Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020), pages 1763–1771
Marseille, 11–16 May 2020

Dataset Reproducibility and IR Methods in Timeline Summarization

Leo Born, Maximilian Bacher, Katja Markert
Department of Computational Linguistics
Heidelberg University
69120 Heidelberg, Germany
{born, bacher, markert}@cl.uni-heidelberg.de

Abstract

Timeline summarization (TLS) generates a dated overview of real-world events based on event-specific corpora. The two standard datasets for this task were collected using Google searches for news reports on given events. Not only is this IR method not reproducible at different search times, it also uses components (such as document popularity) that are not always available for any large news corpus. It is unclear how TLS algorithms fare when provided with event corpora collected with varying IR methods. We therefore construct event-specific corpora from a large static background corpus, the *newsroom* dataset, using differing, relatively simple IR methods based on raw text alone. We show that the choice of IR method plays a crucial role in the performance of various TLS algorithms. A weak TLS algorithm can even match a stronger one by employing a stronger IR method in the data collection phase. Furthermore, the results of TLS systems are often highly sensitive to additional sentence filtering. We consequently advocate for integrating IR into the development of TLS systems and having a common static background corpus for evaluation of TLS systems.

Keywords: timeline summarization, dataset construction, reproducibility

1. Introduction

When real-world events get reported on, coverage can be quite exhaustive. This can either be due to the importance of the event or the duration of it. *Timelines* (see Table 1) provide a brief, dated overview of such events. In order to efficiently construct a timeline for a given event, *timeline summarization* (TLS) systems can be employed.

Timeline summarization is a task with similarities to *multi-document summarization* (MDS). However, solving TLS requires generating a temporal framework for the summary, i.e. important dates have to be identified and corresponding date summaries have to be generated. In addition, the number of input documents for TLS is normally an order of magnitude larger than for traditional MDS systems (hundreds instead of tens).

TLS systems use two input components for a given event E (such as the BP oil spill): an input corpus C_E that contains news documents covering E and human-written timelines T_E (such as the one in Table 1) for evaluation (and possibly training). To constrain C_Es to news articles that only cover E, various IR methods have been employed. To this end, multiple corpora for evaluation of the task have been created over the past decade. Unfortunately, most of the resulting corpora are not publicly available. In addition, the only two publicly available corpora *TL17* and *crisis* (Tran et al., 2013b; Tran et al., 2015a) have been collected via the use of a commercial search engine at a given point in time. This means that the method is not reproducible and there are potential copyright problems with regard to the collected texts (see also Kilgarriff (2007) for further criticism of using commercial search engines). There have been no empirical studies on the effects of different IR methods and the resulting C_E on TLS. Instead, most prior work has focused on improving summarization systems, taking the corpora – and therefore the underlying IR methods – for granted. The systems run the risk of optimizing dataset-specifically by overly relying on the provided IR.

2010-04-20
Deepwater Horizon drilling rig explodes about 42 miles off Louisiana, killing 11 men.
2010-04-22
The rig, having burned and been showered with water during firefighting efforts, sinks. The force of the sinking breaks off the rig's drillpipe, allowing oil to spew out into the gulf.
2010-05-02
The federal government closes 3 percent of federal waters in the gulf to fishing.

Table 1: Excerpt of a timeline on the BP oil spill by the Washington Post (from the *TL17* corpus provided by (Tran et al., 2013a; Tran et al., 2013b)).

We therefore aim to move away from existing benchmark datasets and provide reproducible datasets (*newsroomTLS*) to evaluate TLS systems. Our contributions are threefold:

1. We create reproducible TLS input corpora from a large, static background corpus. This allows comparison of IR methods and performance of TLS on different event-specific corpora and facilitates the integration of IR into TLS. The data and tools are publicly available at `https://gitlab.cl.uni-heidelberg.de/newsroomtls/newsroomtls`.

2. We are therefore the first to explicitly investigate the influence of IR on TLS as most prior work has relied on IR as being given. We compare the impact of three simple, yet successively stronger, IR methods on three established TLS algorithms. We show that the IR impact is significant for all algorithms and that a weak TLS algorithm with strong prior data collection can rival a strong TLS algorithm with weaker IR.

3. Most existing TLS algorithms also employ additional sentence filters on the provided corpora, using manually determined event-specific keywords. We are the first to investigate the effect of this sentence filtering. We show that the performance of existing algorithms is highly dependent on these manually determined filters, indicating that much more effort needs to be put into stabilizing TLS architectures, rather than optimizing performance without regarding IR and filtering.

2. Related Work

Previous TLS research (Chieu and Lee, 2004; Yan et al., 2011; Kessler et al., 2012; Tran et al., 2013a; Tran et al., 2013b; Tran et al., 2015a; Tran et al., 2015b; Wang et al., 2015; Wang et al., 2016b; Martschat and Markert, 2017; Martschat and Markert, 2018; Liang et al., 2019; Barros et al., 2019) concentrates on extracting and dating informative and non-redundant sentences from an event-specific corpus C_E to construct a summary. The information in the corresponding papers on the construction of C_E, given an event E of interest, is scarce, with some papers (Yan et al., 2011) not explaining their method at all. We can overall distinguish two approaches: papers that built their own text-based IR system using keywords and indexing, vs. papers that relied on commercial search engines or corpora built by other researchers with search engines.

Keyword-based/Indexing IR. Chieu and Lee (2004) constructed timelines on activities of G8 leaders and built C_Es by extracting articles from the English Gigaword corpus[1] between January and June 2002 using leader names as simple keywords. Similarly, Wang et al. (2015) also used keyword search with entity names (such as *Ukraine*) to extract articles from the New York Times related to a specific event. Nguyen et al. (2014) used the Lucene[2] search engine to extract articles from the AFP corpus.

None of the resulting corpora are publicly available and the impact of keyword or search method choice on TLS has not been investigated. While name keyword search is attractive due to simplicity and potentially high recall, it might lack precision, especially on large background corpora C_B.

Commercial search engines. A different approach to finding relevant articles is to use commercial search engines like Google to extract event-specific articles from the internet instead of a static background corpus. (Tran et al., 2013a; Tran et al., 2013b) introduced the *TL17* dataset. For 17 human-generated timelines taken from news sites, they extracted 400 news articles highly ranked by Google per event, yielding 4,650 articles overall after duplication removal. Similarly, (Tran et al., 2015a; Tran et al., 2015b) provided a new dataset extracted via Google with essentially the same method on four events, the *crisis* dataset. Both corpora have been made publicly available, yielding potential benchmark sets containing both human-generated timelines and event-specific corpora. They have been used as evaluation datasets in subsequent work (Suzuki and Kobayashi, 2014; Wang et al., 2016a; Wang et al., 2016b; Martschat and Markert, 2018; Liang et al., 2019; Barros et al., 2019). However, there are no investigations on the quality of the resulting event corpora or the impact that changes to these corpora would have on the TLS systems (such as extracting more or fewer articles).[3]

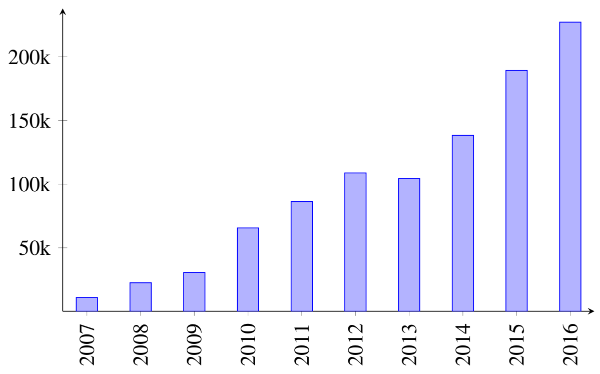

Figure 1: Distribution of articles per year from 2007 to 2016 in the training portion of *newsroom*.

Commercial search engines are sophisticated IR tools where we can expect high precision in the top matches. In comparison to keyword-based approaches, they can utilize a diverse array of additional meta and network information. However, using Google obviates the reproducibility of the dataset generation since Google not only personalizes search results, but also exhibits bias dependent on time and location of search. Using it to collect articles is therefore highly dependent on factors outside the scope of researchers. In addition, there are potential copyright problems with regard to the collected texts. Lastly, the method is not applicable to a given static background corpus (for example a corpus that a news provider might have).

Online summarization. A research field related to TLS is *online* or *update summarization* (Kedzie et al., 2015; Kedzie et al., 2016). Here summaries are generated whilst an event is ongoing, often from highly redundant social media streams. A standard dataset comes from the TREC Temporal Summarization track (Aslam et al., 2013; Aslam et al., 2014; Aslam et al., 2015). Relevance detection here plays the role of IR and is paid more heed than IR is in standard TLS. Apart from being in an offline setting, TLS normally generates timelines for much longer running events than update summarization and has a higher emphasis on dating events.

3. Background Corpus

Our main goal is to facilitate a common practice of corpus generation using static background corpora. This necessitates integrating IR into TLS systems as they would not be able to rely on any provided IR. This is, however, a realistic scenario as it cannot be guaranteed that a TLS system will be used on a pre-filtered dataset. Our proposal thus harkens back to previous work in TLS (Chieu and Lee,

[1]`http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2003T05`.

[2]`https://lucene.apache.org`.

[3]Even a brief investigation by us found that both corpora include artefacts such as parts of gold-standard timelines in the input C_E corpora.

event	# of timelines	event range	keywords
bpoil	5	20/04/2010 - 25/11/2012	bp, oil, spill { news, obama, company, drilling, president }
ebola	7	26/12/2013 - 29/12/2015	ebola { health, news, disease, hospital, state }
egypt	5	01/01/2011 - 07/07/2013	egypt, crisis { president, state, government, news, military }
finan	1	07/09/2008 - 12/12/2008	financial, crisis { market, news, bank, government, year }
global08	3	03/01/2007 - 03/08/2009	financial, crisis { news, market, bank, year, government }
greece	1	02/05/2010 - 09/02/2012	greece, crisis { debt, government, european, market, bank }
h1n1	3	18/03/2009 - 04/05/2009	h1n1, swine flu { government, health, year, mexico, virus }
haiti	1	12/01/2010 - 23/01/2010	haiti, earthquake { haitian, aid, news, relief, help }
libya	7	14/02/2011 - 22/11/2011	libya, war { military, government, president, libyan, state }
snowden	4	31/03/2012 - 31/05/2014	snowden { security, government, intelligence, surveillance, nsa }
swineflu	3	31/01/2009 - 10/08/2010	h1n1, swine flu { health, vaccine, disease, news, virus }
syria	9	16/02/2011 - 06/07/2013	syria, war { president, government, syrian, state, military }
ukraine	5	21/11/2013 - 10/02/2016	ukraine, war { russia, russian, president, state, military }
wikileaks	1	05/04/2010 - 05/12/2011	wikileaks { news, government, state, security, report }
yemen	6	22/01/2011 - 27/02/2012	yemen, war { president, government, state, arab, military }
total	61	03/01/2007 - 10/02/2016	

Table 2: Statistics on the events under consideration. *event range* refers to the first and last dates of the union of all timelines per event. Keywords outside of brackets are manually determined, the ones in brackets are those extracted by bootstrapping (see Section 5. for details).

2004; Nguyen et al., 2014; Wang et al., 2015) that generated input corpora from static background corpora. This methodology, however, has been discarded by the majority of TLS research in favor of using pre-filtered datasets.

We also investigate the impact IR methods have on TLS performance. In particular, given an event of interest E, we want to create systematically different event-specific input corpora C_E from a large, static background corpus C_B in a reproducible fashion. This allows us to evaluate the performance of different TLS systems, given different input C_Es.

As our background corpus to draw articles from, we used the *newsroom* dataset (Grusky et al., 2018). Not only has it already been used to evaluate single-document summarization systems (see e.g., Shi et al. (2019) or Mendes et al. (2019)), but it also provides a large amount of data to experiment with different IR methods – this makes results much more reproducible since the background corpus is always fixed. The corpus contains approx. 1.3M articles, covering the years 1998 to 2016.

One of the key reasons why we opted to use a different dataset than other TLS work lies in the way previous benchmark datasets have been generated. As described in Section 2., the most commonly used datasets, *crisis* and *TL17*, have been constructed via very sophisticated IR – namely, Google search. However, while this provides clear benefits from an IR perspective, it also means that the dataset construction is irreproducible (Kilgarriff, 2007). Since Google searches are dependent on time of search, user account, and user location, there is no way of assessing and backtracing the influence of IR on the datasets, and therefore on the TLS algorithms operating on these datasets.

We extracted our event-specific corpora C_E from the training portion of *newsroom*, containing 994,080 articles.[4] The distribution of articles per year (between 2007 and 2016) is given in Figure 1.

[4]When we started this work, *newsroom* was only provided in link form, i.e articles had to be scraped with a provided script. A small number of articles (961) could not be downloaded.

4. Events

The events E we evaluate TLS systems on need to fulfil certain criteria: availability of human-written timelines for evaluation, sufficient coverage in *newsroom* and preferably also overlap with events previously used in timeline summarization.

We collected all events and timelines from the *crisis* and *TL17* corpora as well as new events taken from Feldhus (2016). From those datasets, we excluded all events that matched any of the following constraints:

- **The event starts before 2007**. *newsroom* contains less than 10,000 articles per year (on *all* topics) for years before 2007.
- **Any event lasting for longer than three years**. Since the number of articles per year increases over time in *newsroom* (see Figure 1), including very long-running events might introduce a bias towards later articles.
- Additionally, the event “Michael Jackson death” was excluded due to an erroneous date (*2011-04-0*) in the reference timeline.

Table 2 shows the events we consider, including the number of reference timelines we have for each and when the event took place.

5. IR Methods

In order to judge the influence of IR systems on TLS performance, we applied three different IR methods. These IR methods do not constitute the state-of-the-art by any means; however, they are successively stronger methods, which allows for investigating the impact of IR methods along the spectrum of {weak, neutral, strong}. Since we are not interested in finding the *best* IR for TLS, we opted for three commonly used methods. They are all purely text-based and do not need access to meta- or network data.

event	# of documents	N	overlap (# of documents)		
			$simple$-$bm25$	$simple$-$bm25_{boot}$	$bm25$-$bm25_{boot}$
bpoil	236,170	515 (0.22%)	444 (86.21%)	347 (67.37%)	358 (69.51%)
ebola	328,553	1,987 (0.60%)	1,985 (99.89%)	1,469 (73.93%)	1,469 (73.93%)
egypt	244,837	81 (0.03%)	65 (80.24%)	18 (22.22%)	15 (18.51%)
finan	7,037	178 (2.53%)	165 (92.69%)	98 (55.05%)	100 (56.17%)
global08	50,092	361 (0.72%)	325 (90.02%)	163 (45.15%)	166 (45.98%)
greece	146,849	109 (0.07%)	82 (75.22%)	43 (39.44%)	42 (38.53%)
h1n1	3,761	3 (0.08%)	3 (100.0%)	3 (100.0%)	3 (100.0%)
haiti	1,421	71 (5.00%)	70 (98.59%)	52 (73.23%)	51 (71.83%)
libya	66,358	136 (0.20%)	113 (83.08%)	44 (32.35%)	42 (30.88%)
snowden	236,613	906 (0.38%)	903 (99.66%)	660 (72.84%)	660 (72.84%)
swineflu	66,454	47 (0.07%)	18 (38.29%)	20 (42.55%)	28 (59.57%)
syria	234,617	278 (0.12%)	219 (78.77%)	57 (20.50%)	55 (19.78%)
ukraine	357,564	262 (0.07%)	211 (80.53%)	35 (13.35%)	34 (12.97%)
wikileaks	132,201	451 (0.34%)	450 (99.77%)	294 (65.18%)	294 (65.18%)
yemen	99,688	19 (0.02%)	14 (73.68%)	8 (42.10%)	8 (42.10%)

Table 3: Overlap on *newsroom* train data between the three IR methods for all events. **# of documents** = total number of documents in event range; N = number of documents found with *simple* IR method. Percentages in column N are in relation to the total number of documents.

simple. The most basic approach, *simple*, retrieves an article if all keywords occur either in the headline or the first two sentences of it. As *simple* is based on a binary decision for every article, the size of C_E per event can differ. This approach is essentially the one used by Chieu and Lee (2004) to create their dataset. The keywords were determined manually in order to cover the most essential information on an event (see Table 2). We chose to prioritize precision over recall to avoid introducing noise into the event-specific corpora C_E as some keywords are very broad (e.g. *crisis*).

bm25. Okapi *bm25* (Robertson and Zaragoza, 2009) is an IR approach based on term and document frequencies. For a given event-specific keyword set, we rank all articles in the event range based on the keywords and extract the top n articles. In contrast to *simple*, *bm25* considers the full text of an article and retrieves the same number of articles n for every event (as long as $n < C_B$). Formally, *bm25* assigns a score w_i^d to a document d with regard to a keyword term i as follows:

$$w_i^d = \frac{tf}{k_1((1-b)+b\frac{dl}{avdl})+tf} + w_i^{IDF}, \qquad (1)$$

where k_1 and b are free smoothing parameters, dl denotes the document length, $avdl$ average document length, and w_i^{IDF} the weighted inverse document frequency for a keyword i. The full document score is the sum of the term weights given in Equation 1 over all keywords.

bm25$_{boot}$. Our last approach is to use keyword bootstrapping in combination with *bm25*. For an event E, we collect all *newsroom* articles from the event range that include **all** keywords (C_E^{all}).[5] We then use *TextRank* (Mihalcea and Tarau, 2004) for the extraction of new keywords. For each document from the pre-selected article collection C_E^{all}, *TextRank* returns a list of keywords. We filter this list to include only nouns and proper names. Additionally, keywords and keyword tokens are excluded if they are already present in the initial keyword set. We use the five keywords that have been found in the most documents and append them to the original keyword set. This new keyword set is then used to rank articles via *bm25* (see above) to generate our final event-specific corpus C_E (top n articles). Table 2 shows both the initial as well as bootstrapped keywords.

event	simple	bm25	bm25$_{boot}$	Kappa
ebola	0.43	0.38	**0.85**†,‡	0.64
global08	0.42	0.52	**0.62**†	0.55
snowden	0.52	**0.85**†	0.82†	0.44
syria	0.68	0.63	**0.87**†,‡	0.64

Table 4: Manual relevance evaluation on 360 articles of four topics to assess their topic relevance. Highest precision values per event are marked in bold. † indicates statistical significance wrt to *simple* and ‡ wrt to *bm25* (all $p = 0.05$).

5.1. Manual Relevance Evaluation

In order to assess IR performance, we manually evaluated the topical relevance of extracted articles. We selected four events with more than 100 documents found by the *simple* method at random and then annotated 90 articles per event as being related to the event or not. The 90 articles were composed of 30 articles per IR method.[6] The two first authors of the paper conducted the annotation, following guidelines based on the TREC annotation guidelines (Sormunen, 2002), while also providing additional guidance for specific problems pertaining to the dataset (see Appendix). The results in Table 4 show that using a stronger IR method (*bm25$_{boot}$*) yields consistently high results, while the other two systems achieve lower precision and are similar to each

[5]We favor precision over recall here, so that our seed corpus for keyword extraction contains only articles highly related to the event.

[6]For *bm25* and *bm25$_{boot}$*, we simply annotated the 30 top-ranked articles. For *simple*, we randomly chose 30 of the extracted articles.

other.[7] Only for one event, *snowden*, does the weaker *bm25* IR method perform better than the stronger one. However, this difference is not statistically significant.

5.2. Overlap of Retrieved Articles

Table 3 shows the overlap of retrieved articles per pair of IR methods for each event. The overlap column compares retrieval performance with regard to the number of articles extracted by the *simple* IR method, denoted by N. Given the total number of documents for each event in the event range, we can see that the *simple* method deems between 0.02% and 5% of articles per event as relevant.

With the *bm25* and *bm25*$_{boot}$ methods, we extracted the same number of documents N as retrieved by *simple* and checked overlap between the three methods. As we can see, overlap between *simple* and *bm25* is high and in most cases much higher than between *simple* and *bm25*$_{boot}$ (with the exception of the event *swineflu*). This indicates that *per se bm25* does not find different articles from *simple* (however, they might be in a more relevant order than with *simple*), but that *bm25*$_{boot}$ indeed seems to diverge more from the *simple* method.

We also compared overlap between *bm25* and *bm25*$_{boot}$ when extracting $n \in \{1000, 2000, 5000\}$ articles. While for some events, the more articles were extracted, the lower overlap (relatively speaking) became,[8] generally, the overlap ratios between *bm25* and *bm25*$_{boot}$ were similar to what is reported in Table 3. This points to an inherent difference in the retrieval performance of *bm25* and *bm25*$_{boot}$ that is stable across different corpus sizes.

6. TLS Algorithms

In order to assess the impact of IR on TLS, we chose three TLS algorithms of different strengths to run our experiments with. While there have been a plethora of different TLS algorithms in the past, *tilse* (Martschat and Markert, 2018) constitutes the state-of-the-art on the established datasets. The remaining two algorithms therefore needed to be weaker, yet successively stronger baselines.

We compared the following algorithms: *Chieu* (Chieu and Lee, 2004), *regression*, and the best-performing system by Martschat and Markert (2018). The last is denoted as *tilse* and specifically is the algorithm labeled TLSConstraints+$f_{DateRef}$+reweighting in their paper. We use the implementation of all three algorithms as provided by the *tilse* package.[9]

***Chieu*.** Chieu and Lee (2004) present an unsupervised system. Sentences are ranked using two metrics: *interest* and *burstiness*. Based on this ranking and additional constraints preventing redundancy, the system chooses sentences for every date in a timeline.

***regression*.** Based on the work by Tran et al. (2013a) and Wang et al. (2015), *regression* is a supervised linear regression model. The implementation provided by *tilse* represents sentences by features including length, number of named entities, unigram features, and averaged/summed tf-idf scores. The system is trained using the reference timelines, by computing the F_1 score of each sentence to the reference timeline. Constraints are added to keep temporal coherence. During prediction, the system greedily generates a timeline by predicting the F_1 scores of sentences.

***tilse*.** Martschat and Markert (2018) present a system for TLS based on submodular functions. Submodular functions have previously been used in MDS. Based on the coverage and diversity functions for MDS introduced by Lin and Bilmes (2011), they introduce similar functions for TLS. In addition, a date selection function measures the importance of a date by the number of sentences referring to it. These three functions are then combined in an unweighted fashion into one objective function. Two constraints, the maximum number of dates in the timeline and the maximum number of sentences for a date, are set as well. In their experiments, *tilse* outperformed both *regression* and *Chieu* on the established datasets *TL17* and *crisis*.

6.1. Sentence Filtering

Since Chieu and Lee (2004), TLS algorithms (including *tilse*) usually employ additional sentence filters as well. These filters act as a second level of keyword-based IR during algorithm runtime. Given an event-specific corpus C_E, the sentence filter will discard any sentence that does not contain any pre-defined keyword for that event. This way, the algorithm only ever gets presented with a limited selection of sentences to choose from to generate a timeline.

This filter seems to be introduced to reduce the number of sentences an algorithm has to handle for efficiency reasons, but also will be a form of additional relevance/importance criterion. However, it also means that the algorithms will be limited in the amount of information they can select. We are the first to investigate the impact the sentence filter has on performance.

7. Experiments

We investigate two separate issues related to corpus generation and TLS: direct IR impact and sentence filtering impact. While both experiments seek to determine which algorithm/IR combination yields best performance, the first setup (Setup I) employs sentence filtering, whereas the second (Setup II) does not.

Setup I aims to investigate the impact of IR on the three TLS algorithms presented in Section 6. We follow the standard procedure of previous TLS work in that we use the sentence filter as implemented in the package *tilse* throughout. Setup II explicitly drops the sentence filter, while leaving the remaining parameters (corpora, algorithm settings) unchanged. In doing so, we are able to isolate the effect this second level of IR has on the algorithms' performance.

7.1. Constructing C_Es

We collect an event-specific corpus C_E for each E with the IR methods discussed in Section 5. We collect corpora for the events based on the *event range*. That is, for each event E, we consider the time span between the first day of the union of all reference timelines T_E for that event until

[7] The two systems' similarity is further underscored by their retrieval overlap, as discussed in Section 5.2.

[8] For example, for *bpoil*, overlap is 81.4% at $n = 1000$ and 47.58% at $n = 5000$, and for *snowden*, overlap is 69.9% at $n = 1000$ and 19.46% at $n = 5000$.

[9] `https://github.com/smartschat/tilse`.

algorithm	IR	concat		agree		align+ m:1		Date Selection
		R1	R2	R1	R2	R1	R2	F1
Chieu	simple	0.264	0.044	*0.026*‡	*0.006*‡	*0.044*‡	0.008‡	0.168
	bm25	0.258	0.045	0.020	0.004	0.039	0.007	0.168
	bm25$_{\text{boot}}$	*0.279*†,‡	*0.049*†,‡	0.023	*0.006*‡	0.042	*0.009*	*0.170*
regression	simple	0.279	0.043	0.030	0.005	0.041	0.007	0.271
	bm25	0.286	0.043	0.031	0.005	0.043	0.007	*0.295*†
	bm25$_{\text{boot}}$	***0.320***†,‡	*0.051*†,‡	*0.035*†	*0.007*	*0.049*†,‡	*0.008*	0.290
tilse	simple	0.284	0.050	0.036	0.008	0.048	0.009	0.273
	bm25	0.291	0.053†	0.037	0.008	0.049	0.010	0.288
	bm25$_{\text{boot}}$	*0.309*†,‡	***0.058***†,‡	***0.044***†,‡	***0.010***†,‡	***0.056***†,‡	***0.012***†,‡	***0.305***†

Table 5: Macro-averaged results for Setup I (with sentence filter). Highest value per metric is given in bold, highest value per algorithm in italics. † indicates statistical significance wrt to *simple* and ‡ wrt to *bm25* (all $p = 0.05$).

the last day of the union of timelines. This serves as an approximation of the actual time range of the event. Within this range, we extract the top N documents for both *bm25* and *bm25$_{boot}$*, where N is the number of articles retrieved by *simple*.[10]

In order to make the resulting C_Es compatible with the package *tilse*, we furthermore temporally tag each document with *heideltime* (Strötgen and Gertz, 2013).

7.2. Evaluation

Automatic evaluation of TLS is mostly done with the same evaluation metrics as standard summarization, namely ROUGE (Lin, 2004). However, Martschat and Markert (2017) presented TLS-specific variants of ROUGE: *concat*, *agreement* and *align+ m:1*. These metrics perform evaluation by concatenating all daily summaries, evaluating only matching days, and evaluating aligned dates based on date and content similarity, respectively. We report ROUGE-1 and ROUGE-2 F$_1$ scores for the *concat*, *agreement* and *align+ m:1* metrics. Since TLS has an emphasis on date selection – which is not present in standard summarization –, *date selection* is another important metric for TLS evaluation. We evaluate date selection using F$_1$ score. Following Martschat and Markert (2018), we evaluate against each reference timeline individually but report results macro-averaged over events. For evaluation, we use the scripts provided by the *tilse* package.

7.3. IR Impact on TLS: Results and Discussion

Table 5 shows the results of Setup I (with sentence filter). The results reported here are not directly comparable to results in previous work as we specifically constructed different corpora. Additionally, evaluation in our experiments was done on a higher number and a different set of events. While the results are therefore worse – in absolute terms – than Martschat and Markert (2018), we still validate the superior performance of *tilse* compared to both *Chieu* and *regression*. We note that performance increases for all TLS algorithms when using gradually stronger IR methods. For each IR method, however, the algorithm ordering remains the same: *tilse* > *regression* > *Chieu*.

Overall speaking, *tilse* in combination with *bm25$_{boot}$* as the IR method performs best. For all metrics, it significantly outperforms *simple* and, with the exception of date selection, it is also significantly better than the standard *bm25* IR method.[11] The only metric where *tilse-bm25$_{boot}$* is outperformed is concat-R1 (here, *regression-bm25$_{boot}$* performs best). However, since date selection performance of *tilse* is unmatched, the more informative measures of agreement and alignment ROUGE are also better with *tilse* than with the other algorithms by a large margin.

Most importantly, we see that a weaker algorithm combined with a stronger IR method is competitive to a stronger algorithm with a weaker IR method. For example, *Chieu* with *bm25$_{boot}$* IR performs close to *tilse* with *simple* IR; while *regression* with *bm25$_{boot}$* even outperforms *tilse-simple* on most metrics.

This provides strong evidence for considering IR a vital component of the task itself and therefore for any TLS algorithm. This influence of the IR method on TLS performance has not been considered sufficiently by previous research in this field. Previous work tended to optimize on existing corpora that relied on sophisticated, but ultimately irreproducible IR. Therefore, they might not adapt well to real-world scenarios where the applicability of such IR cannot be guaranteed.

7.4. Filtering Impact on TLS: Results and Discussion

Table 6 shows the results for Setup II, where we did not employ the sentence filter. As mentioned above, sentence filtering only considers sentences for the timeline that contain one of the original keywords. As we can see from Table 6, using *bm25$_{boot}$* again yields performance gains for all algorithms.

In direct comparison to Setup I, however, dropping the sentence filter results in an overall worse performance. Both *Chieu* as well as *tilse* demonstrate a strong dependence on sentence filtering. This effect is especially prominent for *tilse*, which without sentence filtering does not outperform *Chieu* and *regression* on many metrics. In particular, *tilse* without sentence filtering is now worse than weaker algorithms with sentence filtering. Of the three algorithms, *regression* is most stable with regard to the use of the sentence

[10]We keep the corpus size the same for all three IR methods as we leave the investigation of corpus size impact and the determination of ideal corpus sizes to future work.

[11]We perform the paired t-test with $p = 0.05$.

algorithm	**IR**	**concat** R1	**concat** R2	**agree** R1	**agree** R2	**align+ m:1** R1	**align+ m:1** R2	**Date Sel.** F1
Chieu	simple	0.207 ↘	0.029 ↘	0.016 ↘	0.003 ↘	0.027 ↘	0.005 ↘	0.179 ↗
	bm25	0.227† ↘	0.034† ↘	0.016 ↘	0.003 ↘	0.031 ↘	0.006 ↘	*0.185* ↗
	bm25$_{\text{boot}}$	*0.247*† ↘	*0.041*† ↘	*0.017* ↘	*0.004* ↘	*0.034* ↘	*0.007* ↘	0.177 ↗
regression	simple	0.272 ↘	0.039 ↘	0.026 ↘	0.004 ↘	0.035 ↘	0.005 ↘	0.289 ↗
	bm25	0.273 ↘	0.043† —	*0.031*† —	*0.005*† —	0.039† ↘	0.006† ↘	***0.320***†,* ↗
	bm25$_{\text{boot}}$	***0.295***†,‡ ↘	***0.045***† ↘	*0.031*† ↘	*0.005* ↘	***0.042***† ↘	*0.007*† ↘	0.292 ↗
tilse	simple	0.206 ↘	0.031 ↘	0.026 ↘	0.004 ↘	0.034 ↘	0.005 ↘	0.278 ↗
	bm25	0.224† ↘	0.041† ↘	***0.034***† ↘	***0.008***† —	***0.042***† ↘	***0.009***† ↘	0.319 ↗
	bm25$_{\text{boot}}$	*0.239*† ↘	*0.043*† ↘	0.031† ↘	0.006† ↘	0.041† ↘	0.007† ↘	*0.301*† ↘

Table 6: Macro-averaged results for Setup II (**without** sentence filter). Highest value per metric is given in bold, highest value per algorithm in italics. † indicates statistical significance wrt to *simple*, ‡ wrt to *bm25*, and * wrt to *bm25$_{boot}$* (all $p = 0.05$). ↘ denotes a performance decrease, ↗ increase, and — same performance in relation to Setup I in Table 5.

filter. It is noteworthy that the only metric that actually benefits from the lack of a sentence filter is date selection, indicating that the usual filtering of sentences causes a loss in informative sentences regarding timeline dates.

Sentence filtering boosts performance: By limiting the set of sentences considered for timeline generation to only sentences containing an event-specific keyword, the chance of choosing a highly relevant sentence rises. However, this dependence on a very simple, keyword-based sentence filter has to be seen critically. As the algorithms only generate summaries from a filtered set of sentences, the actual influence of the algorithms becomes unclear. The algorithms have trouble generating timelines from the full set of sentences in the corpus although partially using complex sentence ranking functions. In addition, resulting summaries with sentence filter will read redundant. For example, a timeline for the Syrian war will consist of sentences where each and every one will either contain the keywords *Syria* or *war*, a factor not present in human-written timelines.

8. Conclusion

Most current TLS evaluation is carried out on two corpora collected with sophisticated, commercial search engines. This, however, makes corpus collection itself irreproducible and TLS system performance harder to compare as the influence of IR is not accounted for. We therefore advocate for integrating IR into the development of TLS systems by having a common background corpus to generate event-specific corpora for evaluation of TLS systems. This becomes especially important with regard to real-world scenarios where corpora are likely to be unfiltered and static.

We thus conducted the first investigation into the impact of IR on TLS performance. We used the *newsroom* corpus to extract event-specific corpora for 15 events with three different IR methods. We then compared the performance of three well-established TLS algorithms using these corpora. All algorithms improve their performance in combination with a stronger IR method. Importantly, however, a weaker TLS algorithm can match a stronger one in performance by employing a stronger IR method.

In a second experiment, we also showed that the results of TLS systems are highly sensitive to keyword-based sentence filtering, often resulting in considerably worse results when such overly simplistic sentence filtering is not used. This suggests that some improvements in TLS reported in the literature is actually not due to algorithm intricacies but to pre-processing and filtering decisions which are not the focus of the papers.

In the future, this line of work can be extended in multiple directions. One is to investigate even more sophisticated, yet reproducible IR methods. This could diminish the gap between different TLS systems even further. Another is to construct TLS algorithms with the impact of IR in mind as well as eliminating the sentence filter. This might produce more stable systems across different real-world scenarios. Consequently, this could result in TLS systems jointly performing IR and summary generation.

Appendix: Annotation Guidelines

We now detail the guidelines for the manual annotation of document relevance for an event, as conducted in Section 5.1.

TREC Annotation Guidelines

As preliminary guidelines, we take the TREC four-point scale, as outlined in Section 2.2 of Sormunen (2002), to assess the relevance of documents:

> (0) The document does not contain any information about the topic.
> (1) The document only points to the topic. It does not contain more or other information than the topic description. Typical extent: one sentence or fact.
> (2) The document contains more information than the topic description but the presentation is not exhaustive. In case of a multi-faceted topic, only some of the sub-themes or viewpoints are covered. Typical extent: one text paragraph, 2-3 sentences or facts.
> (3) The document discusses the themes of the topic exhaustively. In case of a multi-faceted topic, all or most sub-themes or viewpoints are covered. Typical extent: several text paragraphs, at least 4 sentences or facts.

Additional annotation guidelines

Relevance of documents is based on the TREC guidelines outlined above. Articles that fit into either category 2 or 3 are considered as related to the topic. Additionally, we consider the following specific guidelines due to the idiosyncrasies of *newsroom* and the document requirements of timeline summarization:

- Opinion pieces are excluded.

  To recognize opinion pieces we use the following heuristics:

  - Use of first person pronouns
  - Anecdotal information
  - Personal experience of the event

- Only news articles that cover essential information for the event or a sub-event are considered relevant.

- Articles on "spin-off" events are considered not relevant. "spin-off" events are events which are connected to the original event, but not part of the main event, e.g., a lawsuit resulting from a hospital not following strict quarantine measures during the Ebola outbreak.

- Articles providing general information on an entity involved in an event, but no substantial information on the event itself, are considered as not relevant. E.g. an article on the biography of Saddam Hussein or an article on the symptoms of the Ebola infection.

- Articles containing errors are considered not relevant. Some articles in *newsroom* were not extracted correctly. Most notably, these are articles containing comments or repeated sentences.

9. Bibliographical References

Aslam, J., Diaz, F., Ekstrand-Abueg, M., Pavlu, V., and Sakai, T. (2013). TREC 2013 Temporal Summarization. In *Proceedings of the 22nd Text Retrieval Conference*, pages 1–14.

Aslam, J., Diaz, F., Ekstrand-Abueg, M., McCreadie, R., Pavlu, V., and Sakai, T. (2014). TREC 2014 Temporal Summarization Track Overview. In *Proceedings of the 23rd Text Retrieval Conference*, pages 1–15.

Aslam, J., Diaz, F., Ekstrand-Abueg, M., McCreadie, R., Pavlu, V., and Sakai, T. (2015). TREC 2015 Temporal Summarization Track Overview. In *Proceedings of the 24th Text Retrieval Conference*, pages 1–16.

Barros, C., Lloret, E., Saquete, E., and Navarro-Colorado, B. (2019). NATSUM: Narrative abstractive summarization through cross-document timeline generation. *Information Processing and Management*, 56(5):1775–1793.

Chieu, H. L. and Lee, Y. K. (2004). Query based event extraction along a timeline. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 425–432.

Feldhus, N. (2016). *An in-depth investigation on timeline summarization evaluation*. Bachelor's thesis, Heidelberg University.

Grusky, M., Naaman, M., and Artzi, Y. (2018). Newsroom: A Dataset of 1.3 Million Summaries with Diverse Extractive Strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 708–719.

Kedzie, C., McKeown, K., and Diaz, F. (2015). Predicting salient updates for disaster summarization. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1608–1617.

Kedzie, C., Diaz, F., and McKeown, K. (2016). Real-time web scale event summarization using sequential decision making. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*.

Kessler, R., Tannier, X., Ege, C., Moriceau, V., and Bittar, A. (2012). Finding Salient Dates for Building Thematic Timelines. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 730–739.

Kilgarriff, A. (2007). Googleology is Bad Science. *Computational Linguistics*, 33(1):147–151.

Liang, D., Wang, G., and Nie, J. (2019). A Dynamic Evolutionary Framework for Timeline Generation based on Distributed Representations.

Lin, H. and Bilmes, J. (2011). A Class of Submodular Functions for Document Summarization. In *Proceedings of the 49th Annual Meeting ofthe Association for Computational Linguistics*, pages 510–520.

Lin, C.-Y. (2004). ROUGE: A Package for Automatic Evaluation of Summaries. In *Proceedings of the Text Summarization Branches Out Workshop at ACL '04*, pages 74–81.

Martschat, S. and Markert, K. (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.

Martschat, S. and Markert, K. (2018). A Temporally Sensitive Submodularity Framework for Timeline Summarization. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 230 – 240.

Mendes, A., Narayan, S., Miranda, S., Marinho, Z., Martins, A. F. T., and Cohen, S. B. (2019). Jointly extracting and compressing documents with summary state representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3955 – 3966.

Mihalcea, R. and Tarau, P. (2004). TextRank: Bringing order into texts. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411.

Nguyen, K.-H., Tannier, X., and Moriceau, V. (2014). Ranking Multidocument Event Descriptions for Building Thematic Timelines. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 1208–1217.

Robertson, S. and Zaragoza, H. (2009). The Probabilistic Relevance Framework: BM25 and Beyond. *Foundation and Trends in Information Retrieval*, 3(4):333–389.

Shi, T., Wang, P., and Reddy, C. K. (2019). LeafNATS: An

Open-Source Toolkit and Live Demo System for Neural Abstractive Text Summarization. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Demonstrations)*, pages 66 – 71.

Sormunen, E. (2002). Liberal Relevance Criteria of TREC - Counting on Negligible Documents? In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 324–330.

Strötgen, J. and Gertz, M. (2013). Multilingual and cross-domain temporal tagging. *Language Resources and Evaluation*, 47(2):269–298.

Suzuki, S. and Kobayashi, I. (2014). On-line Summarization of Time-series Documents using a Graph-based Algorithm. In *Proceedings of the 28th Pacific Asia Conference on Language, Information and Computation*, pages 470–478.

Tran, G. B., Alrifai, M., and Quoc Nguyen, D. (2013a). Predicting relevant news events for timeline summaries. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 91–92.

Tran, G. B., Tran, T. A., Tran, N.-K., Alrifai, M., and Kanhabua, N. (2013b). Leveraging Learning To Rank in an Optimization Framework for Timeline Summarization. In *Proceedings of SIGIR 2013 Workshop on Time-aware Information Access*.

Tran, G., Alrifai, M., and Herder, E. (2015a). Timeline Summarization from Relevant Headlines. In *Proceedings of the 37th European Conference on Information Retrieval*, pages 245–256.

Tran, G., Herder, E., and Markert, K. (2015b). Joint Graphical Models for Date Selection in Timeline Summarization. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 1598–1607.

Wang, L., Cardie, C., and Marchetti, G. (2015). Socially-Informed Timeline Generation for Complex Events. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1055–1065.

Wang, C., Zhang, R., He, X., and Zhou, A. (2016a). NERank: Ranking Named Entities in Document Collections. In *Proceedings of the 25th International Conference Companion on World Wide Web*, pages 123–124.

Wang, W. Y., Mehdad, Y., Radev, D. R., and Stent, A. (2016b). A Low-Rank Approximation Approach to Learning Joint Embeddings of News Stories and Images for Timeline Summarization. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 58–68.

Yan, R., Kong, L., Huang, C., Wan, X., Li, X., and Zhang, Y. (2011). Timeline generation through evolutionary trans-temporal summarization. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 433–443.