Ranking Models for the Temporal Dimension of Text

Ranking Models for the Temporal Dimension of Text

STEFANO GIOVANNI RIZZO, Amazon, Luxembourg[†]
MATTEO BRUCATO, Microsoft Research[‡], USA
DANILO MONTESI, University of Bologna, Italy

Temporal features of text have been shown to improve clustering and organization of documents, text classification, visualization, and ranking. Temporal ranking models consider the temporal expressions found in text (e.g., "in 2021" or "last year") as time units, rather than as keywords, to define a temporal relevance and improve ranking. This paper introduces a new class of ranking models called Temporal Metric Space Models (TMSM), based on a new domain for representing temporal information found in documents and queries, where each temporal expression is represented as a *time interval*. Furthermore, we introduce a new frequency-based baseline called Temporal BM25 (TBM25). We evaluate the effectiveness of each proposed metric against a purely textual baseline, as well as several variations of the metrics themselves, where we change the aggregate function, the time granularity and the combination weight. Our extensive experiments on five test collections show statistically significant improvements of TMSM and TBM25 over state-of-the-art temporal ranking models. Combining the temporal similarity scores with the text similarity scores always improves the results, when the combination weight is between 2% and 6% for the temporal scores. This is true also for test collections where only 5% of queries contain explicit temporal expressions.

ACM Reference Format:
Stefano Giovanni Rizzo, Matteo Brucato, and Danilo Montesi. 2022. Ranking Models for the Temporal Dimension of Text. 1, 1 (September 2022), 34 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Text is very rich with various kinds of lexical items, ranging from part-of-speech entities, to spatial and temporal references. As, on one hand, research in Natural Language Processing (NLP) advances, becoming more able to automatically recognize and accurately interpret these elements in text, on the other hand, Information Retrieval (IR) systems can greatly benefit from the richer understanding and structuring of text[1] provided by new NLP tools. For example, a traditional vector space model can be augmented to work differently with terms that refer to specific part-of-speech entities, such as nouns, verbs, subjects, and objects, or with terms referring to locations, persons or *dates* [18].

An important class of lexical items that has received a lot of attention by the IR community in recent years is the class of *temporal expressions*, often referred to as *timexes* in the NLP literature [41]. A temporal expression is a part of text (often a sequence of words) that collectively identifies one or more *time periods*. For example, the expressions

[1]In this work, we refer to *text* as the content of either queries or documents.

[†]Work carried out prior to joining Amazon.
[‡]Work carried out prior to joining Microsoft.
Authors' addresses: Stefano Giovanni Rizzo, srizzo@amazon.lu, Amazon, Luxembourg[†], Luxembourg; Matteo Brucato, mbrucato@microsoft.com, Microsoft Research[‡], Redmond, WA, USA; Danilo Montesi, danilo.montesi@unibo.it, University of Bologna, Bologna, Italy.

"`last year`[2]" and "`in 2020`" are temporal expressions, both referring (considering the time of writing this article) to the year 2020. Similarly, "`every weekend in the past two years`" is also a temporal expression, as it refers to several weekends during a specific time frame. The definition of a temporal expression can also include more vague utterances, such as "`every so often`" or "`not long ago`", or the so-called *named temporal expressions* [8], such as "`Independence Day`", which refer to time through the use of common names.

Temporal expressions whose meaning can be correctly interpreted, with or without a degree of ambiguity or uncertainty, and either automatically or manually, can augment the capabilities of a search engine in discovering relevant documents for queries that display some temporal *sensitiveness*. A relevant document for temporally sensitive queries not only has to satisfy the "content" requirement of the query (i.e., what the document is about), but also its "time" requirement (i.e., what period of time the document is about). As an example, consider the query ***obama election results***. Clearly, a relevant document should be about the results of the presidential campaigns in which Obama participated. However, it is not clear whether the query is requiring documents about the results of the 2008 or 2012 election. If the user intent is to search for the 2008 election results, the engine can ignore documents that are only about the 2012 results, consequently improving its effectiveness. Understanding whether the user intent is to obtain the 2008 or 2012 results also requires a notion of temporal "context" in which the query is issued. The notion of context also extends to documents, if we analyze the period of time when the document was written. The context plays a crucial role especially in the correct interpretation of the *relative* temporal expressions found in queries and documents (e.g., "`yesterday`" or "`this morning`"). Current state-of-the-art NLP tools are able to exploit the context to provide a correct interpretation of the temporal content, which our models utilize to improve the effectiveness of the final ranking.

Despite the important role of timexes in Information Retrieval, traditional IR systems continue to treat these expressions as normal text terms instead of time intervals. Using timexes to improve search engines poses several challenging opportunities, many of which are still unsolved [11, 15]. In this work, we address *three challenges*. The first one pertains the identification and proper interpretation of the temporal expressions from the text. Some temporal expressions follow regular patters, such as formatted dates (e.g. "`November 4th, 2008`"), and are easier to recognize and interpret than others. Some other expressions are easy to find but harder to interpret, such as *relative* expressions (e.g., "`today`", "`last year`"), as they require a context time to be resolved (yesterday with respect to which day?). Other expressions are hard even to identify, as they require specific domain knowledge, such as holiday names [34]. The task of recognizing and interpreting temporal expressions has been a focus of NLP research, with noticeable success over the last decade [53]. For instance, HeidelTime [51] scored 85% accuracy in identifying timexes and normalizing to their exact value in the annotation challenge TempEval 2 [55], while in TempEval 3 best precision obtained was 99.05% and best recall 91.30% [53]. Our work does not further address these NLP challenges, but rather builds on state-of-the-art NLP tools that perform successful identification and interpretation of temporal expressions. Instead, we present a overview of the process needed to take advantage of the interpreted temporal expressions to improve the retrieval.

The second challenge is that most queries do not expose their temporal sensitiveness explicitly. In particular, it is hard to decide *when* a query is temporally sensitive, to *what degree*, and what exactly its *temporal intent* is [13, 23, 27]. This is especially hard when temporal queries do not explicitly contain temporal expressions, but their temporal sensitiveness is *implicit*. For example, the query ***fifa world cup brazil*** does not contain explicit dates. However, supposing the query is issued around 2014, it is easy to guess that it is most likely referring to the FIFA World Cup held in Brazil in 2014, rather than the cup that took place in 1950. In prior work, it has been argued that most queries

[2]Throughout the paper, we will use this font and format to indicate a timex.

are not temporal (i.e., considering time information cannot improve the ranking) just because queries contain explicit time expressions very rarely (about 1.5%, according to [37]). However, as the previous example shows, the absence of temporal expressions in a query does not always imply null temporal sensitiveness. Another argument made in prior work is that queries without time expressions, if they are temporal, then they simply focus on current events (i.e., they implicitly ask for the most recent information on a topic). In this paper, we provide evidence that *most* queries are indeed temporal and not necessarily focused on current information. We show that even in test collections with no explicitly constructed temporal queries (i.e., the TREC collections), the temporal components of the documents (i.e., their temporal expressions) play a fundamental role in their relevance. In fact, we show that by carefully identifying temporal features for a supposedly non-temporal query, a temporal model can improve the ranking over a purely textual model.

Finally, even if we knew exactly the temporal sensitiveness of the query, and how to properly recognise and interpret the temporal expressions in documents and queries, search engines still face another major challenge: how to use this information to correctly rank the documents not only according to their content, but also to the temporality of their content with respect to the temporal sensitiveness of the query. Despite the widespread presence of time in documents and queries, this third challenge has not been sufficiently addressed by the research community. Temporal Information Retrieval (TIR) has emerged in recent years with the goal of improving the effectiveness of retrieval systems along the temporal dimension [3]. Early work in this area has studied how to exploit the creation date of documents and the issue time of queries [33]. Only more recent work has addressed the extraction and interpretation of temporal expressions [4, 9, 30]. In this paper, we build on and extend these results with more sophisticated ranking models that exploit the temporal information found in text. Our models use *distances* (in metric, hemi-metric, or quasi-metric spaces) between *time intervals* (dates and periods of time) found in text to capture their containment, overlapping, and matching properties.

In this paper, we make the following contributions:

- We define a formal *temporal domain* for representing temporal information in both documents and queries. The core of our approach consists on: (1) Using existing NLP tools to extract and interpret temporal expressions; (2) Mapping the interpreted expressions into *temporal intervals*, which constitute the temporal representation of documents and queries in the ranking model.
- As a first, simple step, we show how this information can be used to enhance the effectiveness of existing textual ranking models, such as BM25 [43].
- We introduce new, more sophisticated ranking models that exploit the peculiarities of the temporal representations of documents and queries. These models are based on standard and generalized distances (hemi-metric and quasi-metric) between temporal intervals.
- We discuss how to derive effective ranking models by combining temporal and textual scores, and how score normalization can impact the final ranking.
- We conduct extensive experiments on different collections that show that: (1) The temporal information found in text is plenty and informative enough for substantially and consistently improving the effectiveness of both traditional time-agnostic retrieval models and our novel temporal models, for a typical ad hoc search task; (2) Treating temporal expressions differently from textual terms largely improves the effectiveness of traditional textual ranking models (BM25); (3) Traditional ranking models are, however, insufficient for modeling the relevance in the temporal domain, and our metric space models prove more effective in all cases; (4) The choice

of the granularity for the discretization of time and the specific distance function used to rank documents have different effects depending on the collection. While there is no single choice that is better for all collections, our models are robust to imperfect settings since the results are always at least as good as the baseline and better in most cases.

This paper focuses on defining and evaluating the effectiveness of our newly introduced temporal ranking models. While there are performance aspects such as how to efficiently access the temporal similarity scores, these systems aspects are orthogonal to the scope of this work and part of future work (Section 9). This work extends the conference paper that presented the first metric model for TIR [9], in several ways: (1) it introduces a new temporal model, TBM25, based on the Okapi BM25 scoring function, which considers the frequency of the intervals in the document and in the corpus; (2) it introduces five new temporal distances to consider both temporal coverage and distance between the intervals ends; (3) it addresses the problem of normalizing the temporal scores prior to combination with textual scores; (4) it introduces a novel method to extract time intent from queries that do not contain explicit temporal expressions; (5) it provides a qualitative analysis and intuitive explanation of all the temporal distances being introduced; (6) it extends the experimental evaluation to a total of five different collections for a total of 7 million documents, with the temporal annotation of 42 million timexes; (7) it adds a comparison of our metric models with the state-of-the-art model, LMTU [4], across all the five collections; (8) it extends the evaluation to variations of the aggregation functions, granularities for the representation of the time intervals (day, month, year).

The remainder of this paper is organized as follows. In Section 2, we give a summary of the related work and outline the advantages of our novel approach. In Section 3, we describe how time is found, implicitly and explicitly, in document and queries, and how it can be tagged and interpreted to form what we call the "temporal scope" of a document or a query. In Section 4, we present two temporal similarities: frequency-based a metric-based temporal similarities. In Section 5, we define several versions of the metric-based temporal similarity, each capturing different temporal relationships between a query and a document. In Section 6, we describe and evaluate different methods for normalizing and combining temporal and non-temporal similarity scores. In Section 7, we provide a qualitative comparison of proposed and state-of-the-art temporal similarities. In Section 8, we evaluate the improvements in the effectiveness of the proposed models in relation to state-of-the-art models. In Section 9, we conclude by outlining our contributions and future work.

2 RELATED WORK

Work in Temporal Information Retrieval addresses a variety of tasks. Early work focuses on indexing and retrieval of the "freshest" versions of web documents that change over time [45]. The concept of *freshness* is the focus of several ranking tasks proposed in the literature [24, 33, 39]. Close to this area are *popularity measures* of web pages, such as PageRank [38], specialized to take into account the creation time of each "backlink" (i.e., the link pointing to a page): work in this area proposes to penalize pages with stale backlinks (i.e., links coming from old pages), while favoring recently linked pages [5, 61].

The notion of *temporal relevance* is the subject of substantial work. Perkiö et al. postulates that ranking should promote documents whose topics are most active at the time of the query [40]. Authors typically agree on the fact that temporal similarity should be combined with traditional text similarity for better performance. In [25], time is extracted from the content of the web documents, but also from the document creation time and the time of page crawling, and ranking is obtained with the linear combination of keywords similarity, temporal similarity, and PageRank. In [29], the

implicit time of the query is determined from the query's keywords and the textual similarity is combined with the similarity between the time of the query and the creation time of documents. In [1], terms relevance is boosted based on its frequency on the revision history of documents. Berberich et al. define a combination of temporal similarity with text similarity, where the temporal similarity is based on a language model with the requirements of specificity, coverage and maximality, in order to capture the probability of generating the temporal expressions in the query from those of a document [4]. This work is the closest to ours for its intent and applicability, and therefore we fully compare it with our proposed methods in later sections of the paper (Section 7 and Section 8). A later work [30] proposes a time-aware ranking approach based on learning-to-rank techniques for temporal queries and tested using the same collection of [4]. A similar approach is presented by Khodaei et al. [31], in which the temporal similarity between two *timespan* depends on how many individual time components they share, and the temporal and non-temporal similarities are combined through a weighted sum. While learning-to-rank solutions can provide further improvements in the effectiveness of TIR models, they require (1) labeled data and (2) several temporal similarity models to form the learning features (e.g., the temporal language model in [4] is one of the features used in [30]). Our similarity models presented in this paper can also be used in conjunction with any learning-to-rank method. However, given that we do not use any labeled data in our evaluation, exploring learning-to-rank solution is orthogonal to the scope of our work.

The model proposed in our work follows more recent approaches to TIR, which rely on the use of automatic taggers, such as TARSQI [54], to extract temporal expressions from the document's text. The intuition behind this approach was originally introduced by Alonso et al. [2, 3]. As Alonso et al. stated: "The central idea in temporal information retrieval is to utilize the temporal expressions that have been determined for each document in a given document collection in order to rank search results." Berberich et al. [4] extract time expressions from queries and documents using the TARSQI temporal tagger, before applying a temporal language model to the interpreted intervals. Related work by Campos et al. [10, 12], instead of extracting temporal expressions of all document content, extracts those from the web snippet, using an ad hoc rule-based tool for explicit year temporal expressions.

Because explicit temporal expressions in queries are generally very rare, an orthogonal problem that encompasses all aspects of TIR is identifying the *temporal intent* of a query. Jones and Diaz [27], define the temporal profile of a query as the difference between the temporal distribution of top-k documents and the collection. Another approach for temporal query intent classification is to use queries logs to extract the time of the keywords popularity peaks [32]. More recent research applies machine learning approaches to temporal query intent classification [19, 23], using different kinds of features, such as part of speech tags, pseudo-relevant documents and ontologies, to train a classifier. Campos et al. [13] use word-year co-occurence in a corpus to identify the set of dates most relevant to a query.

3 MODELING THE TEMPORAL DIMENSION OF TEXT

We identify two main classes of temporal features that can be associated with a document or query: *meta* temporal features, and *content* temporal features. Meta temporal features are dates or times associated with a document or query that are not necessarily found in their text. The *document creation time* (DCT) [36] and the *query issue time* (QIT) [16] are examples of meta temporal features. Other examples are the *revision time* [1] and *deletion time* of a document, and *reissue time* of a query [52]. These features provide the temporal *context* of documents and queries.

Differently from the meta features, the content temporal features consist on dates or times that are contained within, extracted, or inferred from, the text of documents and queries. Temporal expressions fall within this category. While the meta features inform about when a document was created, modified, deleted, and so forth, the content features carry information about the content and meaning of the document: what periods of time does the document refer to?

Similarly, while the meta features can identify when a query was issued, the content features tell about what periods of time the user who issued the query is mostly interested in finding information about.

The meta temporal information has been extensively studied to improve the ranking in time-sensitive queries [14, 27, 29]. The focus of this paper is on the content features. In particular, we show how temporal expressions can be used to successfully model the temporal content of documents and queries to improve the retrieval effectiveness on traditional ad hoc queries.

Exploiting content temporal information poses *three* key challenges:

(1) First, it is hard to *identify* and correctly *interpret* the content temporal information in text. While a domain expert is able to understand what exact periods of times a document is about by simply reading the text, a computer finds this task extremely complicated. This is particularly true for relative expressions for which a reference time is needed, for example "`last June`" or "`earlier this year`" [51].
(2) Second, the content temporal information found in text need to be correctly *represented* as precise mathematical objects. The representation is crucial for the success of the ranking process. To understand this, consider two simple options for representing time in documents and queries. One consists of representing a temporal expression, such as "`December 2020`", as a textual *token* (e.g., `2020-12`). Another option is to represent the same expression as an *interval*, with a beginning and an end (e.g., `2020-12-01..2020-12-31`). The first representation can easily be used to find all documents referring to the same month, using fast methods such as inverted indexes. However, it may fail to identify documents referring to dates that are close to December 2020, or to give higher scores to documents that have references to specific days inside of that month. Representing a time expression as an interval would offer more flexibility in devising better scoring functions for such cases.
(3) Third, even with a complete and accurate semantic understanding of the temporal information, we still need a *ranking model* that can effectively exploit the temporal aspects of queries and documents. For example, suppose a query is asking for information about "`yesterday`", and there are two documents, one about "`two days ago`" and one about "`last week`". Deciding which document is preferable for this query is nontrivial and requires a precise notion of distance between temporal intervals.

We address the first challenge by focusing our attention on temporal expressions. This is a somewhat simplifying approach, as the temporal information of documents and queries does not necessarily need to exist in the form of dates and time references, but could potentially be inferred by a deeper semantic understanding of the text. However, this approach is motivated by the fact that there exist a number of NLP tools that are able to identify and interpret temporal expressions with very high precision [53]. A search engine can thus easily and readily incorporate these tools to augment its capabilities and improve its ranking models. Nonetheless, the solutions that we provide for the second and third challenges are independent of whether we use timexes or extract temporal information in more sophisticated ways.

The second and third challenges are the main focus of this work. In this section, we describe how to represent the temporal expressions found in documents and queries (Section 3.1). Using this representation, we define two temporal similarities in Section 4: a temporal similarity modeled after the BM25 [43] *textual* similarity, and a purely-temporal similarity model based on distances between temporal intervals. The latter requires the definition of metric spaces in order to capture different time relations, which we introduce in Section 5. Finally, in Section 6 we discuss how a purely-temporal ranking model should be combined with a purely-textual (or term-based) model to improve the effectiveness of either of the models.

3.1 Modeling the temporal information in text

We now introduce the definitions and notation used in this paper to model temporal information in text. In Section 3.2, we discuss how this information can be automatically extracted from the text using modern NLP tools.

Given a a set of *time instants* $\mathbb{T} \subseteq \mathbb{R}$, the smallest piece of information that we attribute to an excerpt of text, regardless of whether it is a document or a query text, is a *temporal* or *time interval*:

Definition 3.1 (Temporal Interval). A temporal interval $[t_s, t_e]$ is a *closed interval* identified by an *ordered pair* of time instants $t_s, t_e \in \mathbb{T}$, $t_s \leq t_e$.

The first component of the pair, t_s, indicates the *starting time* of the time interval, while the second component, t_e, indicates its *end time*. The meaning of a temporal interval $[t_s, t_e]$ is, therefore, "the period of time starting from time t_s, until, and including, time t_e". The time instances, t_s and t_e, have an associated meaning as well. For instance, t_s can be interpreted as "January 1st, 2017", and t_e as "December 31st, 2017", in which case the interval would indicate the whole year 2017.

For practical reasons, it is common to discretize $\mathbb{T}$, i.e., $\mathbb{T} \subseteq \mathbb{Z}$, and fix a minimum time, $t_{\min} \in \mathbb{Z}$, and a maximum time, $t_{\max} \in \mathbb{Z}$, for the minimum and maximum time instants that the system can handle, such that $\mathbb{T} = [t_{\min}..t_{\max}]$.

Definition 3.2 (Temporal Domain). Given a discretized $\mathbb{T} = [t_{\min}..t_{\max}]$, the *temporal domain* is the finite set of all possible time intervals: $\Delta_{\mathbb{T}} = \{[t_s, t_e] \mid t_s, t_e \in \mathbb{T}, t_s \leq t_e\}$.

A discretized temporal domain is then grounded to reality by setting a *time granularity*. This is commonly referred to as the "chronon", and it represents the smallest discrete unit of time that the system can refer to. Typical examples of chronons are: a day, a week, a month, or a year. Once a chronon is set, all time instances adhere to it (i.e., they all refer to days, or they all refer to months, etc.).

Table 1. Time intervals of temporal domain with year chronon, and $\mathbb{T} = [0, 2]$, where time instant 0 indicates year 2014.

t_s	t_e	Corresponding time period	Time period length
0	0	2014	1 year
0	1	2014–2015	2 years
0	2	2014–2016	3 years
1	1	2015	1 year
1	2	2015–2016	2 years
2	2	2016	1 year

Example 3.3. With a *year* chronon, $t_{min} = 0$ corresponding to 2014, $t_{max} = 2$, Table 1 depicts the full discretized temporal domain $\Delta_{\mathbb{T}}$, i.e., all the time intervals that the system can internally represent. Notice that, in this example, the longest time interval is $[0, 2]$, corresponding to the three-year-long time period 2014–2016, and there are three shortest time intervals, $[0, 0]$, $[1, 1]$, and $[2, 2]$, corresponding to the years 2014, 2015, and 2016, respectively.

Making a parallel with traditional term-based retrieval, $\Delta_{\mathbb{T}}$ is analogous to a *temporal dictionary*, that is, the set of all possible temporal tokens that we are interested in capturing from the text. Like in term-based retrieval, the entire text of a document or a query can have several time intervals associated with it. We now define the *temporal scope* of a document (or query), i.e., its representation in the temporal domain.

Definition 3.4 (Temporal Scope). The set of all time intervals of a document D (or query Q, respectively) is called its *temporal scope*, and it is denoted by T_D (or T_Q, respectively).

3.2 Automatic construction of temporal scopes

Our model does not make any assumption regarding how T_D and T_Q are constructed. In principle, temporal intervals could be either manually or automatically identified, without changes to the underlining model. In this paper, we consider their automatic identification through the use of existing, off-the-shelf NLP tools [51, 53] specifically tailored for temporal information. Using NLP is common even in traditional "atemporal" bag-of-words systems, where terms are expanded with synonyms using ontologies [20]. We add temporal support to the ranking model by extending the set of NLP tools to those able to recognize temporal information in text. The schematic process of transforming temporal expressions into the proposed interval representation is summarized in Figure 1.

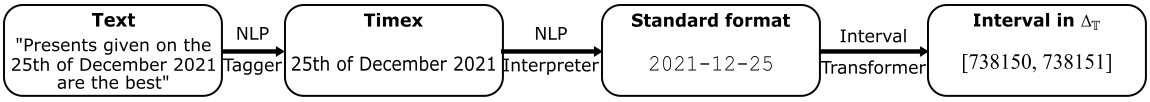


Fig. 1. The process involved in the automatic construction of temporal scopes. First, we use an NLP tagger to identify a timex (e.g., "`25th of December 2021`") from a text. Second, an NLP interpreter converts it to a standard date format (e.g., `2021-12-25`). Finally, we transform the latter into a time interval in the temporal domain $\Delta_{\mathbb{T}}$. In this example, we obtain the interval by using a *day* granularity and counting the number of days starting from t_{min} = `0001-01-01` (the first day of January of Year 1).

3.2.1 Timexes. Temporal information can be found in text in the form of *temporal expressions*, often refer to as *timexes* in the NLP literature. A timex is a (often contiguous) sequence of words found in the raw text of a document or query, to which we can associate a "temporal meaning". For instance, the timex "`two days after Christmas`" means December, the 27^{th}. Notice that this timex does not specify a particular year. Thus, an NLP tool would correctly interpret this timex not as a single temporal interval, but as a set of single-day intervals, one for each year that fall within the temporal domain $\Delta_{\mathbb{T}}$. For example, if $\Delta_{\mathbb{T}}$ spans years 2014, 2015 and 2016, and the chronon is at least as small as a day, then this timex would indicate three single-day intervals: "`2014-12-27`", "`2015-12-27`", and "`2016-12-27`".

Timexes are not restricted to only search terms: They can consist of any part of the text, including stop words, as long as they constitute together an expression with temporal meaning (e.g., "`the 3rd of January`"). In this paper, we consider timexes that can be automatically recognized and interpreted by existing NLP libraries. However, this is not a restriction to the ranking models, as they only utilize the temporal representations, T_D and T_Q, regardless of how they are constructed.

3.2.2 Timex identification and interpretation with NLP tools. Temporal NLP tools for timexes typically work in two steps. First, they *recognize* timexes in the text, and annotate them. The annotation is usually performed by adding XML tags to the original text. An example of this is the TimeML specification language [41], which provides a standardized format for timex annotation. Second, they "normalize" (or "interpret") the annotated timexes. In the rest of the paper, we will refer to timex normalization as "interpretation", to avoid confusion with score normalization (Section 6). Timex interpretation involves the transformation of a previously recognized timex into a standard format for representing dates and times, e.g., ISO 8601 [60], that can be easily parsed by a computer program.

Following the classification found in earlier work [2, 46], explicit timexes appear in two different kinds: *absolute* timexes (e.g., "`12 February 2010`") and *relative* timexes (e.g., "`last week`"). Absolute timexes, once recognized, can be easily interpreted without the need of any contextual knowledge. Conversely, relative timexes need extra knowledge in order to be resolved (e.g., a DCT or QIT), since they relate to some other (implicit or explicit) date. NLP tools are indeed

Table 2. Examples of real timexes recognized (first column), interpreted with NLP tools (second column), and represented as time intervals (third column). Timexes are extracted and interpreted from Robust 2004 TREC collection using HeidelTime. Relative timexes are resolved using the date creation time. Transformation from ISO 8601 to intervals of time instants is applied using 1-day granularity and starting time `0001-01-01T00:00:00`.

Recognized Timex	ISO 8601 Interpretation	Time Intervals with 1-day Chronon
"Today"	1994-03-01	[727989 , 727989]
"next January"	1995-01	[728295 , 728325]
"28 Feb"	1994-2-28	[727988 , 727988]
"the second half of 1993"	1993-H2	[727571 , 727929]
"autumn 1893"	1893-FA	[691306 , 691397]
"last week"	1993-W06	[727602 , 727608]
"Saturday night"	1977-01-01TNI	[721721 , 721721]
"28 July 1914 to 11 November 1918"	1914-07-28, 1918-11-11	[698918 , 700485]

able to successfully recognize both absolute and relative expressions, and to interpret them into a standard format, such as ISO 8601.

We refer to the relevant NLP literature for the details on the interpretation [51, 53].

3.2.3 Transforming timexes to temporal scopes. After the NLP-based timex interpretation, we transform the resulting ISO 8601 strings into temporal intervals (ordered pairs of time instants, cfr. Definition 3.1). We represent a time instant as the number of chronons elapsed from the starting time `00:00:00` of the first of January of the year 1 AD in the Gregorian calendar, or more shortly `0001-01-01T00:00:00` in ISO 8601 format. For example, with a year chronon, this is equivalent to extracting the Gregorian calendar's year (e.g., the discrete time instant of the ISO 8601 timex "`2015-03-20`" is represented with the ordinal 2015). This transformation is *exact* and, thus, it does not lose any information contained in the ISO 8601 strings that the NLP tools produces.

Table 2 shows a few examples of timexes recognized from Robust 2004 [56] using HeidelTime [51] (first column), their ISO 8601 interpretation (second column), and their final transformation as temporal intervals of the temporal domain $\Delta_{\mathbb{T}}$ (third column). In the last column of Table 2, we show examples of the intervals created with a day granularity and starting time `0001-01-01T00:00:00`. It must be noted that the choice of starting time is arbitrary and does not affect the results of the similarity models, as long as the temporal domain fully contains the intervals mentioned in the text.

In the last row of Table 2 we also show a case that expresses a period of time, "`28 July 1914 to 11 November 1918`", that has two timexes and is also converted to a single interval [698918 , 700485], from the starting to the end of the expressed period.

4 TEMPORAL SIMILARITIES

Having a homogeneous representation of temporal expressions in text in the form of temporal intervals allows us to compare different temporal scopes. More interestingly, it is possible to define similarities between the temporal scope of a query and the temporal scope of a document, thus estimating a *purely-temporal* relevance for retrieval purposes.

We present two purely-temporal similarity models:

(1) **Temporal BM25 (TBM25)**: A frequency-weighting scheme, specialized for temporal scopes, as a simple adaptation of the well-known Okapi BM25 similarity [43] to the temporal domain.
(2) **Temporal Metric Space Model (TMSM)**: A class of novel similarity models based on generalized metrics over the temporal scopes of documents and queries. Some of these models were originally introduces in [9].

We show, in Section 6, how these purely-temporal models can be combined with traditional (purely-textual) models to generate the final ranking of documents.

4.1 TMB25: Temporal BM25

The Okapi BM25 [43] is a well-known text similarity based on a weighting scheme that uses term frequencies, the inverse document frequency, and the length of the document. The scheme gives higher weight to query terms that are rare in the document collection (inverse document frequency), and it favors documents where the query term is more frequent, while penalizing documents with too many terms.

TBM25 works exactly like BM25, with the only difference that it can only use the time intervals extracted from documents and queries, rather than their original text. This can be easily achieved in practice in the following way:

- Create T_D, the temporal scope of document D;
- Create a new textual document $\hat{D}$, by including each interval $t \in T_D$, repeated as many times as there are timexes in D whose interpretation is t;
- Add document $\hat{D}$ to the document collection (disabling stop word or number elimination).

By doing so, we specialize the BM25 weighting scheme to estimate the temporal similarity between document and query: Instead of constructing a bag-of-words using the text terms of the document, we use the temporal intervals to construct a *bag-of-intervals*.

Definition 4.1 (TBM25). Given a query Q and a document D, let $T_Q \subseteq \Delta_{\mathbb{T}}$ and $T_D \subseteq \Delta_{\mathbb{T}}$ be their respective temporal scopes. The temporal similarity between Q and D is

$$\underset{\text{TBM25}}{\text{sim}}(Q, D) = \sum_{t_Q \in T_Q} IDF(t_Q) * \frac{f(t_Q, T_D) * (k_1 + 1)}{f(t_Q, T_D) + k_1 * (1 - b + b * \frac{|T_D|}{avg(|T_D|)})},$$

where $f(t_Q, T_D)$ is the frequency count of the interval t_Q from the query in the temporal scope of document D, $|T_D|$ is the number of intervals in document D, $avg(|T_D|)$ is the average number of intervals per document, $IDF(t_Q)$ is the inverse document frequency of t_Q (i.e., an inverse function of the number of documents in which it occurs, as a measure of its specificity), k_1 and b are parameters to control, respectively, the term-frequency saturation and the effect of temporal scope size interpretation.

Similarly to BM25, TBM25 captures the temporal similarity because it favors documents where the query interval is more frequent and the number of intervals is lower, while weighting the query intervals depending on their rarity in the document collection. TBM25 is appealing because of its simplicity and its renowned effectiveness for textual retrieval. It is also straightforward to incorporate into an existing search engine that already provides a standard BM25 implementation: a new index can be generated using, as dictionary, the set of all temporal intervals extracted from the document collection; then, the standard BM25 would work out-of-the-box as a TBM25 model.

However, the model has a major drawback: If a document interval is not identical to at least one of the query intervals, then it is not considered for frequency counting. This means that intervals in the document that are temporally close to the query intervals may not contribute to the document's similarity score. This is even true with very close and related intervals such as "`31 December 2018`" and "`1st of January 2019`" (because their interval representations are different), and with overlapping intervals like "`25 December 2018`" and "`December 2018`" (because with a granularity smaller than a month these are still two separate intervals). This problem is analogous to the well-known problem of *vocabulary mismatch* in traditional IR [22]. In this paper, we refer to a model like TBM25 as a *unigram model*. Unigram

models treat time intervals as atomic dictionary terms. Another example of a unigram model is the baseline language model used in [4].

In order to capture the temporal similarity of non-identical intervals, depending on how close or overlapped these intervals are on the timeline, like Berberich et al. [4], we replace the representation of a temporal expression with the set of time instants it refers to. Differently from their work, we model their similarity with Temporal Metric Space Models (TMSM), capturing a more fine-grained notion of distance between time intervals. In Section 7, we fully describe and analyze the differences between the models introduced by Berberich et al., and our TMSM models. In Section 8, we also compare the two classes of models on large test collections.

4.2 TMSM: Temporal Metric Space Models

The TBM25 model is not able to take advantage of the richer numerical representation we provided in Section 3.1 because it considers each temporal interval as a single term, and the match between intervals has to be exact. The TMSM model, instead, uses a *distance function* δ^* between document and query intervals.

Definition 4.2 (Temporal distance, δ^*). Given a query Q and a document D, let $T_Q \subseteq \Delta_{\mathbb{T}}$ and $T_D \subseteq \Delta_{\mathbb{T}}$ be their respective temporal scopes. The *temporal distance* function δ^*, $\delta^* : \mathcal{P}(\Delta_{\mathbb{T}}) \times \mathcal{P}(\Delta_{\mathbb{T}}) \rightarrow \mathbb{R}$, is a real-valued distance function between the temporal scopes T_Q and T_D.

In Section 5, we introduce several instances of δ^*, each of which captures different properties of the distance between documents and queries in the temporal domain. We discuss the differences of each distance function in great detail in Section 7. In the experimental section (Section 8), we show how the choice of distance is collection-dependent, while, at the same time, all distances exhibit a significant improvement in effectiveness against the simpler TBM25 model.

Because the temporal distance δ^* is a measure of the *dissimilarity* between a document and a query in the temporal domain, we need to transform it into a *similarity* score, in oder to embed its contribution into a ranking system. Although any strictly decreasing function is suitable for transforming a dissimilarity score into a similarity score, we use *exponential decay*, as used in prior works (e.g., New Event Detection [21], and Web Documents Clustering [50]) to estimate a similarity score given a distance value.

Definition 4.3 (TMSM). Given a temporal distance function $\delta^* : \mathcal{P}(\Delta_{\mathbb{T}}) \times \mathcal{P}(\Delta_{\mathbb{T}}) \rightarrow \mathbb{R}$, the *temporal similarity* between a query Q and a document D via δ^* is

$$\underset{\text{TMSM}}{\text{sim}}(Q, D, \delta^*) = e^{-\delta^*(T_Q, T_D)}.$$

This transformation is also well known in psychology to estimate the similarity of entities and situations experienced by individuals, given the metric distances between their characteristics [47]. While there may be other dissimilarity-to-similarity transformation functions, exploring them is orthogonal to the scope of this paper.

5 TEMPORAL METRIC SPACES

The TMSM model introduced in the previous section uses a distance function $\delta^*(T_Q, T_D)$ (Definition 4.2) to model the temporal distance between a query and a document. This function defines a distance between the temporal scopes of a query, T_Q, and a document, T_D, thus it is a distance between sets of intervals. We define δ^* as an aggregate temporal distance δ^{agg} of interval-to-interval distances δ, computed between all pairs of intervals from T_Q and T_D.

Definition 5.1 (Aggregate temporal distance, δ^{agg}). Given an interval-to-interval distance function $\delta : \Delta \times \Delta \rightarrow \mathbb{R}$, and an aggregate function $agg : \mathcal{P}(\mathbb{R}) \rightarrow \mathbb{R}$, δ^{agg} is the *aggregate* temporal distance between all pairs of query and

document intervals:

$$\delta^{agg}(T_Q, T_D) = agg(\{\delta([a_Q, b_Q], [a_D, b_D]) \mid [a_Q, b_Q] \in T_Q, [a_D, b_D] \in T_D\})$$

The aggregation function *agg* can be any aggregation that maps a set of distances into a single distance. Although this can be an arbitrary complex function, we consider three simple and efficient options for which the underlying intuition is straightforward and relevant for the task: *minimum*, *maximum*, and *average* of the distances. We now proceed to explain the intuition behind each of these options.

The minimum distance δ^{min} takes into account only the closest pair of intervals, which leads to the minimum distance and thus the highest similarity between the two temporal scopes. For example, it is common to have one interval in the query temporal scope T_Q and many in the document temporal scope T_D. By using δ^{min}, we are considering only the most favorable interval in the document, disregarding the ones that are further away from the query. Otherwise stated, the minimum distance favors high temporal recall, at the expense of a lower temporal precision.

Conversely, the maximum distance δ^{max} ensures that only the worst pair is taken into account: it is the distance between the farthest intervals. For example, in order to have $\delta^{max}(T_Q, T_D) = 0$, all intervals in the document should have a zero distance from all the intervals in the query. Because $\delta^{max} \geq \delta^{min}$ for any temporal scope, this aggregation results in a lower similarity score, thus lowering the recall, with possibly higher precision.

Finally, the average distance δ^{avg} takes into account all the pairs, by averaging the distance between each interval in the document and each interval in the query, in an attempt to balance between recall and precision.

When combined with a text similarity, the minimum aggregation function δ^{min} consistently produced the best results in our evaluation, as we show in Section 8. This suggests that the temporal similarity has a role of promoting documents loosely similar in time, while precision is mainly handled by its textual counterpart.

5.1 Interval-to-interval distances

In this section, we introduce several instances of δ, each of which captures different properties of the temporal relevance of a document for a given temporal query. In the experiments, we show that a specific distance may be more suitable than others depending on the specific collection of documents and queries.

Definition 5.2 (Metric space). A metric space is an ordered pair (M, d) where M is a set and d is a metric distance on M, i.e., a function $d : M \times M \to \mathbb{R}$ such that for any $x, y, z \in M$ the following metric properties hold:

(1)	*Non-negativity*	$d(x, y) \geq 0$
(2a)	*Identity of indiscernibles* ($\Rightarrow$)	$d(x, y) = 0 \Rightarrow x = y$
(2b)	*Identity of indiscernibles* ($\Leftarrow$)	$x = y \Rightarrow d(x, y) = 0$
(3)	*Symmetry*	$d(x, y) = d(y, x)$
(4)	*Triangle inequality*	$d(x, z) \leq d(x, y) + d(y, z)$

In addition to metric distances we consider two relaxed metric definitions (also known as *generalized* metrics): *quasi-metric* distances and *hemi-metric* distances.

Definition 5.3. [Quasi-metric space] In a quasi-metric space, only axioms (1), (2) and (4) from Definition 5.2 hold.

Dropping the symmetry axiom is useful in our context: queries and documents are inherently different and so are the temporal intervals found in them. We define query-to-document quasi-metric distances to reflect this asymmetry.

Definition 5.4. [Hemi-metric space] A hemi-metric space further relaxes the metric conditions, only satisfying axioms (1), (2b) but not (2a) (i.e., $d(x, y) = 0$ does not imply that $x = y$), and (4).

For instance, in comparison with quasi-metrics, hemi-metrics allow the existence of several y elements for which $d(x, y) = 0$. Clearly, this is a desirable property if we want a set of document time intervals with some common features (e.g., being all contained in the query interval) to have all the same distance with respect to the query interval.

The following distance definitions focus on different aspects of temporal intervals and relations between them. Each distance defines a different metric space (or generalized metric space) and can be suitable under certain conditions. However, as we will show through several experiments, some distances generally performs better then others, in both precision and recall measures.

5.1.1 Metric distances. In the 2-dimensional space, the Manhattan distance (also known as L_1 Mikowski distance) is a metric distance defined as follows [7]:

$$d_1(p, q) = |p_1 - q_1| + |p_2 - q_2|$$

where $p = (p_1, p_2)$ and $q = (q_1, q_2)$ are pairs, i.e., members of $\mathbb{R} \times \mathbb{R}$. Our first distance between temporal intervals applies the Manhattan distance by assuming that the two dimensions refer to the begin and the end of an interval, respectively.

Definition 5.5 (Manhattan distance). Given a time interval $[a_Q, b_Q]$ from the temporal scope T_Q of a query Q, and a time interval $[a_D, b_D]$ from the temporal scope T_D of a document D, the Manhattan distance between the two intervals is defined as:

$$\delta_{man}([a_Q, b_Q], [a_D, b_D]) = |a_Q - a_D| + |b_Q - b_D|$$

The Euclidean distance (also known as L_2 Mikowski distance) is the straight line distance between two points:

$$d_2(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}$$

We apply the Euclidean distance to our interval definition so that p and q are the two intervals in comparison.

Definition 5.6 (Euclidean distance). Given an interval $[a_Q, b_Q]$ from the temporal scope T_Q of a query Q and an interval $[a_D, b_D]$ from the temporal scope T_D of a document D, the Euclidean distance between these intervals is defined as:

$$\delta_{eucl}([a_Q, b_Q], [a_D, b_D]) = \sqrt{(a_Q - a_D)^2 + (b_Q - b_D)^2}$$

The Manhattan distance and the Euclidean distance are metrics, since they satisfy all the four metric properties. This does not hold for the next generalized metric distance definitions.

5.1.2 Hemi-metric distances. The *coverage* of an interval by another interval (how much of the first one is covered by the second one) is an important temporal relation that cannot be captured by the previously defined distances. To capture the coverage, we introduce two generalized metrics for which the distance of two intervals is zero if one interval is totally contained in (or *covered* by) the other: a *query-biased* coverage distance to maximize the similarity if the document interval completely covers the query interval, and a *document-biased* coverage distance to maximize the similarity if the query interval completely covers the document interval.

Definition 5.7 (Query-Coverage distance). Given an interval $[a_Q, b_Q]$ from the temporal scope T_Q of a query Q, and an interval $[a_D, b_D]$ from the temporal scope T_D of a document D, the Query-Coverage distance is defined as:

$$\delta_{covQ}([a_Q, b_Q], [a_D, b_D]) = (b_Q - a_Q) - (\min\{b_Q, b_D\} - \max\{a_Q, a_D\})$$

The intuitive meaning of this distance is that if the interval of the query is fully covered by the interval from the document, then the distance is zero. Otherwise, it is equal to the length of the "uncovered" section of the query interval.

The Query-Coverage distance is suitable when, in referring to the same event, the temporal intervals mentioned in the query are narrower than the temporal intervals in the relevant documents. This is because when the document intervals cover the query intervals, the Query-Coverage distance is lower, resulting in a higher similarity.

In the same way as we defined the Query-Coverage distance, we define a Document-Coverage distance to boost the similarity if the document interval is covered by the query interval.

Definition 5.8 (Document-Coverage distance). Given an interval $[a_Q, b_Q]$ from the temporal scope T_Q of a query Q, and an interval $[a_D, b_D]$ from the temporal scope T_D of a document D, the Document-Coverage distance is defined as:

$$\delta_{covD}([a_Q, b_Q], [a_D, b_D]) = (b_D - a_D) - (\min\{b_Q, b_D\} - \max\{a_Q, a_D\})$$

The Document-Coverage distance is the appropriate distance when relevant documents have their time interval covered by the interval in the query, i.e., for temporally vague query and precise dates in documents. The intuition is that the users, while searching for information, have less precise knowledge on the dates of the events than the document describing them. For example, they may be looking for information about the Indian Ocean earthquake and tsunami of 2004 using the query ***2004 tsunami***, while the document describing the event contains only the exact timex "`12/26/2004 00:58:49`".

5.1.3 Quasi-metric distances. While the coverage distances capture overlapping and inclusion of two intervals, they are not able to discern between two intervals that are identical and two intervals that are just included in one another but very dissimilar, such as "`3 August 1992`" and "`the nineties`". A *quasi-metric* is a function that satisfies all the metric distance properties except symmetry. Unlike hemi-metrics, the identity of indiscernibles satisfies both the axioms (2b) and (2a) of Definition 5.2.

A quasi-metric can be obtained by averaging, or linearly combining with positive weights, a metric distance with a hemi-metric distance. To show this, let a metric distance be $d_M(x, y)$, a hemi-metric distance $d_H(x, y)$, and its weighted sum $d_Q = \alpha d_M(x, y) + \beta d_H(x, y)$, with α and β positive non-zero values:

(1) Given that $\alpha d_M(x, y) \geq 0$ and $\beta d_H(x, y) \geq 0$, their sum $d_Q(x, y) = \alpha d_M(x, y) + \beta d_M(x, y)$ is also non-negative.
(2a) Suppose $\alpha d_M(x, y) + \beta d_H(x, y) = 0$. This is only possible if $d_M(x, y) = 0$, being α and β positive and non-zero, and d_M and d_H non-negative. Thus, $\alpha d_M(x, y) = 0$. Because $d_M(x, y)$ is a metric, $\alpha d_M(x, y) = 0 \Rightarrow x = y$.
(2b) If $x = y$ than $d_M(x, y) = 0$ and $d_H(x, y) = 0$, as both satisfy the ($\Leftarrow$) implication of the property. It follows that for $x = y$, $\alpha d_M(x, y) + \beta d_H(x, y) = \alpha \cdot 0 + \beta \cdot 0 = 0$.
(4) We know that this property holds for both d_M and d_H, that is $d_M(x, z) \leq d_M(x, y) + d_M(y, z)$ and $d_H(x, z) \leq d_H(x, y) + d_H(y, z)$. We can rewrite the inequalities as $d_M(x, z) - d_M(x, y) - d_M(y, z) \leq 0$ and $d_H(x, z) - d_H(x, y) - d_H(y, z) \leq 0$. Replacing, for simplicity, the left-hand side of the two inequalities respectively with a and b, we have that $a \leq 0 \wedge b \leq 0 \implies a + b \leq 0$. Therefore, for all $\alpha, \beta > 0$, $\alpha a + \beta b \leq 0$.

We define the Manhattan-Query-Coverage distance as the average of the Manhattan distance and the Query-Coverage distance.

Definition 5.9 (Manhattan-Query-Coverage distance). Given an interval $[a_Q, b_Q]$ from the temporal scope T_Q of a query Q and an interval $[a_D, b_D]$ from the temporal scope T_D of a document D, the Manhattan-Query-Coverage distance is defined as:

$$\delta_{mcovQ}([a_Q, b_Q], [a_D, b_D]) = \frac{\delta_{man}([a_Q, b_Q], [a_D, b_D]) + \delta_{covQ}([a_Q, b_Q], [a_D, b_D])}{2}$$

By averaging a metric distance with a coverage distance, we obtained a distance that satisfies the identity of indiscernibles while taking into account the asymmetric coverage. More specifically, the document intervals that cover the query interval have lower distance, while only the exactly matching intervals have zero distance.

Similarly to the Manhattan-Query-Coverage distance, the Manhattan-Document-Coverage distance is a quasi-metric obtained by averaging the Manhattan distance with the Document-Coverage distance.

Definition 5.10 (Manhattan-Document-Coverage distance). Given an interval $[a_Q, b_Q]$ from the temporal scope T_Q of a query Q and an interval $[a_D, b_D]$ from the temporal scope T_D of a document D, the Manhattan-Document-Coverage distance is defined as:

$$\delta_{mcovD}([a_Q, b_Q], [a_D, b_D]) = \frac{\delta_{man}([a_Q, b_Q], [a_D, b_D]) + \delta_{covD}([a_Q, b_Q], [a_D, b_D])}{2}$$

The δ_{mcovD} distance is low for document intervals which are covered by the query interval, however, conversely to the Document-Coverage distance, only the exactly matching interval has zero distance.

In a similar fashion, we define the the quasi-metrics combining the Euclidean distance and the coverage distances.

Definition 5.11 (Euclidean-Query-Coverage distance). Given an interval $[a_Q, b_Q]$ from the temporal scope T_Q of a query Q and an interval $[a_D, b_D]$ from the temporal scope T_D of a document D, the Euclidean-Document-Coverage distance is defined as:

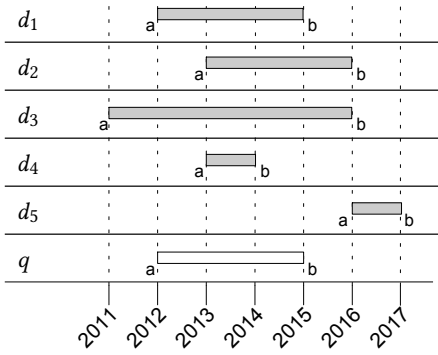
$$\delta_{ecovD}([a_Q, b_Q], [a_D, b_D]) = \frac{\delta_{eucl}([a_Q, b_Q], [a_D, b_D]) + \delta_{covQ}([a_Q, b_Q], [a_D, b_D])}{2}$$

Definition 5.12 (Euclidean-Document-Coverage distance). Given an interval $[a_Q, b_Q]$ from the temporal scope T_Q of a query Q and an interval $[a_D, b_D]$ from the temporal scope T_D of a document D, the Euclidean-Document-Coverage distance is defined as:

$$\delta_{ecovD}([a_Q, b_Q], [a_D, b_D]) = \frac{\delta_{eucl}([a_Q, b_Q], [a_D, b_D]) + \delta_{covD}([a_Q, b_Q], [a_D, b_D])}{2}$$

In Table 2, we show examples of the 8 distances we defined. Each distance is computed for the query interval $q = [2012, 2015]$ with respect to 5 different document intervals $d_1, \ldots, d_5$. When the document interval is exactly the same as the query ($d_1 = q$), the distance is zero for all the four distances, as a consequence of the right-to-left identity of indiscernibles (Definition 5.2, (2b)). However the converse is not true for the two coverage distances: for d_3 and d_4, the distance is 0 even if the two are not equal to q. It follows that, for coverage distances, the implication of identity of indiscernibles is true only in one direction. It is worth noting that the sum of these two complementary coverage distances always results in the Manhattan distance.

The quasi-metric distances (i.e., δ_{mcovQ}, δ_{mcovD}, δ_{ecovQ} and δ_{ecovD}) offer a balance between coverage and mere distance, mitigating the effects that the hemi-metrics have on full coverage. In the last column of Table 2, $\delta_{ecovD}(d_1, q) = 0$ only for the exact same interval. When the document interval is completely covered, but not the same as the query interval, then $\delta_{ecovD}(d_4, q) = 0.705$, which is higher than 0 but still less then the case of a partially covered interval, such as $\delta_{ecovD}(d_2, q) = 1.205$.



	$\delta(d_i, q)$							
	δ_{man}	δ_{eucl}	δ_{covQ}	δ_{covD}	δ_{mcovQ}	δ_{mcovD}	δ_{ecovQ}	δ_{ecovD}
d_1	0	0	0	0	0	0	0	0
d_2	2	1.41	1	1	1.5	1.5	1.205	1.205
d_3	2	1.41	0	2	1	2	0.705	1.705
d_4	2	1.41	2	0	2	1	1.705	0.705
d_5	6	4.47	4	2	5	4	4.235	3.235

Fig. 2. Examples of the eight distance functions on five different document intervals $d_1, \ldots, d_5$, for a fixed query interval q. Each document interval shows a different case of containment or overlap with the query interval. The distances only agree when the document and query intervals exactly match ($d_1 = q$).

6 COMBINING TEMPORAL AND TEXTUAL SIMILARITIES

In Section 4, we introduced our purely-temporal similarities, $\overset{\text{sim}}{\text{TBM25}}$ and $\overset{\text{sim}}{\text{TMSM}}$. For the final ranking, it is important to combine a temporal similarity score with a non-temporal (purely-textual) similarity score. The choice of the non-temporal similarity function is orthogonal to the scope of this work. In this paper, we consider the Okapi BM25 [43] and the Language Model Jelinek-Mercer (LMJM) text similarities as non-temporal similarity functions.

Given a temporal similarity $\overset{\text{sim}}{\text{TIME}}$ and a text similarity $\overset{\text{sim}}{\text{TEXT}}$, a simple yet effective technique to combine the two similarity scores is a convex combination:

$$\text{sim}(Q, D, \delta^*) = \alpha \overset{\text{sim}}{\text{TIME}}(Q, D, \delta^*) + (1 - \alpha) \overset{\text{sim}}{\text{TEXT}}(Q, D)$$

The parameter α, $\alpha \in [0, 1]$, denotes the weight of the temporal similarity with respect to the text similarity. For example, setting $\alpha = 1$ only ranks documents according to their temporal similarity to the query; $\alpha = 0.5$ corresponds to averaging the temporal and non-temporal similarities; $\alpha = 0$ is the equivalent of only taking the text relevance into consideration.

Even a simple linear combination like this hides nontrivial subtleties that, if not carefully considered, can possibly render the resulting scores poor, ultimately hurting the ranking's effectiveness [35]. The main issue is that combining scores from different models requires a proper score *normalization* before combination can happen. This is especially necessary if the distributions of the two scores being combined vary substantially between the two systems [42]. Cases like this are often true when the features considered (e.g., in our case, text terms vs. time intervals) and the similarity schemes (e.g., terms frequency vs. metric distances) are very different, which is clearly the case in our setting.

To further explain the effects of different score distributions when we combine them with or without prior normalization, we show an example in Figure 3. The figure depicts the similarity scores distribution for $\overset{\text{sim}}{\text{TIME}}$ (diamond markers) and $\overset{\text{sim}}{\text{TEXT}}$ (square markers) of a top-5 retrieval. While the scores are synthetically generated to fit a small example, the shape of the scores distribution for text and time reflects the actual one. Also for simplicity, text retrieval and time retrieval documents are two disjoint sets in this example. The black markers represents the original, unnormalized

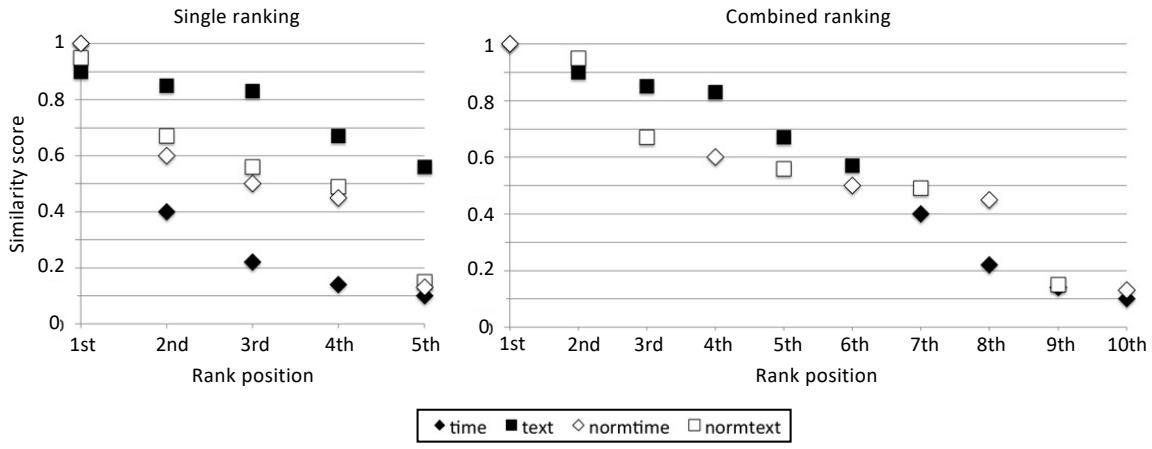


Fig. 3. This example shows the original and normalized scores of the top-5 results for temporal and text similarity. The two similarities produce different score distributions. In combined ranking, normalization of scores affects the ranking position of the 10 documents.

scores, showing two different distributions for time and text scores. The white markers represents the scores of the same documents after a distribution-based normalization [17] is performed. On the left hand-side, we show all these scores before combination. Note that any proper normalization do not change the ranking of the documents. On the right hand-side instead, we show the impact on the ranking in combining the scores with and without prior normalization. As it is clear from the figure, combining unnormalized and normalized scores results in different rankings of the documents. In this example, keeping the scores unnormalized (black markers) has the effect of ranking all the top-5 results for $\overset{\text{sim}}{\text{Text}}$ close together and higher up in the ranking, leaving most of the top-5 documents for $\overset{\text{sim}}{\text{Time}}$ at the bottom of the ranking. This in an effect of the fact that the original scores for $\overset{\text{sim}}{\text{Text}}$ are generally higher than those of $\overset{\text{sim}}{\text{Time}}$. On the other hand, normalized scores (white markers) are more similar to each other and thus their combination results in a more alternating ranking of top-textual and top-temporal results.

We now proceed to discuss and experimentally evaluate some of the existing methods to normalize the scores prior to their combination. The simplest normalization method is a linear transformation, in which all the scores are multiplied by the same constant to fit in a range, such as $[0, 1]$. We then consider normalization methods that take into account not only the score range, but also the inner distribution of scores for the two similarities. We apply the method presented in [17] to map scores to a common target distribution. The choice of an appropriate target distribution is critical to this method. We instantiate the method using 3 different target distributions: the text scores distribution, the temporal scores distribution, and the average distribution of the previous two.

In summary, we consider and evaluate 4 different normalization methods:

- Linear transformation ($norm_{lin}$): the scores are linearly scaled within the same range.
- Textual score distribution ($norm_{text}$): a distribution-based method where the target distribution is the average distribution of text similarities $\overset{\text{sim}}{\text{Text}}$.
- Time score distribution ($norm_{time}$): a distribution-based method where the target distribution is the average distribution of the time similarities $\overset{\text{sim}}{\text{Time}}$.

- Average score distribution ($norm_{avg}$): a distribution-based method where the target distribution is the average distribution of time and text similarities.

We evaluate the normalization methods over top-k precision and recall, with a fixed $\alpha = 0.5$ for the linear combination, that is, the arithmetic mean of the two similarities. We conducted these experiments on the Novelty 2004 collection [49].

	Precision			**Recall**		
	P@5	**P@10**	**P@20**	**R@5**	**R@10**	**R@20**
$norm_{lin}$	0.846	**0.854**	**0.827**	0.183	**0.372**	**0.721**
$norm_{text}$	0.846	0.839	0.785	0.183	0.366	0.687
$norm_{time}$	0.815	0.823	0.808	0.178	0.357	0.704
$norm_{avg}$	**0.877**	0.846	0.808	**0.191**	0.367	0.704

Table 3. Top-k precision and recall for the considered normalization methods with $\alpha = 0.5$ on the Novelty 2004 collection. Compared to linear transformation, average score distribution shows better results only for top-5 documents.

Table 3 shows that, while the average score distribution normalization yields better precision and recall for the top-5 results, linear transformations provides higher precision and recall for greater values of k. Textual score distribution and time score distribution do not yield any improvement.

Given that the linear transformation is overall more effective and more efficient—it is a rather simpler computation—we apply this method for all the experiments presented in Section 8.

7 QUALITATIVE ANALYSIS

Before proceeding with our experimental evaluation (Section 8), in this section, we analyze the fundamental differences between the temporal similarities introduced in this paper, and the relevant ones from the literature. This section provides an easy-to-understand visual insight into the effects of each similarity measure.

Unigram. As we discussed in Section 4.1, our TBM25 similarity (Definition 4.1) and the baseline model used in [4] are examples of unigram models, where each temporal interval is treated like a single dictionary term. A unigram model is able to capture the similarity between "`Christmas day last year`" and "`the 25th of December 2020`": given that their NLP annotation is the same, they are represented with the same dictionary term. For example they could both be represented by the string `20201225_20201225`, that will be indexed as a unique, atomic token like any other dictionary term. However, with this model, even the slightest difference between the document interval and the query interval would result in zero similarity score. For example, the two timexes "`This December`" and "`From Dec. 1 to Christmas day`" will result in two different tokens, `20211201_20211231` and `20211201_20211225`.

We depict this *binary* notion of similarity of the left-hand side of Figure 4, using a 10-year time span from 2006 to 2016, and a fixed query interval [2009, 2012], marked in the figure as a red cross. The figure uses a heatmap to indicate the temporal similarity score for every possible time interval from a document—the darker the color, the lower the similarity to the query interval, and a gray color indicates zero or undefined similarity. For a unigram model, there are only two possible scenarios: either the document interval "matches" the query interval, or it does not. And if it does, the similarity is 1—the maximum possible.

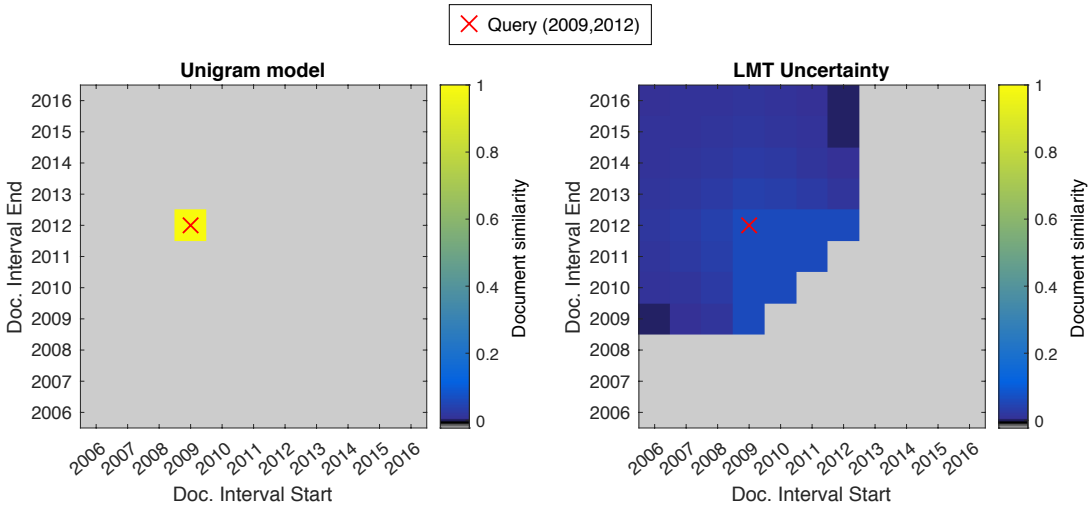


Fig. 4. Given the same query [2009, 2012], each cell represents the document similarity score for each document interval in the range 2006–2016, with yearly granularity. The similarity score is displayed using a color map in the range [0,1], with maximum similarity equal to one; gray spots have zero or undefined similarity.

LMT with Uncertainty. Prior to this paper, the state of the art for temporal similarity was the *Language Model with Temporal Uncertainty* (LMTU) [4], and the models derived from it using learning to rank [30]. The main idea behind *Temporal Uncertainty* is that a user, while expressing their temporal intent, is not completely certain about the precise temporal scope of its search. For instance, a user can remember that the last Obama election was in the year 2012, without knowing the exact date, therefore issuing the query ***Obama election 2012*** instead of ***Obama election November 6, 2012***. While this model is still based on a text similarity, the temporal expression is represented as the set of time units instead of as a unique token. LMTU captures not only the exact match between intervals, but also the similarity of intervals that have a non-zero intersection with the query interval. In this language model, the document interval t_D can refer to any interval of any size, contained in T_D. The same is true for the query interval t_Q, so $|T_D|$ and $|T_Q|$ are the number of possible intervals that the document interval and the query interval can refer to, respectively. The probability of generating t_Q from the language model of t_D is defined as:

$$\frac{|t_D \cap t_Q|}{|t_D| \cdot |t_Q|}$$

that is, the ratio between the intersection of the two sets of possible intervals in t_Q and in t_D and all the pairs of possible intervals in t_Q and t_D.

The similarity score of the LMTU model and its relation with overlapping intervals is visualized on the right-hand side of Figure 4. The triangle representing smaller intervals than the query interval (i.e., contained in the query interval [2009, 2012]) exhibit the highest scores; the bigger intervals that contain [2009, 2012] show a slightly lower value; all the other intervals without any overlap have zero score.

Differently from unigram models (left-hand side of Figure 4), this model is able to capture the similarity between two intervals even if these two intervals do not perfectly match. LMTU results in higher similarity if the time in the document is contained in the time of the query. As we will show later, this behavior is similar to the document-coverage

similarities we introduce in this paper (δ_{covD}, δ_{mcovD}, and δ_{ecovD}). However, LMTU only gives a positive score if there is some overlap between the two intervals; if there is no overlap, the similarity score is zero, as shown in the top-right and bottom-left corners of the LMTU plot in Figure 4.

This last property, however, is a limitation for this model. For instance, consider the query timex t_Q "`7 December 2019`" and two documents, D_1 with the timex t_{D_1} "`8 December 2019`" and D_2 with the timex t_{D_2} "`11 November 1918`". The interval in D_1, being the day after t_Q, is clearly more related to the query than t_{D_2}, being the latter interval the beginning of World War I. However, because both t_{D_1} and t_{D_2} have no overlap with t_Q, their LMTU similarity scores are both zero. The model can capture the uncertainty of the user when expressing their temporal needs *only if* the relevant documents overlap with the query. If relevant documents do not overlap with the query interval, this model will fail in retrieving them (unless their textual scores are high enough). Users, in order to retrieve relevant documents, would be forced to provide larger query intervals, which would result in too many documents receiving high temporal scores, defeating the purpose of assigning temporal scores in the first place.

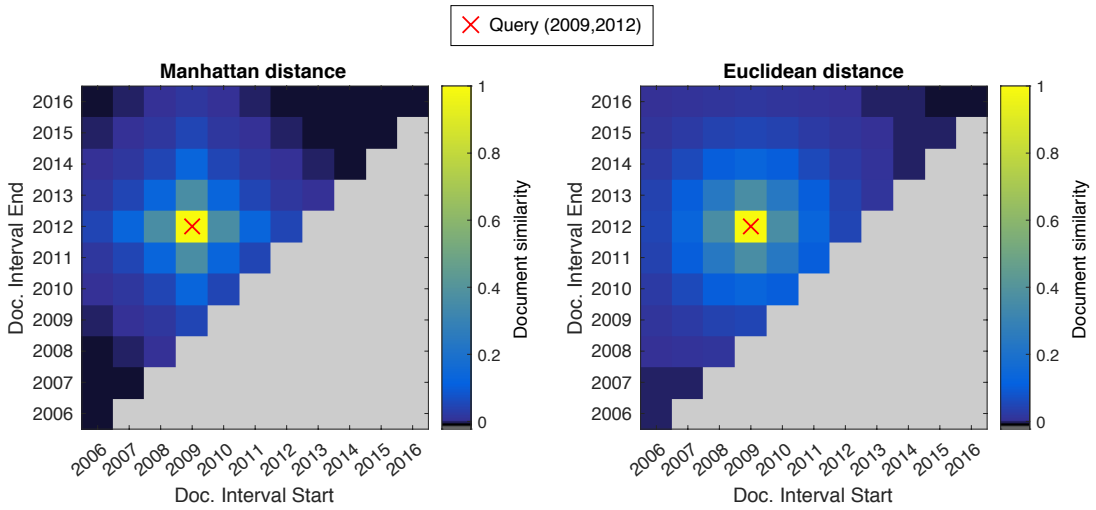


Fig. 5. Similarity scores of metric distances for the query [2009, 2012] and all the document intervals in the range 2006–2016. Gray cells have zero or undefined similarity.

Metric similarities. We now discuss our metric-based TMSM similarities. Like unigram models, the Manhattan and Euclidean similarities have their maximum score only when query interval and document interval are the same (*identity of indiscernibles*). However, the similarity score is not binary, but proportional to the distance between the two intervals. Moreover, not being based on set operations, like the LMTU similarity, they are able to give different and meaningful scores to the two previous examples, t_{D_1} and t_{D_2}. The similarity between "`7 December 2019`" and "`8 December 2019`" is $e^{-1} = 0.368$, while the similarity between "`7 December 2019`" and "`from 28 July 1914 to 11 November 1918`" is a negligible e^{-72493}. Figure 5 highlights the differences between the two similarity models. In Manhattan, the diagonal cells are doubly penalized with respect to the other adjacent cells. This is due to the nature of Manhattan distance: it is the distance between two points in a grid, based on a strictly horizontal/vertical paths, much like the grid street topography of Manhattan. Conversely, the Euclidean distance shows a smoother decay of similarity score with a circular pattern around the query.

A shortcoming of the similarities based on metric distances is that they are not able to capture the containment of intervals. Because of their symmetry, they make no distinction between query intervals and document intervals, despite queries and documents serve different purposes, with queries often acting like "filters" for the relevant documents the user is looking for.

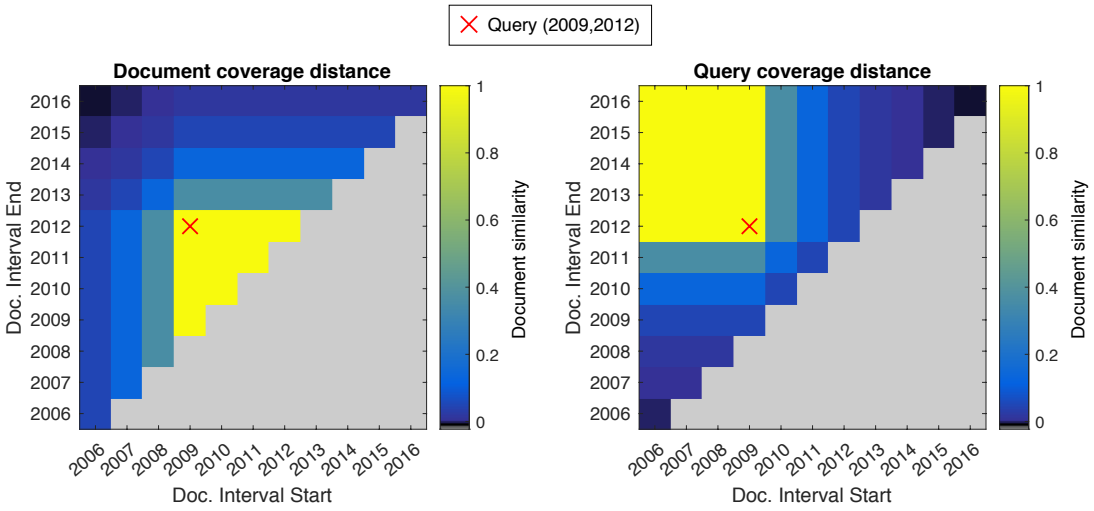


Fig. 6. Similarity scores of hemi-metric distances for the query [2009, 2012] and all the document intervals in the range 2006–2016. Gray cells have zero or undefined similarity.

Hemi-metric similarities. The coverage-based similarities share properties of all the similarities so far discussed: their maximum score occurs with a perfect match of the two intervals, and they are able to differentiate between very distant and close intervals without these intervals having any overlap. Moreover, like LMTU, both the document coverage and the query coverage are asymmetric, and take into consideration the containment of the interval of the document in the interval of the query, or vice versa. As shown in Figure 6, the triangle of the intervals contained in [2009, 2012] has the highest document-coverage score, just like the LMTU model. However, in LMTU the maximum score is penalized by the query interval size, while in document-coverage the highest score coincides with the maximum score of one. The figure also clearly shows the opposite coverage notion generated by the query-coverage distance. None of the other existing similarity measures can model query-coverage, including LMTU. Interestingly, in some of the experiments we ran, query-coverage performed better that the other models, showing that some of the temporal query intents are indeed based on query-coverage, rather than document-coverage or symmetrical distances.

One shortcoming of coverage-based scores is that they cannot differentiate between a perfectly matching document interval and a document interval contained in the query interval (or vice versa). For example, in document-coverage, if the query interval is "`from 28 July 1914 to 11 November 1918`", the document intervals "`28 July 1914 - 11 November 1918`" and "`2 February 1916`" will both have equal maximum score of one. This is exacerbated in the extreme case of very large query intervals. For example, for a query spanning two millenia, all the documents would have the same score under the document-coverage distance.

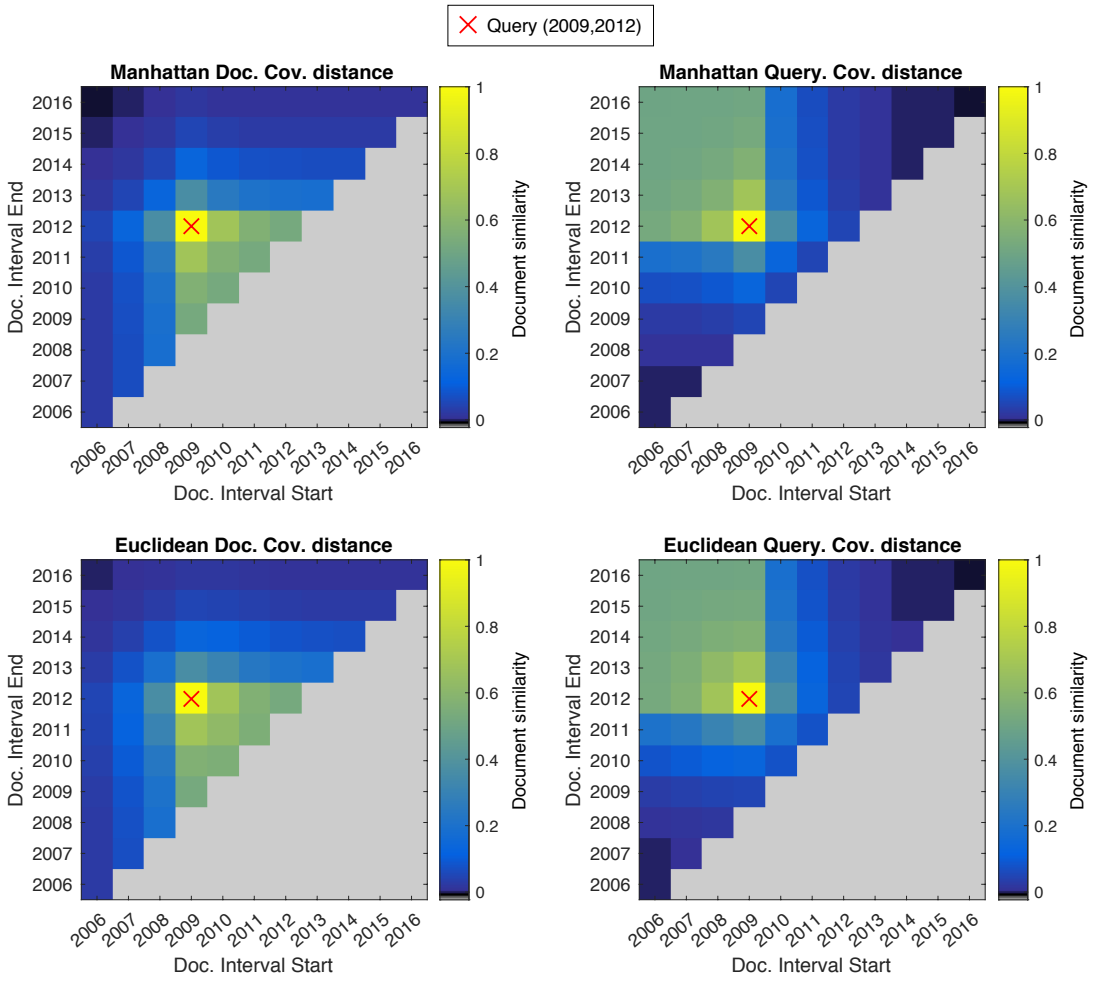


Fig. 7. Similarity scores of quasi-metric distances for the query [2009, 2012] and all the document intervals in the range 2006–2016. Gray cells have zero or undefined similarity.

Quasi-metric similarities. When dealing with queries for which it is important to make this distinction, we can make a coverage-based score act like a metric score by combining a coverage distance with a metric, obtaining quasi-metric scores. The similarities built upon the two quasi-metric models, shown in Figure 7, take into consideration both the distance between the ends of intervals and the coverage of one interval over the other, thus including all the temporal relations considered so far in one single score. The quasi-metric distances share features with all the previous distances: their similarity scores are maximum when the intervals exactly match one another; their scores decrease gradually with distance; and they are boosted in the coverage area.

8 EXPERIMENTAL EVALUATION

In this section, we present an experimental evaluation of our temporal ranking models aimed at answering the following research questions:

Q1 Previous studies hypothesized that queries without timexes have no temporal intent. Is this true or do most queries, even if devoid of explicit timexes, hide a temporal intent that favors information of particular dates and periods of time? By somehow identifying temporal intent for queries without timexes, can our proposed models improve the retrieval? In Section 8.1.3, we show a simple method to identify *implicit* temporal scopes from queries devoid of any timex. We use these implicit features in all our experiments, showing that queries without timexes still have a temporal intent, and that a temporal model does not need explicit timexes to improve the effectiveness of a purely textual ranking model.

Q2 Do the proposed temporal similarities improve the precision and recall of the retrieval results when combined with a text similarity, in comparison with using the text similarity alone (see Section 8.2.1)?

Q3 How much weight should we assign to the text similarity component in the text-time combination, to gain the highest improvement (see Section 8.2.1)?

Q4 Are the proposed temporal similarities more effective than state of the art TIR models, in the same text-time combination setting (see Section 8.2.2)?

Q5 How does the granularity of time interval representation affect the effectiveness of the temporal models (see Section 8.2.3)?

Q6 Which of the considered distance aggregations is on average the most effective (see Section 8.2.4)?

Q7 Which of all the proposed metric distances is on average the most effective (see Section 8.2.5)?

Before presenting the results that answer these questions, we proceed by describing our experimental setup.

8.1 Experimental setup

The experimental settings that follow describes the tools, the choices and the assumptions involved in evaluating the proposed models on IR standard test collections.

8.1.1 Implementation. We annotated and extracted temporal expressions from documents and queries using HeidelTime [51], a state-of-the-art NLP tool for temporal expressions.

The NLP tool requires the date creation time in order to resolve relative timexes. For this, we built custom Python modules that parse each document collection and their queries, extracting the text body of documents and the metadata containing the date creation time. We used Apache Lucene [6] to compute the text similarity scores using its implementations of the Okapi BM25 [43] similarity and Language Model with Jelinek-Mercer smoothing [62] similarity. We precomputed the textual similarity scores for all queries and documents using Lucene, as well as the measurements for all the systems evaluated, and stored them into a MySQL database. Temporal similarity scores are instead computed on-the-fly. We implemented[3] all the models defined in this paper in Python, including: transformation of interpreted timexes in discrete intervals; generalized metric distances; transformation of granularity on extracted intervals; normalization functions for temporal and text scores; aggregation of distances δ^*; transformations from distance to similarity.

8.1.2 Test Collections. Table 4 shows the five text collections used in our experiments. We use three collections from NIST TREC information retrieval tracks: Novelty 2004 [49], Robust 2004 [56], Robust 2005 [57]. These collections were not specifically constructed for time sentitive queries. We also include two temporal-aware test collections: Temporalia [26], released for the international conference NTCIR-11, and, lastly, the collection, which we henceforth

[3]We provide the main components of the code on the github repository `https://github.com/Strizzo/tmsm`

refer to as LMT-NYT, constructed by Berberich et al. from the New York Times Annotated Corpus [44] to evaluate their Language Model with Temporal Uncertainty [4]. The collections have an average number of timexes per document between 4 and 9. The table also shows the ratio between the number of timexes per document and the number of words per document, showing that a timex appears every 52 to 123 words, and the percentage of timexes for each time granularity (D=day, M=month, Y=year), showing a prevalence of day-wise timexes in all the collections.

		Timex occurrences			Timex granularities			
	Documents	**Tmx/Doc**	**Wrd/Doc**	**Tmx:Wrd**	**% D**	**% M**	**% Y**	**% Other**
Novelty 2004 [49]	1,808	6.46	500	1:77	46.22	12.48	32.18	9.12
Robust 2004 [56]	528,155	6.90	516	1:74	47.67	11.64	33.57	7.12
Robust 2005 [57]	1,033,461	9.14	476	1:52	44.09	7.43	26.16	22.32
Temporalia [26]	3,648,716	3.92	482	1:123	53.47	16.05	30.48	0
LMT-NYT [4, 44]	1,855,140	8.17	574	1:70	49.07	9.40	34 .60	6.93

Table 4. For each collection: total number of documents, statistics on timex and word occurrences (average number of timexes per document as Tmx/Doc, average number of words per document as Wrd/Doc, ratio between timexes and words as Tmx:Wrd) and statistics on the granularities of found timexes (granularity of one day as D, one month as M, one Year as Y and other intervals different from the previous).

8.1.3 Temporal scope of queries. For each query from the five test collections, we construct its temporal scope looking for time intervals from one of four different sources of temporal information. The first three of them are directly available in the query description: the *title*, *description*, and *narrative* of the topic.

(1) Topic title: a brief keyword-based description of the query intent.
(2) Topic description: a one-sentence description of the query.
(3) Topic narrative: one or more sentences that concisely describes the documents relevant to the query.

We run the NLP tools to identify temporal references from any of those, in that order. If the topic title contains time intervals, those intervals will constitute the temporal intervals of the query. Otherwise, we look for intervals in the topic description or, lastly, in the topic narrative. If we can identify temporal intervals from any of these three sources, we refer to the query as being *explicitly temporal*. Examples of explicitly temporal queries found in the considered collections are:

- *China's first spaceflight "Shenzhou" successful launch and retrieval on 20/21 Nov 1999.*
- *East Timor vote for independence from Indonesia in August 1999.*
- *OIC Balkans 1990s*

If no temporal intervals can be found in the title, description, or narrative, the query is *implicitly temporal*. Examples of implicitly temporal queries found in the considered collections are:

- *What are the negative impacts of Argentina's policy of pegging their peso to the U.S. dollar?*
- *Czechoslovakia breakup*
- *What was the impact of the Exxon Valdez oil spill on the marine life and wildlife of the area?*

In this case, we extract time intervals from a fourth source: the top-3 pseudo-relevant documents of the collection, obtained using BM25 similarity. For all the above queries, we were able to find at least one timex in the top-3 retrieved

documents. On average, between 4 and 9 timexes (depending on the collection) can be found in the top-3 similar documents. We select the most occurring interval to form the query's temporal scope. We choose to only take one interval since all explicitly temporal queries across the test collections have a singleton temporal scope. Our focus is not on devising the best method to extract time features for implicitly temporal queries, but to show that even a simple method works well and that these implicit time features help our retrieval models improve the effectiveness of a purely textual ranking (see research question Q1). More sophisticated approaches, such as [13, 59], are orthogonal to the scope of this paper. Table 5 summarizes the number of explicitly and implicitly temporal queries in the five test collections.

		Temporal queries	
	Total queries	**Explicitly temporal queries**	**Implicitly temporal queries**
Novelty 2004	50	13 (26%)	37 (74%)
Robust 2004	250	13 (5%)	237 (95%)
Robust 2005	50	3 (6%)	47 (94%)
Temporalia	200	79 (40%)	121 (60%)
LMT-NYT	50	50 (100%)	–

Table 5. Number of explicitly and implicitly temporal queries in the five test collections.

8.1.4 Evaluation metrics. We evaluate each system via the following effectiveness metrics:

- *Precision at position k (Precision@k)*: For each query, the fraction of *relevant* documents among the top-k documents returned by the system.
- *Mean Average Precision (MAP)*: The *mean* of the *average precisions* among all queries in a test collection, where the *average precision* of a single query is the average of all precisions at each returned relevant document in the ranked list.

8.2 Results

8.2.1 Effects of the combination parameter α. Recall from Section 6 that all ranking models use a linear combination parameter, α, to balance purely textual and purely temporal scores into a final ranking score, where a higher α indicates a higher importance of the temporal component. In Figure 8, we illustrate the effects on Mean Average Precision of various choices of α (similar trends happen with Precision@k. From the figure, it is clear that the α maximizing the MAP is always greater than zero across all collections, meaning that combining a purely textual ranking with our temporal ranking is always better than using the textual ranking alone (see our research question Q2). It is also noteworthy that MAP is a concave function of α. This indicates that it is possible to use simple numerical optimizations, such as the *golden section search*, for identifying the best value for α (see our research question Q3). Note that the value for α can be tuned on a subset of the test collection, in our experiment 10% of the queries, and still be effective on unseen new queries, as shown in all of the presented experiments.

In the remainder of the experiments, we first identify the best α in the range $[0, 1]$, using the golden section search method, with tolerance set to 0.005, on a subset of the documents. We then report the evaluation measures (Precision@k and MAP) on a different subset of documents, and average the results obtained with 10-fold cross-validation.

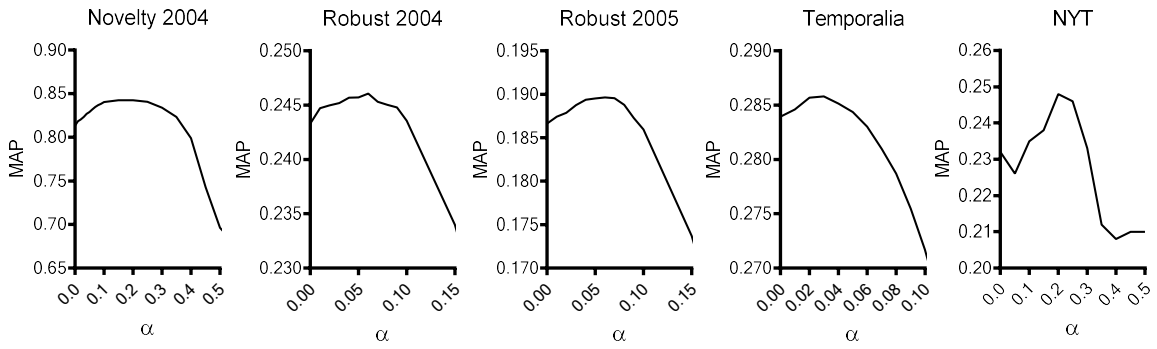


Fig. 8. Mean Average Precision as a function of the temporal weight α. The function is concave, maximum MAP is obtained with temporal weight always > 0. The results are shown using the distance δ^{min}_{covD}, with monthly granularity.

8.2.2 Comparison with the state of the art. In Sections 2 and 5, we presented related work on Temporal Information Retrieval and showed how those differ from the defined metric similarity. In this section, we empirically compare our temporal model with the state of the art temporal models, Language Model with Temporal Uncertainty (LMTU) [4]. We run the original LMTU code provided by its authors, combined with the Language Model Jelinek-Mercer (LMJM) [62] for the text similarity, on the same test collections we used for the evaluation of our model and on the test collection used in the original LMTU work. We compare the results with our model using the same text similarity (LMJM) linearly combined with the document coverage similarity, with cross-validated weights. In addition, we also provide the results of an experiment in which we replace our temporal similarity in their combination model to further demonstrate the capabilities of the metric models.

We evaluate the LMTU temporal baseline in comparison with the metric model on the three TREC collections. Moreover, we evaluate the models on the test collection used in the LMTU work [4], we named it LMT-NYT. We take into account the LMT-NYT collection with some reservations due to its pooling bias, because it does not satisfy the requirements of *depth-k pooling* [28]. Specifically, while TREC collections provide relevance judgement on a pool of results obtained from 100 and more different systems, taking the first 100 results (k=100) [58], the pooling in the LMTU test collection is made of only 4 very similar systems of which only the first 10 results (k=10) are taken for judgement. Despite the experimental framework in [4] is robust for evaluation of those models, the pooling design make this collection not suitable for comparison with other models.

To make a compared evaluation with the LMTU system, we replicate the settings used in the original LMTU work [4]. We use the same text similarity, the Language Model with Jelinek-Mercer smoothing, with the same smoothing parameter ($\lambda = 0.75$). We use granularity day for all the temporal similarities and the same temporal intervals for all the temporal models in comparison.

The provided results are obtained from the following models:

(1) The LMJM text-only baseline with smoothing parameter $\lambda = 0.75$.
(2) The LMTU system [4] using the most effective smoothing parameters ($\lambda = 0.75$, $\gamma = 0.5$).
(3) Our metric model combined with LMJM using a *multiplicative* combination, instead of the linear combination discussed in Section 6 (LMJM $\cdot\ \delta^{min}_{covD}$).
(4) Our metric model combined with LMJM using a linear combination, as described in Section 6, with 10-fold cross-validation for the tuning of α (LMJM + δ^{min}_{covD}).

We include model (3) in the experiment to provide a fairer comparison with LMTU [4], and to better explain its performance compared to our model (4). In fact, model (3) replaces our linear combination with a multiplicative combination, effectively *imitating* the combination model of LMTU, which is also multiplicative albeit in a probabilistic setting. Notice that a fifth model, where we instantiate LMTU using a linear combination instead of a multiplicative combination is not viable due to LMTU being a probabilistic model.

In Table 6 we show the evaluation of these four models, on the three test collection from TREC (Novelty 2004, Robust 2004, Robust 2005) and also on the test collection LMT-NYT, that has been created in [4] to specifically evaluate the LMTU system. Effectiveness is measured by precision of the top 10 results of each system, in order to replicate and compare the results in [4].

Table 6. Overall comparison Precision@10 between LMTU and Document Coverage distance on the 4 test collections. The parameters λ and γ are the smoothing parameters used in the LMTU system.

		Precision@10				
Model	**Settings**	**Novelty 2004**	**Robust 2004**	**Robust 2005**	**Temporalia**	**LMT-NYT**
LMJM (Text only)	λ=0.75	0.7760	0.4072	0.312	**0.2716**	0.36
LMJM · LMTU-EX	λ=0.75, γ=0.5	0.6740	0.1067	0.128	0.0261	**0.48**
LMJM · δ^{min}_{covD}	G=Day	0.6979	0.1331	0.232	0.0009	0.3375
LMJM + δ^{min}_{covD}	G=Day	**0.8159**	**0.4353**	**0.354**	**0.2716**	0.14

The results show several insights. First, our temporal similarities, both linear and multiplicative, give better results than LMTU across all TREC collections and Temporalia. The outcomes flip only on the LMT-NYT collection, which was constructed to originally evaluate LMTU in [4], with LMTU obtaining the best scores. Second, our model using a linear combination (LMJM + δ^{min}_{covD}) is always better than the text-only model (LMJM) across all collections except LMT-NYT. Third, our model using a multiplicative combination (LMJM · δ^{min}_{covD}) always obtains worse results compared to the linear combination (LMJM + δ^{min}_{covD}) across all collections, except for LMT-NYT, where it is better than linear. This shows that: (1) the LMT-NYT collection has an inherent bias in favor of multiplicative models (such as LMTU and the version of our model that uses a multiplicative combination); (2) in all other collections that do not have that bias, a linear model is always substantially better than a multiplicative model (see our research question Q4).

All the differences in results between Document Coverage and LMTU have been positively tested for statistical significance using the bootstrap method and t-test [48], taking into account the variability across queries. In other words, given any two similarities, we first produce the average precision for each query, then pairing for the paired significance tests is performed on this evaluation metric. We obtained a p-value < 0.01, although it is not possible to provide unbiased observations on the LMT-NYT collection because of the particularly small pooling of relevance assessments.

8.2.3 Metric distances and granularities. In Section 5 we presented different metric distances able to estimate the dissimilarity between two time intervals: Manhattan distance, Euclidean distance, Document Coverage distance and Query Coverage distance.

Moreover, thinner or coarser granularities of time can be used to represent intervals, a choice that clearly affects the similarity between two intervals. Consider, for example, the dates "2018-01-01" and "2019-01-01": they have a

distance of 1 year, 12 months, or 365 days, depending on the granularity applied to represent them (year, month, or day, respectively).

Using the 10-fold cross validation to select a fitting alpha parameter, we run all the distance-based similarities on four test collection: Novelty 2004, Robust 2004, Robust 2005 and Temporalia. In Table 7 and 8 we show how each distance-based similarity performs in terms of Mean Average Precision (MAP) and precision at the top 10 results (P@10), in the different collections and for the granularities day, month and year.

Table 7. Comparison between distance-based temporal similarities on day, month and year granularities using day, month or year granularity. In bold, the best score for each granularity column, bold and underlined is the best score for the whole collection. For all the collections and granularity choices, all the proposed models show higher values of Mean Average Precision.

	MAP											
	Novelty 2004			**Robust 2004**			**Robust 2005**			**Temporalia**		
Metric	G=D	G=M	G=Y	G=D	G=M	G=Y	G=D	G=M	G=Y	G=D	G=M	G=Y
BM25 (Text only)		0.8131			0.2433			0.1866			0.2696	
BM25 + δ^{min}_{man}	0.8465	0.8423	0.8262	0.2448	0.2459	0.245	0.1875	0.1893	0.1884	0.2704	0.2704	0.2701
BM25 + δ^{min}_{covD}	0.8454	0.8426	0.8251	0.2448	0.2462	0.2452	0.1877	**0.1898**	0.1888	0.2704	0.2703	0.2702
BM25 + δ^{min}_{covQ}	0.8349	0.8319	0.8238	0.2453	0.2462	0.2451	0.1879	**0.1898**	0.1891	**0.2707**	0.2706	0.2702
BM25 + δ^{min}_{eucl}	**0.8481**	0.8413	0.8251	0.2451	**0.2465**	0.2452	0.1877	0.1894	0.1889	0.2703	0.2704	0.2702
BM25 + δ^{min}_{mcovD}	0.8474	0.8428	0.8261	0.2448	0.2459	0.245	0.1875	0.1895	0.1886	0.2704	0.2703	0.2701
BM25 + δ^{min}_{mcovQ}	0.8472	0.8422	0.826	0.2448	0.2459	0.245	0.1875	0.1894	0.1886	0.2704	0.2704	0.2701
BM25 + δ^{min}_{ecovD}	0.8479	0.8427	0.8251	0.2448	0.2463	0.2452	0.1876	0.1902	0.1889	0.2703	0.2704	0.2702
BM25 + δ^{min}_{ecovQ}	0.8476	0.841	0.825	0.245	**0.2465**	0.2452	0.188	0.1888	0.1889	0.2704	0.2704	0.2702

Given the variance in the results, we cannot identify a single better distance. The different results for different collections suggests that a specific distance can be more suitable for a specific collection. While the Euclidean distance shows a majority of highest scores in different collections, the coverage distances carry slightly better results in Robust 2005 and, depending on the setting, in Novelty 2004 and Temporalia (see our research questions Q5 and Q7).

Looking at MAP and P@10 scores for each granularity, thinner granularities of one day and one month generally have the best scores in all the collections.

While the best choice of granularity and distance function varies across collections, all models are always at least as good as the textual baseline, and better in most cases, showing that our models are robust to imperfect settings.

In order to evaluate the significance of the improvements we run paired significance tests (Bootstrap and t-test) on the average precision obtained from the different models on each query, to take into account the variability across different queries [48]. The improvement obtained from all the proposed distance-based similarities with respect to the text baseline is always significant ($p\text{-}value < 0.05$).

8.2.4 Aggregations. In Section 5, we presented three different strategies to aggregate the temporal distances of a document: minimum, average and maximum of distances. In Figure 9, we show the results obtained on the four test collections using different aggregations. We measure the results as improvement percentage of the MAP and P@10 over the text baseline.

Table 8. Comparison between distance-based temporal similarities on day, month and year granularities using day, month or year granularity. Underlined are the best score for each granularity column, bold and underlined is the best score for the given collection. Precision results confirm that the proposed models in combination with the textual baseline always improve the retrieval effectiveness, although it is not possible to identify a clear winner for all collections.

	P@10											
	Novelty 2004			**Robust 2004**			**Robust 2005**			**Temporalia**		
Metric	G=D	G=M	G=Y	G=D	G=M	G=Y	G=D	G=M	G=Y	G=D	G=M	G=Y
BM25 (Text only)		0.816			0.4337			0.4			0.517	
BM25 + δ^{min}_{man}	0.85	0.854	0.838	0.4386	0.4373	0.4349	0.404	0.416	0.412	0.5215	0.5235	0.5195
BM25 + δ^{min}_{covD}	0.852	**0.856**	0.836	0.4369	0.4357	0.4337	0.406	0.416	0.408	0.5185	0.52	0.521
BM25 + δ^{min}_{covQ}	0.844	0.838	0.83	0.4365	0.4361	0.4325	**0.42**	0.412	0.418	0.5215	0.524	0.521
BM25 + δ^{min}_{eucl}	0.854	0.854	0.836	**0.4418**	0.4386	0.4337	0.402	0.418	0.41	0.521	**0.5245**	0.521
BM25 + δ^{min}_{mcovD}	0.85	0.852	0.836	0.4382	0.4378	0.4337	0.404	0.414	0.408	0.5215	0.521	0.5195
BM25 + δ^{min}_{mcovQ}	0.85	0.854	0.834	0.4382	0.4382	0.4337	0.404	0.414	0.408	0.5215	0.5235	0.5195
BM25 + δ^{min}_{ecovD}	0.852	0.854	0.836	0.4378	0.439	0.4337	0.402	0.416	0.41	0.5215	**0.5245**	0.521
BM25 + δ^{min}_{ecovQ}	0.854	0.852	0.836	0.4402	0.4382	0.4337	0.406	**0.42**	0.41	0.5205	0.523	0.521

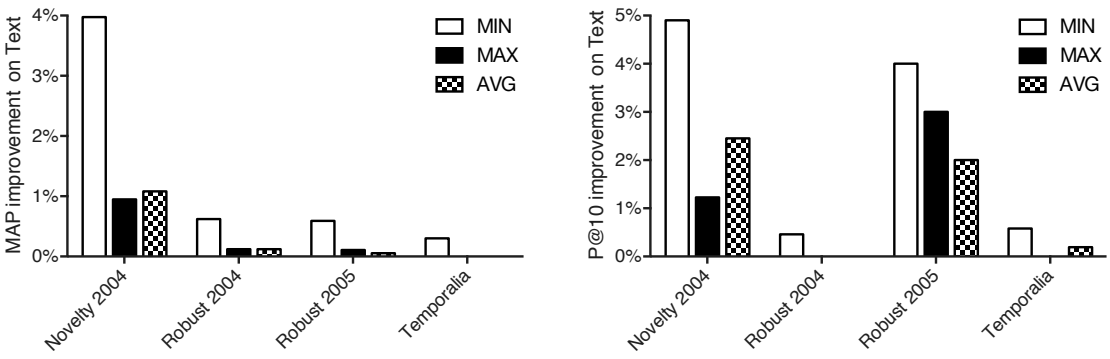


Fig. 9. MAP improvement on text baseline by distance aggregation. (`Distance=`δ^{agg}_{covD}`,Granularity=Month`).

Figure 9 shows that aggregating the distances by taking the minimum (MIN in the figure) distance produces the best results for all collection, both for MAP and P@10 measures. The minimum aggregation is by far the most relaxed aggregation because it ignores all the dissimilar intervals in a document and takes into consideration only the interval closest to the query. The average aggregation (AVG in the figure) is stricter because it considers all the intervals in the document. However, together with the maximum aggregation (MAX in the figure), which takes only the farther interval into consideration, AVG always performs worse than MIN. Moreover, maximum and average do not improve over text baselines in Temporalia, by MAP measure, and in Robust 2004, by P@10 (see our research question Q6).

8.2.5 Metrics and TBM25 Unigram model. In order to implement the TBM25 similarity that we defined in Section 5, we build a temporal representation of documents as sets of unigram intervals: every document is represented as the set of unique intervals that it contains, weighted with the frequency the intervals occur, following the Okapi BM25 weighting scheme [43]. In Table 9 and 10, we compare our unigram model, TBM25, against the two coverage distances that we defined. Both the coverage distances and the TBM25 similarity are combined with the BM25 similarity score. We show the results for month granularity as this is on average the most effective granularity.

Table 9. MAP comparison between Temporal BM25 (Unigram) and the coverage distances on the four test collections.

		MAP			
Model	**Settings**	**Novelty 2004**	**Robust 2004**	**Robust 2005**	**Temporalia**
BM25 (Text only)		0.8131	0.2433	0.1866	0.2696
BM25 + TBM25	G=Month	**0.8529**	0.2448	0.1897	0.2696
BM25 + δ^{min}_{covD}	G=Month	0.8426	**0.2462**	**0.1898**	**0.2703**
BM25 + δ^{min}_{covQ}	G=Month	0.8319	**0.2462**	**0.1898**	**0.2706**

Table 10. Precision of the first 10 results, comparison between Temporal BM25 (Unigram) and the coverage distances on the four test collections.

		P@10			
Model	**Settings**	**Novelty 2004**	**Robust 2004**	**Robust 2005**	**Temporalia**
BM25 (Text only)		0.836	0.4337	0.4	0.517
BM25 + TBM25	G=Month	**0.874**	0.4341	0.41	0.5195
BM25 + δ^{min}_{covD}	G=Month	0.856	0.4357	**0.416**	0.52
BM25 + δ^{min}_{covQ}	G=Month	0.838	**0.4361**	0.412	**0.524**

Results in Table 9 show that the TBM25 similarity that we defined is indeed a very good temporal similarity with a remarkable advantage over the MAP of the text baseline. Moreover, in the single case of the Novelty 2004 collection, the TBM25 similarity exhibit the best result among all the proposed similarities. However in bigger collections the coverage distances are still slightly above the effectiveness of TBM25. The same result can be observed in Table 10, with the precision of the top 10 documents of TBM25 being better in the TREC Novelty 2004 collection. These results suggests that considering features such as the cardinality of temporal scopes and the frequency of time intervals in documents and collection improves the effectiveness even with units different than keywords (see our research question Q7).

Summary of results. In summary, to answer our initial research questions, our results show that:

(1) Despite previous belief, most queries exhibit some temporal intent, even though they do not explicitly contain temporal expressions or they were not generated with explicit temporal intent.
(2) The proposed temporal similarities, when combined with a textual similarity, significantly improve the retrieval effectiveness in comparison to only using the textual similarity alone.

(3) On collections of medium size (more than 500,000 documents), the best combination weight for temporal and textual similarities (α) is between 2% and 6%. All the presented results use a weight tuned on a separate 10% of queries through numerical optimization, showing that it is possible to learn this parameter on a small training set while being effective also on unseen queries.
(4) The proposed distance-based similarities provide significant improvement over state-of-the-art temporal similarities for standard IR test collections.
(5) Results shows that a monthly or finer granularity is needed to obtain the highest improvement from temporal similarities, independently of the choice of alpha.
(6) The minimum (i.e., most conservative) distance between temporal scopes always provides the best results.
(7) Among the proposed temporal similarities, the most effective model varies across different test collections. However, our models are always at least as good as the textual baseline, and better in most cases, showing that our models are robust to imperfect settings.

9 CONCLUSION AND FUTURE WORK

In this paper, we considered the temporal dimension of documents and queries in order to improve the effectiveness of retrieval tasks. After defining the temporal scope of documents and queries as sets of time intervals, we proposed a variety of ad hoc distances for time intervals that model the temporal similarity between documents and queries for different scenarios. From a qualitative perspective, we showed how the proposed distance-based similarities capture more temporal relations, such as containment and distance, than other approaches in literature. We also proposed a simpler model by specialising the well-known BM25 similarity on the time intervals extracted from text, treating each time interval as a single text token. We addressed the problem of combining the temporal similarity score with the traditional text similarity score. Finally, we provided extensive experimental results on five different collection, showing that combining the text similarity with the proposed temporal similarities yields a significant improvement in effectiveness. The experimental evaluation showed also better results on general-purpose retrieval collections when compared with the state-of-the-art time-aware retrieval model.

This study confirms the importance of taking into account the temporal dimension of documents and queries, obtaining improved effectiveness even when there is no explicit time mentioned in the query. In particular, it highlights how representing time intervals and modelling temporal similarity within a proper space, instead of relying on keyword-oriented representations and models, allows to capture temporal relations such as distance and containment. We also found out that, while the proposed time distances perform better than text-only similarities and time-aware models, there is no clear winner among the proposed similarities for all the test collections, confirming that a temporal distance can be more suitable than others depending on the specific query and underlying document corpus.

This work paves the way for many important future research directions. First, future work needs to address how to best *tune* the model, selecting the best distance, aggregate function, and granularity on a query-by-query basis and by analyzing the collection of documents. For example, further study is needed to find how to associate the query's properties, such as the width of the query's temporal interval, with the selection of the right metric distance. Second, the *speed performance* of the proposed model needs to be addressed as well. The metric representation of time in documents and queries cannot not rely on bag-of-words evaluation strategies or data structures such as inverted indices and joins of posting lists. We carried out our extensive offline evaluations using a document at a time strategy, but this strategy is not efficient for large collections that include a large amount of temporal information. Building an interactive system needs new efficient query evaluation methods specifically tailored for our TMSM models. We plan to investigate index

and nearest-neighbours techniques for our models, as well as 2-step re-ranking strategies that can exploit fast text retrieval and then compute temporal similarity only on a subset of retrieved documents.

REFERENCES

[1] Ablimit Aji, Yu Wang, Eugene Agichtein, and Evgeniy Gabrilovich. 2010. Using the past to score the present: Extending term weighting models through revision history analysis. In *Proceedings of the 19th ACM international conference on Information and knowledge management*. ACM, 629–638.

[2] Omar Alonso, Michael Gertz, and Ricardo Baeza-Yates. 2007. On the value of temporal information in information retrieval. *ACM SIGIR Forum* 41, 2 (Dec. 2007), 35–41.

[3] Omar Rogelio Alonso. 2008. *Temporal Information Retrieval*. Ph.D. Dissertation. Davis, CA, USA. Advisor(s) Gertz, Michael. AAI3336211.

[4] Klaus Berberich, Srikanta Bedathur, Omar Alonso, and Gerhard Weikum. 2010. A Language Modeling Approach for Temporal Information Needs. In *Advances in Information Retrieval*. Springer Berlin Heidelberg, Berlin, Heidelberg, 13–25.

[5] Klaus Berberich, Michalis Vazirgiannis, and Gerhard Weikum. 2005. Time-aware authority ranking. *Internet Mathematics* 2, 3 (2005), 301–332.

[6] Andrzej Białecki, Robert Muir, Grant Ingersoll, and Lucid Imagination. 2012. Apache lucene 4. In *SIGIR 2012 workshop on open source information retrieval*. 17. https://lucene.apache.org.

[7] Paul Black. 2006. Manhattan Distance. Dictionary of Algorithms and Data Structures. *US National Institute of Standards and Technology. http://www.nist.gov/dads/HTML/manhattanDistance. html. Accessed* 31 (2006).

[8] Matteo Brucato, Leon Derczynski, Hector Llorens, Kalina Bontcheva, and Christian S Jensen. 2013. Recognising and interpreting named temporal expressions. In *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*. 113–121.

[9] Matteo Brucato and Danilo Montesi. 2014. Metric Spaces for Temporal Information Retrieval. In *Advances in Information Retrieval*. Lecture Notes in Computer Science, Vol. 8416. Springer International Publishing, 385–397.

[10] Ricardo Campos, Gaël Dias, Alípio Jorge, and Célia Nunes. 2016. GTE-Rank: A time-aware search engine to answer time-sensitive queries. *Information Processing & Management* 52, 2 (2016), 273–298.

[11] Ricardo Campos, Gaël Dias, Alípio M. Jorge, and Adam Jatowt. 2014a. Survey of Temporal Information Retrieval and Related Applications. *ACM Comput. Surv.* 47, 2, Article 15 (aug 2014), 41 pages. DOI: http://dx.doi.org/10.1145/2619088

[12] Ricardo Campos, Gaël Dias, Alípio Mário Jorge, and Célia Nunes. 2014b. GTE-Rank: Searching for Implicit Temporal Query Results. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management (CIKM '14)*. ACM, New York, NY, USA, 2081–2083.

[13] Ricardo Campos, Gaël Dias, Alípio Mário Jorge, and Célia Nunes. 2017. Identifying top relevant dates for implicit time sensitive queries. *Information Retrieval Journal* 20, 4 (2017), 363–398.

[14] Wisam Dakka, Luis Gravano, and Panagiotis Ipeirotis. 2012. Answering general time-sensitive queries. *IEEE Transactions on Knowledge and Data Engineering* 24, 2 (2012), 220–235.

[15] Leon Derczynski, Jannik Strötgen, Ricardo Campos, and Omar Alonso. 2015. Time and information retrieval: Introduction to the special issue. *Information Processing Management* 51, 6 (2015), 786–790. DOI: http://dx.doi.org/https://doi.org/10.1016/j.ipm.2015.05.002

[16] Anlei Dong, Yi Chang, Zhaohui Zheng, Gilad Mishne, Jing Bai, Ruiqiang Zhang, Karolina Buchner, Ciya Liao, and Fernando Diaz. 2010. Towards Recency Ranking in Web Search. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining (WSDM '10)*. ACM, New York, NY, USA, 11–20.

[17] Miriam Fernández, David Vallet, and Pablo Castells. 2006. Probabilistic score normalization for rank aggregation. In *Advances in Information Retrieval*. Springer, 553–556.

[18] Miriam Fernández, Iván Cantador, Vanesa López, David Vallet, Pablo Castells, and Enrico Motta. 2011. Semantically enhanced Information Retrieval: An ontology-based approach. *Web Semantics: Science, Services and Agents on the World Wide Web* 9, 4 (2011), 434 – 452. DOI: http://dx.doi.org/https://doi.org/10.1016/j.websem.2010.11.003 JWS special issue on Semantic Search.

[19] Michele Filannino and Goran Nenadic. 2014. Using Machine Learning to Predict Temporal Orientation of Search Engines' Queries in the Temporalia Challenge.. In *NTCIR*.

[20] Sharon Flank. 1998. A Layered Approach to NLP-based Information Retrieval. In *Proceedings of the 17th International Conference on Computational Linguistics - Volume 1 (COLING '98)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 397–403.

[21] Martin Franz, Abraham Ittycheriah, J Scott McCarley, and Todd Ward. 2001. First story detection: Combining similarity and novelty based approaches. In *Topic Detection and Tracking Workshop Report*. 193–206.

[22] George W. Furnas, Thomas K. Landauer, Louis M. Gomez, and Susan T. Dumais. 1987. The vocabulary problem in human-system communication. *Commun. ACM* 30, 11 (1987), 964–971.

[23] Mohammed Hasanuzzaman, Gaël Dias, and Stéphane Ferrari. 2014. Hultech at the NTCIR-11 temporalia task: ensemble learning for temporal query intent classification. In *The 11th NTCIR Conference on Evaluation of Information Access Technologies*. p–478.

[24] Adam Jatowt, Yukiko Kawai, and Katsumi Tanaka. 2005. Temporal ranking of search engine results. In *Web Information Systems Engineering–WISE 2005*. Springer, 43–52.

[25] Peiquan Jin, Jianlong Lian, Xujian Zhao, and Shouhong Wan. 2008. *TISE: A Temporal Search Engine for Web Contents*. Vol. 3. IEEE.

[26] Hideo Joho, Adam Jatowt, Roi Blanco, Hajime Naka, and Shuhei Yamamoto. 2014. Overview of NTCIR-11 Temporal Information Access (Temporalia)

Task.. In *NTCIR*. https://sites.google.com/site/ntcirtemporalia/.

[27] Rosie Jones and Fernando Diaz. 2007. Temporal Profiles of Queries. *ACM Trans. Inf. Syst.* 25, 3, Article 14 (July 2007).

[28] Sparck Jones. 1975. Report on the need for and provision of an "ideal" information retrieval test collection. (1975).

[29] Nattiya Kanhabua and Kjetil Nørvåg. 2010. Determining Time of Queries for Re-ranking Search Results. In *Advances in Databases and Information Systems*. Springer Berlin Heidelberg, Berlin, Heidelberg, 261–272.

[30] Nattiya Kanhabua and Kjetil Nørvåg. 2012. Learning to rank search results for time-sensitive queries. In *the 21st ACM international conference*. ACM Press, New York, New York, USA, 2463–2466.

[31] Ali Khodaei, Cyrus Shahabi, and Amir Khodaei. 2012. Temporal-Textual Retrieval: Time and Keyword Search in Web Documents. *INTERNATIONAL JOURNAL OF NEXT-GENERATION COMPUTING* 3, 3 (2012).

[32] Anagha Kulkarni, Jaime Teevan, Krysta M. Svore, and Susan T. Dumais. 2011. Understanding Temporal Query Dynamics. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining (WSDM '11)*. ACM, New York, NY, USA, 167–176.

[33] Xiaoyan Li and W. Bruce Croft. 2003. Time-based Language Models. In *Proceedings of the Twelfth International Conference on Information and Knowledge Management (CIKM '03)*. ACM, New York, NY, USA, 469–475.

[34] Hector Llorens, Leon Derczynski, Robert J Gaizauskas, and Estela Saquete. 2012. TIMEN: An Open Temporal Expression Normalisation Resource.. In *LREC*. 3044–3051.

[35] Raghavan Manmatha, T Rath, and Fangfang Feng. 2001. Modeling score distributions for combining the outputs of search engines. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 267–275.

[36] Sérgio Nunes, Cristina Ribeiro, and Gabriel David. 2007. Using Neighbors to Date Web Documents. In *Proceedings of the 9th Annual ACM International Workshop on Web Information and Data Management (WIDM '07)*. ACM, New York, NY, USA, 129–136.

[37] Sérgio Nunes, Cristina Ribeiro, and Gabriel David. 2008. Use of temporal expressions in web search. In *European Conference on Information Retrieval*. Springer, 580–584.

[38] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank citation ranking: Bringing order to the web.* Technical Report. Stanford InfoLab.

[39] Maria-Hendrike Peetz and Maarten De Rijke. 2013. Cognitive temporal document priors. In *Advances in Information Retrieval*. Springer, 318–330.

[40] Jukka Perkiö, Wray Buntine, and Henry Tirri. 2005. A temporally adaptive content-based relevance ranking algorithm. In *the 28th annual international ACM SIGIR conference*. ACM Press, New York, New York, USA, 647–648.

[41] James Pustejovsky, José Castaño, Robert Ingria, Roser Saurí, Robert Gaizauskas, Andrea Setzer, and Graham Katz. 2003. Timeml: Robust specification of event and temporal expressions in text. In *in Fifth International Workshop on Computational Semantics (IWCS-5*.

[42] Stephen Robertson. 2007. *On score distributions and relevance*. Springer.

[43] Stephen Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, and others. 1995. Okapi at TREC-3. *NIST SPECIAL PUBLICATION SP* 109 (1995), 109.

[44] Evan Sandhaus. 2008. The New York Times annotated corpus. *Linguistic Data Consortium, Philadelphia* 6, 12 (2008), e26752. https://catalog.ldc.upenn.edu/LDC2008T19.

[45] Nobuyoshi Sato, Minoru Uehara, and Yoshifumi Sakai. 2003. Temporal information retrieval in cooperative search engine. In *Database and Expert Systems Applications, 2003. Proceedings. 14th International Workshop on*. IEEE, 215–220.

[46] Frank Schilder and Christopher Habel. 2001. From Temporal Expressions to Temporal Information: Semantic Tagging of News Messages. In *Proceedings of the Workshop on Temporal and Spatial Information Processing - Volume 13 (TASIP '01)*. Association for Computational Linguistics, Stroudsburg, PA, USA, Article 9, 8 pages.

[47] Roger N Shepard. 1987. Toward a universal law of generalization for psychological science. *Science* 237, 4820 (1987), 1317–1323.

[48] Mark D Smucker, James Allan, and Ben Carterette. 2007. A comparison of statistical significance tests for information retrieval evaluation. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. ACM, 623–632.

[49] Ian Soboroff and Donna Harman. 2005. Novelty detection: the TREC experience. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 105–112. http://trec.nist.gov/data/t13_novelty.html.

[50] Alexander Strehl, Joydeep Ghosh, and Raymond Mooney. 2000. Impact of similarity measures on web-page clustering. In *Workshop on artificial intelligence for web search (AAAI 2000)*, Vol. 58. 64.

[51] 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*. Association for Computational Linguistics, 321–324.

[52] Jaime Teevan. 2005. The Re: Search Engine: Helping people return to information on the Web. In *Proceedings of SIGIR*, Vol. 5.

[53] Naushad UzZaman, Hector Llorens, James Allen, Leon Derczynski, Marc Verhagen, and James Pustejovsky. 2012. Tempeval-3: Evaluating events, time expressions, and temporal relations. *arXiv preprint arXiv:1206.5333* (2012).

[54] Marc Verhagen, Inderjeet Mani, Roser Sauri, Robert Knippen, Seok Bae Jang, Jessica Littman, Anna Rumshisky, John Phillips, and James Pustejovsky. 2005. Automating Temporal Annotation with TARSQI. In *Proceedings of the ACL 2005 on Interactive Poster and Demonstration Sessions (ACLdemo '05)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 81–84.

[55] Marc Verhagen, Roser Sauri, Tommaso Caselli, and James Pustejovsky. 2010. SemEval-2010 task 13: TempEval-2. In *Proceedings of the 5th international workshop on semantic evaluation*. Association for Computational Linguistics, 57–62.

[56] Ellen M Voorhees. 2005. The TREC robust retrieval track. In *ACM SIGIR Forum*, Vol. 39. ACM, 11–20. http://trec.nist.gov/data/t13_robust.html.

[57] Ellen M Voorhees. 2006. The TREC 2005 robust track. In *ACM SIGIR Forum*, Vol. 40. ACM, 41–48. http://trec.nist.gov/data/t14_robust.html.

[58] Ellen M. Voorhees and Donna Harman. 1998. The Text Retrieval Conferences (TRECS). In *Proceedings of a Workshop on Held at Baltimore, Maryland: October 13-15, 1998 (TIPSTER '98)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 241–273.

[59] Jingjing Wang and Shengli Wu. 2017. Information Retrieval with Implicitly Temporal Queries. In *International Conference on Intelligent Data Engineering and Automated Learning*. Springer, 103–111.

[60] Misha Wolf and Charles Wicksteed. 1998. Date and time formats. *W3C* (1998). https://www.w3.org/TR/NOTE-datetime

[61] Philip S Yu, Xin Li, and Bing Liu. 2004. On the temporal dimension of search. In *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*. ACM, 448–449.

[62] Chengxiang Zhai and John Lafferty. 2004. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems (TOIS)* 22, 2 (2004), 179–214.