Event Occurrence Date Estimation based on Multivariate Time Series Analysis over Temporal Document Collections

Jiexin Wang
Kyoto University
Kyoto, Japan
wang.jiexin.83m@st.kyoto-u.ac.jp

Adam Jatowt
University of Innsbruck
Innsbruck, Austria
adam.jatowt@uibk.ac.at

Masatoshi Yoshikawa
Kyoto University
Kyoto, Japan
yoshikawa@i.kyoto-u.ac.jp

ABSTRACT

Real world events are quite often mentioned in texts. Estimating the occurrence time of event mentions has many applications in IR, QA, general document understanding and downstream NLP tasks. In this paper we propose an approach to temporal profiling of event mentions in text. Our method utilizes a news article archival collection for collecting temporal as well as textual information containing contemporary and retrospective event references. As we demonstrate in our experiments, the recent method which relies on secondary data sources like Wikipedia is insufficient to correctly estimate the event time, especially, for minor or less well-known events that happened in the past. Our method then harnesses news article archives to effectively infer the occurrence time of past events, and is able to estimate the time at different temporal granularities (e.g., day, week, month, or year). As evidenced through extensive experiments, the proposed model outperforms the existing methods by a large margin at all granularities. We also demonstrate that our approach helps to answer arbitrary questions about past events, when incorporated into a QA framework operating over news article archives.

CCS CONCEPTS

• **Information systems** → **Document representation**; • **Computing methodologies** → **Information extraction**;

KEYWORDS

temporal event profiling; event time estimation; multivariate time series analysis; news archives; transformer

ACM Reference Format:
Jiexin Wang, Adam Jatowt, and Masatoshi Yoshikawa. 2021. Event Occurrence Date Estimation based on Multivariate Time Series Analysis over Temporal Document Collections. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21), July 11–15, 2021, Virtual Event, Canada.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3404835.3462885

SIGIR '21, July 11–15, 2021, Virtual Event, Canada

ACM ISBN 978-1-4503-8037-9/21/07…$15.00
https://doi.org/10.1145/3404835.3462885

1 INTRODUCTION

News articles are one of the most commonly read types of documents online. Time can be leveraged to organize and search relevant information in news texts, aiding in exploration of the causalities, developments, and effects of the events, etc. Event occurrence time, indicating when an event took place, constitutes then one of the most significant type of information about the event. In recent years, utilizing event-related information in IR and NLP tasks has attracted increasing attention. Event time information in particular has been exploited in various diverse tasks, such as search results diversification [3, 13, 38], multi-document summarization [31], timeline construction [11, 26, 39, 49], named entity disambiguation [1] and historical event ordering [14].

This paper addresses the problem of event occurrence time estimation defined as follows: *given a short description of an event and a chosen temporal granularity, the task is to estimate event's occurrence time at the specified granularity using a temporal document collection as the underlying knowledge source.* For example, given the event-describing sentence "A bombing of a Superferry by Abu Sayyaf in the Philippines killed 116" and month granularity, an effective model should infer its occurrence time, which is "2004-02" based on querying a relevant news article archive. Note that the task could be also regarded as a variant of question answering with a particular objective to answer questions about when the events occurred. Though we emphasize that a successful model should infer the correct time even if it is not explicitly mentioned in any available document.

In this paper we propose a model called TEP-Trans (Temporal Event Profiling Transformer-based model) which is a Transformer-based neural network to approach our task, by exploiting both temporal and textual information from different angles, represented by multivariate time series. We are the first to address the time estimation task by applying the ideas of multivariate time series analysis and the Transformer approach [43], which is a deep learning architecture that leverages attention mechanism and has been proved to be especially effective in natural language processing. We note that the performance of the existing methods is unsatisfactory for the temporal event profiling task, especially at fine-grained granularities (e.g., day, week), as they are either statistical approaches [12, 17, 22], or are designed over synchronic document collections (e.g., Wikipedia) [9, 14] that are incapable of utilizing document timestamp information in contrast to methods based on temporal collections of news articles. We then utilize data directly from temporal document collection and propose a neural network based solution for extracting correct temporal signals.

In the experiments, we use the New York Times Annotated Corpus (NYT corpus) [36] as the underlying data source, which contains

Table 1: Examples of event descriptions and their occurrence time in our dataset

No.	Description	Time
1	An official news agency in the Soviet Union reports the landing of a UFO in Voronezh.	1989-10-09
2	Antonov-26 plane crashes at Gyumri,Armenia, 36 killed.	1993-12-26
3	FBI agent Earl Pitts pleads guilty to selling secrets to Russia.	1997-02-28
4	President of Pakistan Pervez Musharaf narrowly escaped an assassination attempt.	2003-12-14
5	George Bell is 1st Blue Jay ever to win the AL MVP.	1987-11-17
6	Toru Takemitsu's "Archipelago" premieres in Aldeburgh England.	1993-06-18
7	Will Clark, National League's Most Valuable Player signs a $15 million four-year contract with San Francisco Giants.	1990-01-22

over 1.8 million news articles published between January 1, 1987 and June 19, 2007. We construct a large dataset containing 22,398 short event descriptions, paired with their occurrence dates which fall into the time frame of the NYT corpus. Table 1 presents example records in our dataset. Note that some events in our dataset, especially the less well-known ones, are not mentioned in Wikipedia or are only reported with temporal information of crude granularity (e.g., year). For example, Wikipedia does not contain any information about event #6 and event #7 in Table 1, and it records only year information of event #5. This necessitates using other resources such as large scale news archives in order to enable temporal event profiling of lesser-known or minor events, as well as to assure providing fine granularity temporal information. The experimental results show that our proposed model outperforms other models by a large margin at all temporal granularities.

To sum up, we make the following contributions in this work:

- We propose a novel TEP-Trans model based on Transformer architecture and multivariate time series analysis which is able to estimate the event occurrence time at different temporal granularities based on a long-term news archive as the underlying knowledge source[1].
- We construct a large dataset of past events and perform extensive experiments to prove the effectiveness of our model.
- We show that our model can be successfully applied on the downstream IR/NLP tasks such as open question answering to further improve their performance.

The remainder of this paper is structured as follows. The next section overviews the related work. In Section 3, we introduce our method. Section 4 describes experimental settings, while Section 5 provides experimental results. In Section 6 we demonstrate how the proposed approach can improve other tasks on the example of Question Answering in news corpora. Finally, we conclude the paper and outline our future work in Section 7.

2 RELATED WORK

2.1 Document Dating

Document dating or document age prediction is relevant to our task. One of the first automatic document dating studies is the work of Jong et al. [18]. They use unigram language models for specific time periods and score articles with log-likelihood ratio scores. That approach was later improved by Kanhabua and Nørvåg [20] by expanding its unigrams with POS tags, collocations, and tf-idf scores. Recent method proposed by Vashishth et al. [42] introduce Graph Convolutional Network (GCN) which jointly exploits syntactic and temporal graph structures of document to solve the problem.

2.2 Document Focus Time Estimation

Document focus time estimation [15] also relates to our research problem as its objective is to determine the temporal distribution reflecting the time periods the content of a given document treats about. Note that it is a different problem from the above-discussed document dating task. For example, one could write an article about 9/11 terrorist attacks in 2021 in which case the creation date would be 2021 while the focus time would be September 11, 2001. The authors of [15] propose a graph-based approach that constructs a date-term association graph based on the co-occurrence of words and temporal expressions, and identify discriminative associations which are then used to estimate the focus time. Shrivastava et al. [37] also introduce a graph-based method but treat documents and years as nodes which are connected by intermediate related Wikipedia concepts. They leverage the temporal relations between the concepts present in the text to estimate the document focus time. The shortcoming here is that documents may not always contain temporal expressions. Unlike the two above-mentioned tasks, the event occurrence time estimation does not aim to predict the publication date of text, it focuses strictly on events (rather than states), as well as it has different input which is not a document but a short event mention.

2.3 Query Temporal Profiling

Another relevant research problem is the task of temporal query profiling aiming to temporally disambiguate queries (e.g., queries about past, future, present or queries that are temporally neutral) as well as identify the time of their interest. This task focuses on short queries rather than event descriptions (e.g. "Hurricane Katrina"). Kanhabua and Nørvåg [22] introduce three different methods to identify the time of interest of queries and exploit this information for re-ranking the retrieved results. Their best-performing method uses only the timestamps of the top k retrieved documents as the query time. Thus the query time contains more than one time point when the timestamps of top k documents are different and the approach cannot determine which one is correct. Methods proposed by Dakka et al. [8], Jones and Diaz [17] also utilize timestamp information and identify query time by analyzing distribution of retrieved documents over time. Unlike these methods, Gupta and Berberich [12] take both timestamp information and temporal expressions from the content into account, and employ a probabilistic approach for the selection of suitable documents for a given query to subsequently generate a time interval from the temporal information. Differently to our approach, the authors mainly focus on the temporal expressions in the content and utilize the timestamp information only as additional temporal information of the content.

2.4 Event Occurrence Estimation

Other related works propose different ways to estimate the occurrence time of a given short event description [9, 14, 32]. Das et al. [9] introduce event-based time vector by integrating word

[1]The code and the dataset are available at https://github.com/WangJiexin/Temporal-Event-Profiling.

vectors and global time vector, and estimate the occurrence time by calculating the cosine similarity between event-describing sentences and event-based time vectors corresponding to temporal expression. Morbidoni et al. [32] utilize Wikipedia as well as the external knowledge base - DBpedia, and estimate the occurrence time by leveraging linked entities' centered representation of sentences and temporal information. Honovich et al. [14] propose two methods to tackle the task where the best one is realized by first extracting relevant sentences from the Wikipedia, and then using LSTM with attention mechanism to compute the encodings of event text and extracted sentences, and finally using an MLP to estimate the occurrence time that takes the concatenated encodings as input. Nonetheless, neither of these three methods is designed to work over primary document collections such as news archives, making them incapable of utilizing temporal information such as document timestamps. Although knowledge bases and Wikipedia contain abundant information on the major things from the past, they cannot provide information on numerous minor events that took place in the history. Finally, those methods work on rather coarse level granularity predicting only year information of the event occurrence time.

In comparison to the existing methods, our proposed model is designed over long-term news article collections. We leverage the novel Transformer architecture [43] and we let it utilize both temporal information and textual information embedded in documents. Our model can infer the event occurrence time at different temporal granularities. We also construct a large dataset for training the proposed model and release it to the research community. Event occurrence time estimation constitutes a significant building block for many downstream tasks (e.g. temporal information retrieval [2, 4], search result diversification [3, 13, 38], etc.), and might even serve as a fallback of question answering when the answer of the question about event date is not explicitly given in the text. Building such a model that can further help to make better use of the past news articles and satisfy different user information needs is of great importance, due to the continuous growth of document archives containing primary sources about the past.

3 METHOD

As already mentioned, the task is to estimate the event occurrence time based on an underlying news archive. For each event description, our approach first retrieves the relevant news articles, and then uses both their temporal and textual information. The temporal and textual signals are represented by four univariate time series, the lengths of which are equal to the length of the time frame of the used temporal document collection. These four time series are then aggregated to form a multivariate time series to be utilized as an input by the proposed TEP-Trans model for predicting the event's occurrence time. The notations used to explain our approach are listed in Table 2. Below we describe the steps of our method.

3.1 Retrieving Relevant News Articles

The first step is to identify keywords for each event description e and use them to retrieve relevant news articles from the news article archive D. We choose Yake![2] [5] as our keyword extraction method, which is a state-of-the-art unsupervised approach that rests on text statistical features extracted from single documents to select the most important keywords. Next, the query, which is composed of the extracted keywords, is sent to the ElasticSearch[3] installation which finally returns the top k relevant documents ranked by BM25.

Table 2: List of notations

Notations	Descriptions
e	A given event description
d, D	A news article and the underlying news archive
l	Length of the time series
$t_{pub}(d)$	Timestamp, i.e., publication date of d
$BM25(d), Rel(d)$	The BM25 and relevance score of d
$top(k)$	The set of top k relevant articles
$T(top(k))$	The set of extracted time intervals of top k articles
$S(top(k))$	The set of extracted sentences which contain extracted time intervals of top k articles
$\boldsymbol{e}, \boldsymbol{s}$	The encodings of e and a sentence s
$Sim(e, s)$	The similarity between e and a sentence s
$\mathbf{X}^{pub}_{temp}, \mathbf{X}^{cont}_{temp}$	Time series from temporal signals (publication and content)
$\mathbf{X}^{doc}_{text}, \mathbf{X}^{sent}_{text}$	Time series from textual information (document and sentence)
$\boldsymbol{\chi}$	The multivariate time series

3.2 Obtaining Time Series from Temporal Information

The second step is to extract the temporal information from timestamp and from the content of each retrieved document d, which are then aggregated and utilized to construct two univariate time series of length l:

$$\mathbf{X}^{pub}_{temp} = \left\{X^{pub}_{temp,1}, X^{pub}_{temp,2}, ..., X^{pub}_{temp,l}\right\} \quad (1)$$

$$\mathbf{X}^{cont}_{temp} = \left\{X^{cont}_{temp,1}, X^{cont}_{temp,2}, ..., X^{cont}_{temp,l}\right\} \quad (2)$$

$\mathbf{X}^{pub}_{temp}$ denotes the publication date time series and $\mathbf{X}^{cont}_{temp}$ denotes the content date time series. As previously mentioned, l equals to the length of the time frame of the news archive D, and its value naturally depends on the specified temporal granularity. In the experiments, we use the NYT corpus as the news archive, which contains articles published from January 1, 1987 to June 19, 2007. When setting the month granularity, l equals to 246 time units, corresponding to the number of all months in the NYT corpus. For the case of the week granularity, l amounts to 1,069 units (weeks). Similarly, at year and day granularities, l equals to 21 units (years) and 7,475 units (days), respectively. For ease of exposition, we will introduce our approach using month granularity in the remainder of this section.

Hence, the time unit i of the time series refers to the i-th month of the time frame of D. For example, $X^{pub}_{temp,1}$ represents the value of time series $\mathbf{X}^{pub}_{temp}$ at "January 1987", which is the first month of the NYT corpus. Below we discuss how to generate the above-mentioned two univariate time series.

3.2.1 Publication date time series. Based on the timestamps of the top k retrieved news articles, the publication date time series X^{pub}_{temp} is created by counting the number of relevant documents published at each time unit i, denoted as $X^{pub}_{temp,i}$:

[2]Yake! is available in the PKE tookit: https://github.com/boudinfl/pke

[3]https://www.elastic.co/

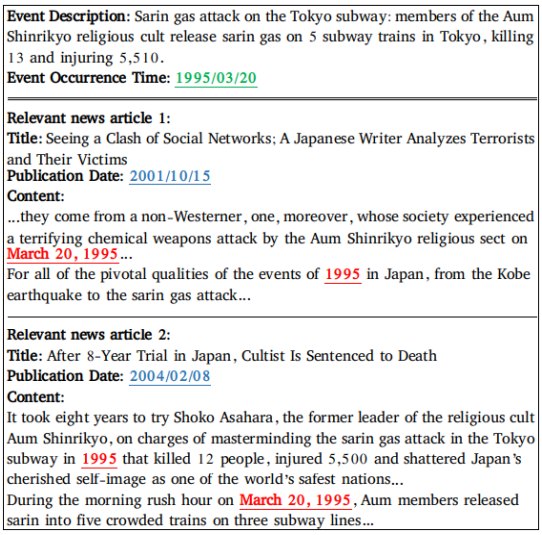

Figure 1: The examples of news articles (middle and bottom cell) that retrospectively refer to the target event (the description of this event is shown in the top cell).

$$X^{pub}_{temp,i} = \sum_{\substack{d \in top(k) \\ s.t.\ t_{pub}(d) = time\ i}} 1 \tag{3}$$

X^{pub}_{temp} indicates the distribution of the top k relevant news articles over time. Previous studies of query temporal profiling [8, 17, 22], which focus on identifying time of interest of queries, show that this distribution can reflect useful information regarding temporal characteristics of events.

3.2.2 Content date time series. The extraction of content temporal information and the calculation of content date time series $\mathbf{X}^{cont}_{temp}$ are slightly more complex. We utilize this information as some news articles, like ones published after the event time, may still retrospectively relate to the event, providing useful information. Such news articles may be even published long time after the target event, and focus on other similar events or on the subsequent development or effect of the target event. For example, as we can see in Figure 1, the two top-relevant news articles retrieved from the NYT collection provide important extra details on the target event (the event is described at the top of the figure). More importantly, they also mention the correct event occurrence time despite having been published six and nine years after the event, respectively. Thus, as we can see based on these examples, the temporal information embedded in document content can be useful for our task.

To utilize the content temporal information, we first use SUTime [6], a popular tool for recognizing temporal expressions, to identify and extract sentences containing temporal expressions from the top k relevant documents. Then, we collect all the extracted temporal expressions and map them to the time interval with the "start" and "end" information. For example, at month granularity, "in May 1990" is mapped to ('1990-05', '1990-05'), and "from 1998 to 2002" is mapped to ('1998-01', '2002-12').[4] More fine-grained time expressions such as "March 5, 2005" and "June 14, 2001 to October 10, 2001" are mapped to ('2005-05', '2005-05') and to ('2001-06', '2001-10'), respectively, when assuming monthly granularity of time series to be constructed. For a temporal expression whose one boundary of the interval cannot be determined, we use the start or end date of the document collection to replace the missing "start" or "end" information. For example, "after March 2000" is normalized to ('2000-03', '2007-06') and "before October 1999" is converted to ('1987-01', '1999-10'). Finally, we retain those time intervals that fall into the time frame of the news archive[5]. We represent the set of such time expressions as $T(top(k))$ and the set of their corresponding sentences as $S(top(k))$, to be used later.

The calculation of the content date time series at time unit i, denoted as $X^{cont}_{temp,i}$, is as follows:

$$X^{cont}_{temp,i} = \sum_{\substack{t \in T(top(k)) \\ s.t.\ time\ i \in t}} \frac{1}{|t|} \tag{4}$$

We first loop over every collected time interval t, and then estimate the probability of generating each time point within that time interval. If the "start" and "end" information are the same (e.g., ('1999-03', '1999-03')), i.e., the temporal expression refers to one particular month i, the length of the time interval $|t|$ is 1, and the probability of generating this time unit i is 100%. Then the corresponding $X^{cont}_{temp,i}$ is incremented by 1. However, if the "start" and "end" date are not the same (i.e., the temporal expression covers multiple months), each corresponding $X^{cont}_{temp,i}$ is increased by the value equal to 1 divided by the length of the time interval $|t|$, which also denotes the probability of generating each time unit i of the time interval. For example, including the time expression that covers ('2000-01', '2000-05') results in $X^{cont}_{temp,i}$ of any i within the time interval ('2000-01', '2000-05') being incremented by $\frac{1}{5}$.

3.3 Obtaining Time Series from Textual Information

The third step is to utilize the textual information from the retrieved documents and from the sentences containing temporal expressions obtained in the previous step, which respectively reflect the relevance between event description and documents' content, and the relevance between event description and the extracted sentences containing temporal expressions. We thus introduce two other univariate time series of length l:

$$\mathbf{X}^{doc}_{text} = \left\{X^{doc}_{text,1}, X^{doc}_{text,2}, ..., X^{doc}_{text,l}\right\} \tag{5}$$

$$\mathbf{X}^{sent}_{text} = \left\{X^{sent}_{text,1}, X^{sent}_{text,2}, ..., X^{sent}_{text,l}\right\} \tag{6}$$

$\mathbf{X}^{doc}_{text}$ denotes the document-to-event relevance time series and $\mathbf{X}^{sent}_{text}$ denotes the sentence-to-event similarity time series. We next introduce the computation of these two univariate time series.

[4] Similarly, for the case of day, week and year granularities, "from 1998 to 2002" is mapped to ('1998-01-01', '2002-12-31'), ('1998-W01', '2002-W53') and ('1998', '2002'), respectively.

[5] Time expressions that refer to periods outside of the time frame of the used news collection are for simplicity discarded, although they could be utilized in the future extensions of the method.

3.3.1 Document-to-event relevance time series. As previously mentioned, the top k relevant documents are ranked by BM25. Their relevance scores are computed by dividing the BM25 scores by the maximum value:

$$Rel(d) = \frac{BM25(d)}{MAX\_BM25(top(k))} \tag{7}$$

The computation of the document-to-event relevance time series at time unit i, i.e., $X^{doc}_{text,i}$, is as follows:

$$X^{doc}_{text,i} = \sum_{\substack{d \in top(k) \\ s.t.\ t_{pub}(d)=time\ i}} Rel(d) \tag{8}$$

The calculation of $X^{doc}_{text,i}$ is similar to Eq. 3, but here we take the relevance between an event description and a document into account, so the timestamps of documents that are less relevant would play a lesser role.

3.3.2 Sentence-to-event similarity time series. Among the sentences in $S(top(k))$ that contain the extracted temporal expressions, those that are relevant to the events should be considered more important, (e.g., the sentences that contain temporal expressions in the two relevant news articles shown in Fig. 1). Thus, for obtaining the last time series, we first calculate the relevance score between the event description and each sentence in $S(top(k))$, which indicates sentence importance and is measured by the cosine similarity between the event description encoding $\boldsymbol{e}$ and the sentence encoding $\boldsymbol{s}$. We utilize Sentence-BERT [34], a state-of-the-art neural network that can derive semantically meaningful sentence embeddings to encode the text. Then, the functions to compute the similarity score $Sim(e, s)$ and the sentence-to-event similarity time series at time unit i, denoted as $X^{sent}_{text,i}$, are as follows:

$$Sim(e, s) = cosine(\boldsymbol{e}, \boldsymbol{s}) \tag{9}$$

$$X^{sent}_{text,i} = \sum_{\substack{(t,s) \in (T(top(k)), S(top(k))) \\ s.t.\ time\ i \in t}} \frac{Sim(s, e)}{|t|} \tag{10}$$

The calculation is similar to the calculation of $X^{cont}_{temp,i}$. However, the corresponding sentence's relevance is taken into consideration, and the temporal information of the sentences that are less relevant to the event would be considered to a lesser extent.

3.4 Constructing Multivariate Time Series

The above-described four univariate time series of each event description are next standardized with mean of 0 and standard deviation of 1, and are aggregated to obtain a multivariate time series $\boldsymbol{\chi}$. The length of $\boldsymbol{\chi}$ equals to l and a slice of $\boldsymbol{\chi}$ at a time unit i is indicated as $\left\{X^{pub}_{temp,i}, X^{cont}_{temp,i}, X^{doc}_{text,i}, X^{sent}_{text,i}\right\}$. Therefore, with a batch size N, the input to the neural network has dimensions (N, M, l), where M equals to 4, and l is the length of the time series, that equals to the length of the time frame covered by the used news archive under the specified temporal granularity.

3.5 TEP-Trans Model

In the proposed TEP-Trans network, the Transformer architecture [43], which has excellent expressive ability for representing sequence information, is introduced to model the features of the input multivariate time series. Transformer is a neural network architecture that leverages self-attention mechanism to process a sequence of data, and is mainly used in NLP tasks. We adopt this architecture to approach the occurrence time estimation problem. Equipped with the self-attention mechanism, Transformer can access any part of the history regardless of distance, making it potentially more suitable for focusing on significant time steps in the past and grasping the temporal features of the time series. Figure 2 shows the overall architecture of the proposed TEP-Trans for estimating the event occurrence time.

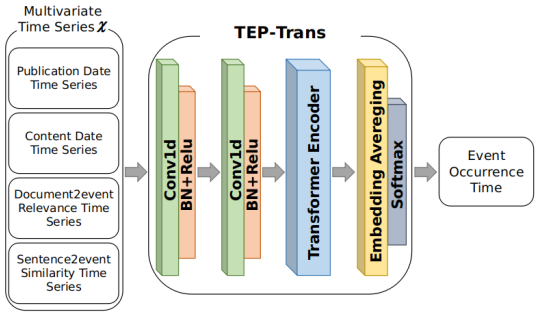

Figure 2: The TEP-Trans Model

TEP-Trans model is comprised of two convolutional blocks, a multilayer Transformer encoder block, followed by an embedding averaging layer and a softmax layer. Each convolutional block consists of a 1-D convolutional layer with the same padding, followed by a batch normalization layer and a ReLU activation layer. The multilayer Transformer encoder block takes the tensor that combines the results obtained from the last CNN block and positional encodings[6] as input, and derives important features of the input time series. Note that the input tensor or output tensor of convolutional blocks with the same padding as well as the Transformer encoder block always have a dimension size equal to length l. We use C_2 to denote the output channels of the second convolutional layers, and the result of the Transformer encoder block has dimensions (l, N, C_2). Then, the embedding averaging layer transforms the dimensions to $(l, N, 1)$, by performing the averaging across the last dimension's values. Finally, the result is transformed with dimension (N, l), and the estimated time is generated by the softmax layer. Note that we retain the tensor with length l and in the end, the features obtained from the Transformer block are fed into an embedding averaging layer instead of a fully connected layer, playing a similar role as global averaging pooling [27], which minimizes overfitting by largely reducing the number of parameters in the model. TEP-Trans model estimates the event occurrence time by exploiting the capability of convolutional layers for extracting useful knowledge and patterns, and then applying the Transformer for learning the internal representation of multivariate time series.

4 EXPERIMENTAL SETTING

4.1 Document Archive and Event Dataset

As previously mentioned, the NYT corpus [36] is used as the underlying temporal news collection, and is indexed by ElasticSearch. Over 1.8 million articles published between January, 1, 1987 and June, 19, 2007 with their publication dates are contained in the

[6]The functions to compute the positional encodings are derived from [43].

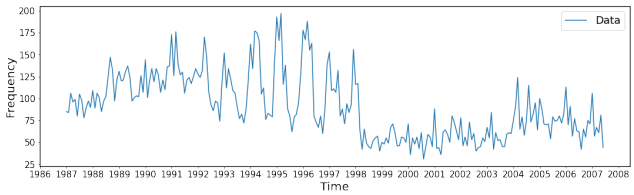

Figure 3: Frequency of event's occurrence time in the event dataset (month granularity)

corpus. We note that NYT has been often used for Temporal Information Retrieval researches [4, 19].

To the best of our knowledge, there is no available large dataset designed specifically for estimating event occurrence time within the time frame of the NYT corpus[7]. Hence, we construct the dataset[8] and make sure that the occurrence times of the included events fall into the time frame of the NYT corpus. We create a dataset containing 22,398 event descriptions, paired with their event occurrence times, and we partition the whole dataset randomly into a training set (80%), a development set (10%), and a test set (10%). The dataset has been constructed by crawling the descriptions and occurrence time of the events (ones between Jan 1, 1987 and Jun 19, 2007) from two resources: Wikipedia year pages[9] and On This Day web pages[10]. As the data extracted from these two resources sometimes contain records of the same events, we manually checked all the records that have the same event occurrence time and removed duplicates from the records that are on the same event. Fig. 3 shows the monthly distribution of events in our dataset[11].

4.2 Hyperparameters of the Model

For each event description, up to 15 keywords are extracted using Yake! with 2-grams as the maximum n-gram size and other parameters set as default. The top 50 ($k = 50$) relevant news articles are then retrieved from the NYT corpus. In the training phase, we run 100 epochs with a batch size of 64, and we apply Adam optimizer with learning rate $1e-3$. The hyperparameters of the TEP-Trans model that are used in the experiments are as follows: the kernel sizes and the strides of two 1-D convolutional layers with the same padding are set to 3 and 1, and the numbers of filters are set to 16 and 32, respectively. For the Transformer encoder layer, the number of layers, the number of heads, head dimension, and Transformer dropout are 3, 4, 200 and 0.2, respectively.

4.3 Evaluation Metrics

For the performance evaluation, we use: accuracy (ACC) and mean absolute error (MAE). The models are evaluated under these two metrics at day, week, month and year temporal granularities.

1) Accuracy (ACC): The percentage of the events whose occurrence time is correctly predicted.
2) Mean absolute error (MAE): The average of the absolute differences between the predicted time and the correct occurrence time, based on the specified granularity[12].

4.4 Compared Methods

We test the following models:

(1) **RG**: Random Guess. The event occurrence time is estimated by random guess, and the average of 1,000 random selections is used as the result.
(2) **DPD**: Data Peak Date. This naive baseline is used as another lower-bound reference besides the random guess. It always returns the date of the peak of the data's distribution (i.e., peak occurrence time of the aggregated events of the entire dataset) as the estimated result (e.g., under month granularity, DPD gives '1995-03', as can also be seen in Fig. 3).
(3) **BD** [44]: The burst detection based method which works such that given the temporal granularity, the occurrence time is estimated as the temporal value of the highest-scored peak within the largest burst of the publication date time series. The two parameters of BD, the window size and the cutoff factor, are set to 3 and 1.0, respectively.
(4) **NLM** [21]: The best proposed method in [21], that directly uses the timestamps of the top 15 retrieved documents as the predicted time. When there is more than one predicted time point, we use the time point that contains the largest number of retrieved documents.
(5) **MSSD**: The most similar sentence date method which works such that the event occurrence time is estimated as the time of the extracted sentence that has the largest similarity score with the event among sentences in $S(top(k))$.
(6) **AA** [12]: The best proposed model in [12]. It mainly focuses on the temporal expressions extracted from the document content and regards the publication date as an additional content temporal information. k is set to 50.
(7) **CNN-LSTM** [24]: The CNN-LSTM model has been often used to solve the multivariate time series prediction problems. We borrow this model to tackle our task which takes χ as input.
(8) **HEO-LSTM** [14]: The recently proposed variant of a method by [14] that was found by the authors to perform best and that estimates the event occurrence time by extracting relevant sentences from the Wikipedia, and applying a combination of task-specific and general-purpose feature embeddings for classification. As it is designed specifically to estimate the time at the year granularity, we compare this approach only at the year granularity. Note that HEO-LSTM is based on Wikipedia[13] and cannot work on other collections.
(9) **TEP-CNN**: Our proposed model without Transformer block, such that the CNN blocks are followed by embedding averaging layer and a softmax layer.
(10) **TEP-Trans**: The proposed Transformer-based model.

For fair comparison, all the above methods (except for HEO-LSTM, which uses entities and actions identified by pre-defined rules to

[7]Note that event extraction datasets such as ACE2005 or others are not applicable to our task as they require extracting event-related information from documents (actors, locations, dates) which is a different task than the event occurrence time prediction. Also, in their case, if the date information is to be delivered, it is always the one explicitly mentioned in text which does not require any prediction.
[8]The dataset is available at https://github.com/WangJiexin/Temporal-Event-Profiling/tree/main/data/dataset.
[9]https://en.wikipedia.org/wiki/List_of_years
[10]https://www.onthisday.com/dates-by-year.php
[11]Note that our dataset contains a subset of events of the dataset used by [14], however their events are annotated with only the yearly granularity dates.
[12]For example, at day granularity and month granularity, if MAE is 1, the average temporal distance is 1 day and 1 month, respectively.
[13]It needs to identify key entities of event descriptions, which are linked to the topics (i.e., titles) of the corresponding Wikipedia articles. For example, for the event description "The Sky Bridge is opened", the Wikipedia article "Sky Bridge" is used.

extract relevant Wikipedia sentences, and the first two naive methods, RG and DPD) use the same document retrieval approach (as described in Sec. 3.1) to retrieve their top k articles. Note also that RG and DPD are added only for determining the lower bound of the task to set a reference for better understanding of its difficulty.

5 EXPERIMENTAL RESULTS

5.1 Main Results

Table 3 shows the performance of the tested models in estimating event occurrence time. We can see that the proposed TEP-Trans model, that takes χ as input, surpasses other models in accuracy and MAE at all temporal granularities. We first note the results of the two straightforward, naive methods, RG and DPD, which both exhibit very poor performance, indicating that the task is not easy to be solved. Among the next four non-deep learning models that do not use χ, MSSD achieves the best performance on accuracy and MAE at all granularities. When comparing TEP-Trans with MSSD using accuracy and MAE, at the granularity of month, the improvements are 38.39% and 18.34% and at the fine-grained granularity of day, the improvements are 72.84% and 2.58%, respectively. MSSD performs actually best among all the baseline models on day granularity, which reveals that the temporal sentences that have large similarities with event descriptions are helpful for estimating the occurrence time.

The remaining approaches are based on neural networks, and except for HEO-LSTM, all take χ as input. The first model, CNN-LSTM, which is one of the most common neural network architectures applied in time series forecasting and prediction [23, 24, 28, 41], achieves relatively good performance on both metrics at year granularity. However, if the granularity turns to be finer, the performance of CNN-LSTM drops dramatically. The reason is that the output size of the last fully-connected layer, whose value equals to l (length of the time frame of the corpus at the chosen granularity) will also increase (e.g., l equals to 7,475 if day granularity is chosen). Thus, CNN-LSTM will overfit the training dataset and more data would be required to solve the problem. We next compare our proposed method with HEO-LSTM at the year granularity. Under the accuracy and MAE measure, our method surpasses HEO-LSTM by a large margin since the improvements are 162.70% and 37.42%, indicating that using their method that relies on Wikipedia is less effective for estimating the event occurrence time. Moreover, except RG and DPD, the other baseline methods also perform much better than HEO-LSTM, revealing that news archives could be used as another useful knowledge source to infer the event times.

Finally, we compare TEP-Trans with TEP-CNN - the model without the Transformer block. We can see that TEP-CNN achieves the second best performance on accuracy measure at week and month granularities. Therefore, CNN block can effectively extract important features of multivariate time series. Yet, by combining the Transformer block with powerful sequence pattern extraction capability, followed by the embedding averaging layer that helps to reduce overfitting problem, the important features useful for the event time estimation can be identified. Interestingly, we can still see quite a large improvement at day granularity. Under the accuracy and MAE measures, the improvements are 95.70% and 18.64%, respectively.

Table 3: Main results: Performance of different models at different granularities. Note that HEO-LSTM is designed specifically to estimate the time only at the year granularity

Model	Day		Week		Month		Year	
	ACC	MAE	ACC	MAE	ACC	MAE	ACC	MAE
RG	0.01	2482.12	0.08	355.25	0.40	81.57	4.77	6.91
DPD	0.04	2690.47	0.17	252.34	0.93	56.71	7.90	5.51
BD [44]	1.42	1418.26	14.01	215.80	18.75	49.70	27.09	4.37
NLM [21]	1.38	1300.34	15.53	194.16	21.87	45.85	33.52	3.80
AA [12]	6.02	1508.73	16.96	216.02	21.65	48.39	32.54	3.99
MSSD	9.50	1268.47	17.05	181.22	22.32	44.32	34.82	3.67
CNN-LSTM [24]	1.38	1382.38	7.49	174.26	23.30	37.04	37.54	3.21
HEO-LSTM [14]	-	-	-	-	-	-	15.58	4.81
TEP-CNN	8.39	1518.93	19.41	194.86	25.35	44.17	34.01	3.87
TEP-Trans	**16.42**	**1235.67**	**23.66**	**166.64**	**30.89**	**36.19**	**40.93**	**3.01**

Table 4: Performance of TEP-Trans model based on different input time series

Features	Day		Week		Month		Year	
	ACC	MAE	ACC	MAE	ACC	MAE	ACC	MAE
X^{pub}_{temp}	7.76	1563.20	13.25	216.92	17.63	48.04	30.26	3.80
X^{cont}_{temp}	6.60	1623.37	12.32	213.85	16.96	48.60	29.10	3.85
X^{doc}_{text}	8.52	1358.91	16.42	197.48	21.29	44.78	33.48	3.59
X^{sent}_{text}	9.86	1480.49	16.24	194.46	20.66	43.75	31.91	3.62
X^{pub}_{temp}, X^{cont}_{temp}	7.41	1578.50	15.31	211.88	19.28	46.16	30.53	3.73
X^{doc}_{text}, X^{sent}_{text}	13.34	1301.54	18.39	183.91	24.06	41.34	34.46	3.43
X^{pub}_{temp}, X^{doc}_{text}	11.02	1217.29	18.92	174.43	25.93	40.14	38.12	3.27
X^{cont}_{temp}, X^{sent}_{text}	12.18	1435.37	18.43	182.97	23.70	41.30	33.12	3.58
χ	**16.42**	**1235.67**	**23.66**	**166.64**	**30.89**	**36.19**	**40.93**	**3.01**

5.2 Input Ablation Study

We next conduct an ablation analysis on the input of the proposed TEP-Trans model. As shown in Table 4, the model using χ as an input achieves the best result, indicating that all the features contribute to the performance of our model. When considering only univariate time series, the models using X^{doc}_{text} or X^{sent}_{text} always perform better than the ones using X^{pub}_{temp} or X^{cont}_{temp}. This suggests that it is useful to combine the relevance of documents or sentences with embedded temporal information to the event descriptions.

We then show the results of aggregating two univariate time series. We can see that in Table 4, except for $\left\{X^{pub}_{temp}, X^{cont}_{temp}\right\}$ at day granularity, the models using one univariate time series achieve worse results on both metrics than models which aggregate the univariate time series with another one. For example, the model whose input is the multivariate time series obtained by aggregating time series of two types of textual information (indicated as $\left\{X^{doc}_{text}, X^{sent}_{text}\right\}$) performs better than the model using X^{doc}_{text} or X^{sent}_{text} only. In addition, we also note that our model achieves relatively good performance by taking $\left\{X^{cont}_{temp}, X^{sent}_{text}\right\}$ as input, which does not utilize the timestamp information. This suggests that our approach can also be applied over document collections without available timestamps.

5.3 Performance with Different Top k

Next, we investigate the effect of top k, that is the number of retrieved relevant documents used for constructing χ. Figure 4 plots the accuracy of different models with respect to k, which ranges

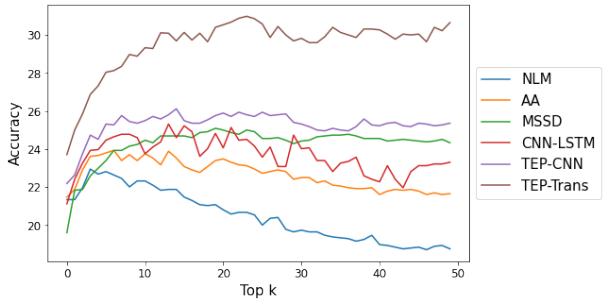

Figure 4: Performance of models with different top k at month granularity. Best viewed in color

from 1 to 50. First of all, TEP-Trans achieves the best result for all different top k and we can observe an initially growing trend of accuracy with larger k. The accuracy stabilizes around $k = 13$ and the TEP-Trans obtains its best accuracy level of 30.98% when $k = 24$. The TEP-CNN model whose last component comprises of an embedding averaging layer and a softmax layer exhibits similar tendency, and its best accuracy is 26.11% at $k = 15$. MSSD performance also reveals a similar trend along with the larger top k, which is reasonable since the event occurrence time is estimated as the time of the extracted sentence with the largest similarity score to the target event, so with the larger number k of top-relevant documents, a more similar and relevant sentence might be found. Unlike the above three methods, downward trends of accuracy of NLM, AA and CNN-LSTM can be observed when k is greater than a certain value (about 4, 14, 22, respectively), indicating that these models are incapable of filtering the noisy data well. Overall, we conclude that for the larger values of k, TEP-Trans can most effectively extract and filter information useful for event time estimation.

5.4 Analysis based on Event Characteristics

We next analyze the performance of our approach with respect to the event characteristics. In particular, we investigate how our model works based on the event description length and the shape of the temporal distribution of relevant documents. The former is represented by the number of words and the latter by the number of bursts in the publication date distribution over time[14], respectively. To test the effect of description length, the original test set of 2,240 event descriptions is first divided into two parts: 1,123 descriptions that have few words (less than or equal to 17) and 1,117 descriptions which are longer than 17 words. Note that when testing the effect of burstiness of the publication date time series, the number of bursts in the publication date distribution of events depends also on the specified granularity (coarser granularity results in less bursts in the distribution). Thus, for analyzing the impact of burstiness we divide the test set into two parts (few bursts and many bursts) that contain a similar number of records for each granularity.

Table 5 and Table 6 show the performance of our method based on the above-described data partitions. When considering the description length, we can see that TEP-Trans achieves better results on the event descriptions that have many words. The events that have longer descriptions are likely to retrieve documents that are more relevant to these events, which causes the obtained temporal or textual information to be more correct and precise. When considering the temporal distribution of the retrieved documents, the proposed model performs much better with the events that have only few bursts. It is more difficult to correctly estimate event time when the temporal distribution of relevant documents exhibits many bursts, since likely many other similar or related events occurred over time, which increases the difficulty of event date prediction.

[14]We use again the burst detection algorithm of [44] with the same parameters to detect and count the bursts.

Table 5: TEP-Trans results for events with few/many words

	Day		Week		Month		Year	
	ACC	MAE	ACC	MAE	ACC	MAE	ACC	MAE
few words	12.55	1312.46	18.96	174.02	25.73	38.16	36.50	3.01
many words	20.21	1158.48	28.22	159.23	35.88	34.20	45.14	3.00

Table 6: TEP-Trans results for events with few/many bursts

	Day		Week		Month		Year	
	ACC	MAE	ACC	MAE	ACC	MAE	ACC	MAE
few bursts	19.36	892.96	28.38	121.85	36.15	28.09	44.69	2.75
many bursts	10.85	1644.11	17.62	214.08	20.38	45.94	36.59	3.23

5.5 Comparison with QA Systems

Recently, several works proposed to employ Question Answering (QA) [50] for a variety of NLP problems [10, 30]. For example, McCann et al. [30] transform 10 different NLP tasks including natural language inference, sentiment analysis and relation extraction, into a QA paradigm and propose MQAN model to tackle all these tasks. In another work, Du and Cardie [10] have proposed a new paradigm for event extraction by formulating it as a QA task. Inspired by those ideas we test whether the event date estimation can be successfully solved using QA solutions.

We examine the performance of QA systems in the task of event occurrence time estimation, by first transferring the event descriptions to "when" questions, based on rule-based pattern matching (e.g., "Sarah Balabagan returns to the Philippines." is transferred to "When did Sarah Balabagan return to the Philippines?"). We then choose DrQA [7] for comparison, which is one of the most popular QA systems and is often used as a baseline in QA researches [25, 29, 47, 48]. Moreover, we examine DrQA models not only using the NYT corpus but also we investigate its performance when utilizing Wikipedia as the knowledge base. They are indicated as DrQA-NYT and DrQA-Wiki, respectively. Note that some answers returned by QA systems do not contain any temporal information and can not be compared with the ground truth (e.g., some numerical values "207", "100" which are not related to time, or other types of unrelated answers). Thus for ease of evaluation we only evaluate the models using accuracy metric. In addition, we test the models without week granularity since Wikipedia usually does not record week information of events.

As shown in Table 7, DrQA-Wiki performs much better than DrQA-NYT at the three granularites. We first found that the main reason is that the news articles often contain implicit temporal expressions, such as, "last month" or "yesterday", which might be returned as answers by DrQA-NYT. We then decided to resolve such implicit temporal expressions by using the inferred time, which is the timestamp information of the corresponding documents in order to improve the performance. We indicate this new system as DrQA-NYT-TempRes, and, as we can see, its performance is now closer

Table 7: Comparison with QA Models

Models	Day	Month	Year
	ACC	ACC	ACC
DrQA-Wiki [7]	7.90	11.56	26.65
DrQA-NYT [7]	0.62	1.74	11.47
DrQA-NYT-TempRes [7]	3.97	7.41	19.28
TEP-Trans	**16.42**	**30.89**	**40.93**

Table 8: Examples of event descriptions that are wrongly estimated by TEP-Trans, based on month granularity

No.	Description	Occurrence Time	Estimated Time
1	William Anthony Odom, North Carolina 15-year-old, accidentally hangs himself staging a gallows scene at a Halloween party.	1990-10	1996-10
2	The flu outbreak in Britain puts pressure on NHS.	2000-01	2005-11
3	Turin, Italy, is awarded the 2006 Winter Olympics.	1999-06	2006-02

to the one of DrQA-Wiki. However, a significant improvement on accuracy metric can be observed when comparing TEP-Trans with DrQA-Wiki and DrQA-NYT-TempRes at three granularites, indicating that common QA systems are incapable of answering "when" questions well. It also suggests that our method could serve as a fallback of a QA system when the answer is not explicitly given in the text or the answer is of coarse granularity.

5.6 Error Analysis

We also analyzed events for which our method has not produced correct results and we show some examples in Tab. 8. We found out that such events are usually not reported in the NYT archive, are periodical or recurring events, or are ones that include information about other popular events. For example, TEP-Trans model was not able to infer the occurrence time of event #1 in Tab. 8 since it is not reported in the NYT archive (although we found that it was actually reported in the LA Times archive.). The model could not correctly estimate the time of event #2 because similar events recurred multiple times and the description of event #2 is not precise enough. For the event #3, TEP-Trans model wrongly estimated the time as Feb. 2006 because most relevant articles are about the Winter Olympics held at that time.

6 APPLICATIONS

Finally, we look at how the proposed approach can be utilized in downstream tasks and we demonstrate its usefulness on one such task. There are quite many potential applications for temporal profiling of event mentions. Improving relevance estimation to enhance search within news archives or temporal diversification of search results [3, 13, 38], supporting entity extraction [1, 35], improving event mention extraction[15] [40], enhancing timeline generation[16] [11, 26, 39, 49] or question answering in long-term temporal news collections [45, 46] are some of the immediate examples.

6.1 Application for Question Answering

In this section we test our approach to see if it can improve effectiveness of answering diverse user questions in news archive. In particular, we use QANA system [45], which is an open QA system designed specifically for answering event-related questions, that do not contain any temporal expressions in their content, over temporal document collections. An important step in the system pipeline is the question time scope estimation aimed to gauge the possible time periods of the mentioned events based on analyzing the distribution of the retrieved documents. For example, since there is no temporal expression in the question: "Which party, led by Buthelezi, threatened to boycott the South African elections?", this step requires QANA to estimate the implicit date of the event mentioned in this question (which is "1993-08" under monthly granularity). We replace this step with our proposed approach, and the new system is indicated as QANA + TEP-Trans. We test both the systems on the dataset [46] composed of 500 questions that do not contain any temporal expressions using the NYT collection. This dataset has been created by merging data from various kinds of resources such as TempQuestions [16], SQuAD 1.1 [33] and questions from several history quiz websites. The results of the two systems are presented in Table 9. We can see that QANA + TEP-Trans system equipped with our proposed event time estimation approach outperforms the original system [45] for all the different ranges of the top N search results used. When considering the top 1 and top 15 documents, the improvement is in the range of 9.50% to 4.97%, and from 6.88% to 5.96% on Exact Match (EM) and F1 metrics, respectively. As demonstrated in this example, the proposed approach can be utilized as a building block for downstream tasks to further improve their performance.

Table 9: Performance of different models in QA task

Model	Top 1		Top 5		Top 10		Top 15	
	EM	F1	EM	F1	EM	F1	EM	F1
QANA [45]	21.00	28.90	28.20	36.85	34.20	44.01	36.20	45.63
QANA + TEP-Trans	**23.00**	**30.89**	**29.60**	**38.17**	**35.40**	**45.49**	**38.00**	**48.35**

7 CONCLUSIONS

In this paper we present an effective TEP-Trans model for estimating the event occurrence time. We are the first to address this task by applying the ideas of multivariate time series analysis and the Transformer architecture, which altogether result in promising performance. The proposed approach is capable of modeling useful features of the input multivariate time series and achieves state-of-the-art results at all the temporal granularities. In addition, unlike most of the existing methods which estimate the occurrence time based on temporal information from timestamp or content signals, or which are designed over synchronic document collections (e.g., Wikipedia), our approach addresses the problem by jointly utilizing two types of temporal information and two types of textual information. Through the experiments we learn that these four types of information contribute altogether to the performance of our model, as demonstrated in the experiments.

In future, we will explore the inter-relations between the retrieved documents that were published at different time units in order to capture the features reflecting the temporal development of events, as such data could be another useful signal for event date prediction. We will also apply the proposed approach to other IR and NLP tasks besides open question answering.

[15] Judging if two text spans are about the same event can be improved since not only text similarity can be considered but also overlap of their estimated temporal profiles.
[16] For generating timelines some approaches use explicit temporal expressions mentioned in news [39]. With our method one could find implicit references to news events as there is no need for any explicit date to be present in such references.

REFERENCES

[1] Prabal Agarwal, Jannik Strötgen, Luciano Del Corro, Johannes Hoffart, and Gerhard Weikum. 2018. Dianed: time-aware named entity disambiguation for diachronic corpora. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 686–693.

[2] Omar Alonso, Michael Gertz, and Ricardo Baeza-Yates. 2007. On the value of temporal information in information retrieval. In *ACM SIGIR Forum*, Vol. 41. ACM New York, NY, USA, 35–41.

[3] Klaus Berberich and Srikanta Bedathur. 2013. Temporal diversification of search results. In *Proceedings of the SIGIR 2013 workshop on time-aware information access*.

[4] Ricardo Campos, Gaël Dias, Alípio M Jorge, and Adam Jatowt. 2015. Survey of temporal information retrieval and related applications. *ACM Computing Surveys (CSUR)* 47, 2 (2015), 15.

[5] Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. 2020. YAKE! Keyword extraction from single documents using multiple local features. *Information Sciences* 509 (2020), 257–289.

[6] Angel X Chang and Christopher D Manning. 2012. Sutime: A library for recognizing and normalizing time expressions.. In *Lrec*, Vol. 2012. 3735–3740.

[7] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051* (2017).

[8] Wisam Dakka, Luis Gravano, and Panagiotis Ipeirotis. 2010. Answering general time-sensitive queries. *IEEE Transactions on Knowledge and Data Engineering* 24, 2 (2010), 220–235.

[9] Supratim Das, Arunav Mishra, Klaus Berberich, and Vinay Setty. 2017. Estimating event focus time using neural word embeddings. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 2039–2042.

[10] Xinya Du and Claire Cardie. 2020. Event Extraction by Answering (Almost) Natural Questions. arXiv:2004.13625 [cs.CL]

[11] Demian Gholipour Ghalandari and Georgiana Ifrim. 2020. Examining the State-of-the-Art in News Timeline Summarization. *arXiv preprint arXiv:2005.10107* (2020).

[12] Dhruv Gupta and Klaus Berberich. 2014. Identifying time intervals of interest to queries. In *Proceedings of the 23rd ACM international conference on conference on information and knowledge management*. 1835–1838.

[13] Dhruv Gupta and Klaus Berberich. 2016. Diversifying search results using time. In *European Conference on Information Retrieval*. Springer, 789–795.

[14] Or Honovich, Lucas Torroba Hennigen, Omri Abend, and Shay B Cohen. 2020. Machine reading of historical events. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 7486–7497.

[15] Adam Jatowt, Ching-Man Au Yeung, and Katsumi Tanaka. 2013. Estimating document focus time. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*. 2273–2278.

[16] Zhen Jia, Abdalghani Abujabal, Rishiraj Saha Roy, Jannik Strötgen, and Gerhard Weikum. 2018. TempQuestions: A Benchmark for Temporal Question Answering. In *Companion of the The Web Conference 2018 on The Web Conference 2018*. International World Wide Web Conferences Steering Committee, 1057–1062.

[17] Rosie Jones and Fernando Diaz. 2007. Temporal profiles of queries. *ACM Transactions on Information Systems (TOIS)* 25, 3 (2007), 14–es.

[18] FM Jong, Henning Rode, and Djoerd Hiemstra. 2005. Temporal language models for the disclosure of historical text. (2005).

[19] Nattiya Kanhabua, Roi Blanco, and Kjetil Nørvåg. 2015. Temporal Information Retrieval. *Foundations and Trends in Information Retrieval* 9, 2 (2015), 91–208. https://doi.org/10.1561/1500000043

[20] Nattiya Kanhabua and Kjetil Nørvåg. 2008. Improving temporal language models for determining time of non-timestamped documents. In *International conference on theory and practice of digital libraries*. Springer, 358–370.

[21] Nattiya Kanhabua and Kjetil Nørvåg. 2009. Using temporal language models for document dating. In *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 738–741.

[22] Nattiya Kanhabua and Kjetil Nørvåg. 2010. Determining time of queries for re-ranking search results. In *International Conference on Theory and Practice of Digital Libraries*. Springer, 261–272.

[23] Tae-Young Kim and Sung-Bae Cho. 2018. Predicting the household power consumption using CNN-LSTM hybrid networks. In *International Conference on Intelligent Data Engineering and Automated Learning*. Springer, 481–490.

[24] Tae-Young Kim and Sung-Bae Cho. 2019. Predicting residential energy consumption using CNN-LSTM neural networks. *Energy* 182 (2019), 72–81.

[25] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. *arXiv preprint arXiv:1906.00300* (2019).

[26] Artuur Leeuwenberg and Marie-Francine Moens. 2018. Temporal information extraction by predicting relative time-lines. *arXiv preprint arXiv:1808.09401* (2018).

[27] Min Lin, Qiang Chen, and Shuicheng Yan. 2013. Network in network. *arXiv preprint arXiv:1312.4400* (2013).

[28] Tao Lin, Tian Guo, and Karl Aberer. 2017. Hybrid neural networks for learning the trend in time series. In *Proceedings of the twenty-sixth international joint conference on artificial intelligence*. 2273–2279.

[29] Shanshan Liu, Xin Zhang, Sheng Zhang, Hui Wang, and Weiming Zhang. 2019. Neural machine reading comprehension: Methods and trends. *Applied Sciences* 9, 18 (2019), 3698.

[30] Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730* (2018).

[31] Arunav Mishra and Klaus Berberich. 2016. Event digest: A holistic view on past events. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 493–502.

[32] Christian Morbidoni, Alessandro Cucchiarelli, and Domenico Ursino. 2018. Leveraging linked entities to estimate focus time of short texts. In *Proceedings of the 22nd International Database Engineering & Applications Symposium*. 282–286.

[33] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250* (2016).

[34] Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084* (2019).

[35] Shruti Rijhwani and Daniel Preotiuc-Pietro. 2020. Temporally-informed analysis of named entity recognition. In *Proceedings of ACL*. 1–13.

[36] Evan Sandhaus. 2008. The new york times annotated corpus. *Linguistic Data Consortium, Philadelphia* 6, 12 (2008), e26752.

[37] Shashank Shrivastava, Mitesh Khapra, and Sutanu Chakraborti. 2017. A concept driven graph based approach for estimating the focus time of a document. In *International Conference on Mining Intelligence and Knowledge Exploration*. Springer, 250–260.

[38] Jaspreet Singh, Wolfgang Nejdl, and Avishek Anand. 2016. History by Diversity: Helping Historians search News Archives. *Proceedings of the 2016 ACM Conference on Human Information Interaction and Retrieval, CHIIR 2016, Carrboro, North Carolina, USA, March 13-17, 2016* (2016), 183–192.

[39] Julius Steen and Katja Markert. 2019. Abstractive Timeline Summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*. 21–31.

[40] Jannik Strötgen and Michael Gertz. 2012. Event-centric search and exploration in document collections. In *Proceedings of JCDL*. 223–232.

[41] Goutham Swapna, Soman Kp, and Ravi Vinayakumar. 2018. Automated detection of diabetes using CNN and CNN-LSTM network and heart rate signals. *Procedia computer science* 132 (2018), 1253–1262.

[42] Shikhar Vashishth, Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. 2019. Dating documents using graph convolution networks. *arXiv preprint arXiv:1902.00175* (2019).

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.

[44] Michail Vlachos, Christopher Meek, Zografoula Vagena, and Dimitrios Gunopulos. 2004. Identifying similarities, periodicities and bursts for online search queries. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*. ACM, 131–142.

[45] Jiexin Wang, Adam Jatowt, Michael Färber, and Masatoshi Yoshikawa. 2020. Answering Event-Related Questions over Long-Term News Article Archives. In *European Conference on Information Retrieval*. Springer, 774–789.

[46] Jiexin Wang, Adam Jatowt, Michael Färber, and Masatoshi Yoshikawa. 2021. Improving question answering for event-focused questions in temporal collections of news articles. *Information Retrieval Journal* (2021), 1–26.

[47] Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerald Tesauro, Bowen Zhou, and Jing Jiang. 2018. R3: Reinforced Ranker-Reader for Open-Domain Question Answering. In *AAAI*.

[48] Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with bertserini. *arXiv preprint arXiv:1902.01718* (2019).

[49] Yi Yu, Adam Jatowt, Antoine Doucet, Kazunari Sugiyama, and Masatoshi Yoshikawa. 2021. Multi-TimeLine Summarization (MTLS): Improving Timeline Summarization by Generating Multiple Summaries. In *Proceedings of the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

[50] Fengbin Zhu, Wenqiang Lei, Chao Wang, Jianming Zheng, Soujanya Poria, and Tat-Seng Chua. 2021. Retrieving and Reading: A Comprehensive Survey on Open-domain Question Answering. *arXiv preprint arXiv:2101.00774* (2021).