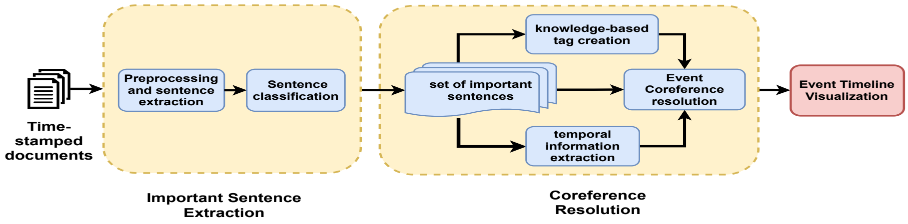

Figure 2: The overall architecture for generating the event timeline.

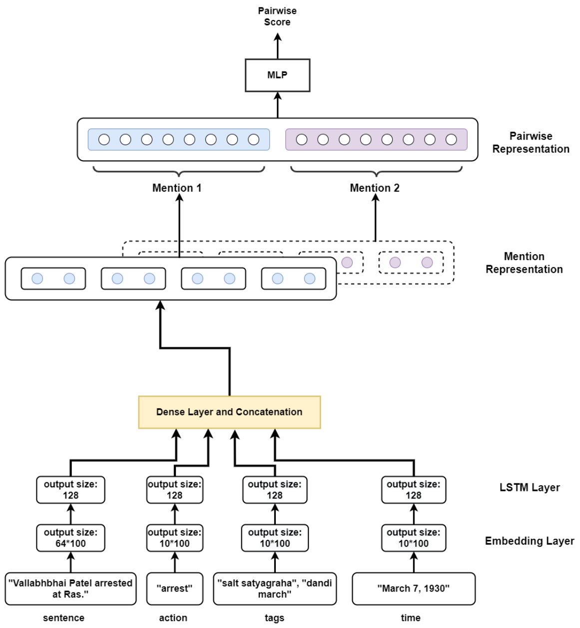

Figure 3: An illustration of the Event mention-pair model.

observed bigrams provide most relevant results) by each of the methods. During the process we automatically filter the stop words, and consider the bigrams which belong to one of the following POS categories - *'JJ', 'JJR', 'JJS', 'NN', 'NNS', 'NNP', 'NNPS'*. The parts of speech tags are determined using *nltk pos_tag* module. Table 10 shows examples of top 5 tags generated for a sentence by each of the three above methods.

A.7 Architecture diagram of supervised mention-pair model

Figure 3 represents the model architecture, which is inspired from Barhom et al. (2019).

A.8 Effectiveness of event detection in TLS task

Table 11 shows how our event detection step improves the performance for a standard TLS dataset also which has not been built from historical text.

System	Timeline17 Dataset	
	AR1-F	AR2-F
MM	0.105	0.03
DT	0.12	0.035
ED (our) + DT	0.122	0.039*
CLUST	0.082	0.02
ED (our) + CLUST	0.085•	0.026•

Table 11: Comparison of the performance with and without incorporating our event detection step for the TLS task on a standard TLS dataset. TLS methods used are – (1) MM (submodularity based method): Martschat and Markert (2018) and (2) DT: datewise and (3) CLUST: clustering based TLS by Gholipour Ghalandari and Ifrim (2020). ED: Our event detection method. † denotes significant[14] improvement over Martschat and Markert (2018), * over DT, and • over CLUST.

A.9 Example summaries

In Table 12 we present a few examples comparing the summaries produced by our method vis-a-vis the approach outlined in using (Gholipour Ghalandari and Ifrim, 2020). The blue portions indicate the parts that are present in the ground-truth.

A.10 Event coreference resolution results without manual filtering of tags

Table 13 shows result obtained from different coreference resolution techniques when we do not include any manual filtering steps to the generat tags. It can be noticed that there is not much difference in the results even when we omit this step.

A.11 Full system evaluation without tags

Table 14 shows the coreference resolution results for the full system using both supervised (event mention-pair model) and unsupervised (s-bert + agglomerative clustering) methods without using external tag embeddings.

Dataset	Coref-resolution type	methods	R	P	F1
CWMG	Supervised	MA	0.76	0.65	0.68
		MP	0.62	0.55	0.52
	Unsupervised	MA	0.55	0.56	0.55
		MP	0.41	0.42	0.41
CWAL	Supervised	MA	0.74	0.62	0.66
		MP	0.48	0.56	0.51
	Unsupervised	MA	0.46	0.48	0.47
		MP	0.31	0.30	0.31

Table 14: Full system evaluation result without tags. MA: Important sentences obtained through manual annotation, MP: Important sentences obtained from model prediction.

Sentence	method	example tags
Paddy fields are reported to have been burnt, eatables forcibly taken.	TextRank	government notices', 'government control', 'non violence', 'private salt', 'young india'
	Rake	without hesitation', 'victims success', 'viceroy house', 'unthinkable cruelties', 'unnecessary bones'
	Pointwise Mutual Information	civil disobedience', 'salt tax', 'civil resisters', 'TO VICEROY', 'satyagraha programme'

Table 10: Examples of generated tags.

[1930-04-06] I feel you are right in confining your attention to the salt tax for the time being .	[1930-05-04] In Karachi , Peshawar and Madras the firing would appear to have been unprovoked and unnecessary . Bones have been broken , private parts have been squeezed for the pur- pose of making volunteers give up , to the Government valueless , to the volunteers precious salt .
[1930-04-30] The addressee had been arrested on April 30 , 1930 , during the Vedaranyam Salt Satyagraha . In reply to the addressee 's letter regarding the order of the Madras Government permitting the collector of Tanjore to prosecute the satyagrahis breaking the salt law in the South 2	[1930-04-11] After returning from the Assembly work at Delhi I immediately held confe- rence of Maharashtra National Party and have decided to start and organ-ise
[1930-04-14] I got the book about salt which you sent with Keshavram	[1930-04-14] It is 10.30p.m. Jawahar has also been arrested .Pandya , Ghia and others have been arrested here .If things continue to move with the present velocity , he wo n't have even six months ' rest .I never expected this phenomenal res- ponse.

Table 12: Sample summary generated using (Gholipour Ghalandari and Ifrim, 2020) (left) and our method (right) on the CWMG dataset. Text in blue indicates the portion present in the ground-truth timeline.

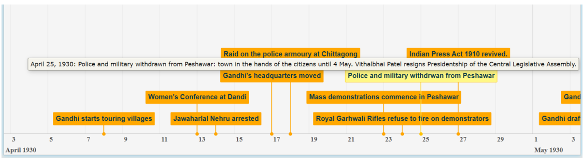

Figure 4: Sample visualization of timeline generated from the CWMG dataset.

A.12 Sample timeline

After resolving the event coreference, the generated data is used to create the timeline. In order to generate the title for a specific event, we have used BERT extractive summarizer (Miller, 2019). The idea of visualisation was to make the tool accessible to historians as well as run a survey of the utility of the tool in the first place. Figure 4 shows a sample event timeline generated by the tool from the CWMG dataset.

A.13 Online survey

In the survey we asked participants a number of questions regarding the readability, correctness and relevance about the information in the generated timeline. 33 participants with various educational backgrounds took part in the survey. 79% of the participants noted that the interface was easily readable. 73% of the total participants reported that they were very satisfied with the overall quality of the automatically generated event timeline summaries.

Dataset	System	MUC	B^3	CEAF_E	BLANC	Avg (overall)		
		F1	F1	F1	F1	Recall	Precision	F1
CWMG	tf-idf + GM	0.61	0.55	0.51	0.58	0.62	0.57	0.56
	tf-idf + AC	0.64	0.59	0.51	0.66	0.58	0.64	0.60
	s-bert + GM	0.68	0.61	0.44	0.63	0.62	0.60	0.59
	s-bert + AC	0.76	0.71	0.50	0.72	0.65	0.72	0.67
	mention-pair model	0.92	0.61	0.85	0.53	0.85	0.70	0.73
CWAL	tf-idf + GM	0.76	0.51	0.44	0.65	0.55	0.59	0.59
	tf-idf + AC	0.75	0.50	0.49	0.65	0.56	0.63	0.59
	S-bert+ GM	0.76	0.40	0.35	0.69	0.51	0.59	0.55
	s-bert + AC	0.81	0.59	0.47	0.70	0.63	0.72	0.64
	mention-pair model	0.95	0.43	0.76	0.36	0.81	0.67	0.62
COVID-19	tf-idf + GM	0.40	0.33	0.26	0.55	0.39	0.44	0.38
	tf-idf + AC	0.42	0.35	0.34	0.43	0.41	0.39	0.38
	s-bert + GM	0.56	0.43	0.36	0.57	0.44	0.49	0.48
	s-bert + AC	0.65	0.44	0.37	0.59	0.52	0.50	0.51
	mention-pair model	0.95	0.93	0.93	0.95	0.93	0.92	0.94

Table 13: Event coreference results without using manual filtering for the tags. GM: Gaussian Mixture based clustering; AC: Agglomerative Clustering; s-bert: sentence-bert. The results mostly remain unaffected.