Where the Event Lies: Predicting Event Occurrence in Textual Documents

Andrea Ceroni, Ujwal Gadiraju, Jan Matschke*, Simon Wingert†, and Marco Fisichella
L3S Research Center, Leibniz Universität Hannover, Appelstr. 9a, Germany
{ceroni, gadiraju, fisichella}@L3S.de
*jmatschke@gmx.de
†simon.wingert@gmx.net

ABSTRACT

Manually inspecting text in a document collection to assess whether an event occurs in it is a cumbersome task. Although a manual inspection can allow one to identify and discard false events, it becomes infeasible with increasing numbers of automatically detected events. In this paper, we present a system to automatize *event validation*, defined as the task of determining whether a given event occurs in a given document or corpus. In addition to supporting users seeking for information that corroborates a given event, event validation can also boost the precision of automatically detected event sets by discarding false events and preserving the true ones. The system allows to specify events, retrieves candidate web documents, and assesses whether events occur in them. The validation results are shown to the user, who can revise the decision of the system. The validation method relies on a supervised model to predict the occurrence of events in a non-annotated corpus. This system can also be used to build ground-truths for event corpora.

Keywords

Event Validation; Event Detection; Evaluation; Corpus Construction

1. INTRODUCTION

Event descriptions can provide concise summaries of news articles, making the seek for event related information more enjoyable for general readers and more effective for professionals like historians and journalists. We do not focus on how events are detected or extracted from text, but rather tackle the problem of assessing their occurrence within a given corpus. In line with both the basic definition of *event* given by the Topic Detection and Tracking (TDT) project [1] and with more recent works on event detection, e.g. [5, 6, 8], we model events as a set of participants related within a given time period. For instance, the event {(*Novak Djokovic*, *Roger Federer*, *US Open*), *2015-08-30 to 2015-09-15*} would represent the participation of two tennis players in the 2015 US Open.

Manually assessing whether an event occurs in a document collection becomes infeasible in scenarios where events are continuously and automatically detected from news streams on a large-scale. In this paper, we present a system to automatize the event validation process by predicting whether a given event has evidence within a set of unannotated documents retrieved from a corpus. Our system can find documents that corroborate the occurrence of an input event, thus simplifying the task of manually searching for event-related information to confirm or deny its verity. In more detail, the developed system allows to specify one event, retrieves candidate Web documents, and assesses what are the documents (if any) where it occurs. The validation judgments are shown to the user, who can explore the documents and possibly revise the decision of the system. Given the possibility for users to provide their own validation judgments, which are made persistent along with events and documents, the application can also be used to acquire ground truth data for a given set of input events. Event validation can also be applied as a post processing step of event detection to boost the precision within the detected set of events by reducing the number of false events with respect to a given corpus.

To validate events, the system relies on a state of the art method [4] that extracts features from events and documents and exploits them to train a model via supervised machine learning. Once trained, the model predicts event occurrence in terms of the mutual conformation of the event participants within the event timespan. Although our system uses the Web as a source for documents due to its easy accessibility and wide event coverage, any document collection could be used in principle owing to the lack of assumptions on the nature of events and documents within the validation model.

To the best of our knowledge our system is the first of its kind, showcasing automatic event validation and supporting users in looking for evidence corresponding to the verity of events.

2. RELATED WORK

The automatic detection of events within text has been widely studied, e.g. in [1, 5, 6]. However, event detection is different from event validation, since the output of the former (i.e., a set of *events*) constitutes the known input for the latter. Although the task of event validation with respect to a corpus can be modeled as the task of retrieving documents relevant to input events, checking the mere appearance of event participants in text has been proved to be insufficient to validate the occurrence of events in documents while establishing mutual relationships and temporal conformation [4].

Besides the method showcased by our system [4], which combines a wide set of features extracted from events and documents within a supervised model, another attempt to event validation has been proposed in [3], where the occurrence of events in documents is evaluated based on hand-crafted rules. Araki et al. [2] performed historical fact validation by casting the problem as *Passage Retrieval*: they assess event validity in terms of the textual similarity between facts and passages of fixed length.

SIGIR '16, July 17-21, 2016, Pisa, Italy

DOI: http://dx.doi.org/10.1145/2911451.2911452

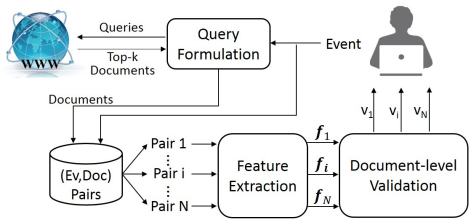


Figure 1: Approach overview of automatic event validation.

Available system implementations do not tackle event validation, but they rather consider other event-related applications like extraction [7], tracking [12], retrieval of event-related information [10], visualization [9], and retrospective exploration [11].

3. AUTOMATIC EVENT VALIDATION

The event validation process is depicted in Figure 1 and it consists of three phases. These are: *query formulation*, which generates queries from the event specified by the user and retrieves candidates documents from the Web; *feature extraction*, which extracts features from (event,document) pairs; *document-level validation*, to identify documents containing evidence of the event (if any).

3.1 Query Formulation

The Query Formulation phase takes an event specified by the user as input. An event is made of (i) a set of participants, representing the participants of the event, and (ii) a start and end date, indicating the timespan within which the event occurred. This is in line with event definitions used in previous works [5, 6, 8].

Given an event, queries are constructed by concatenating the event participants along with month and year of the event timespan (one distinct query for each month). The Bing Search API is used to perform queries and to retrieve the top-20 Web pages for each query in terms of URLs. We chose the Web as a source for documents due to its easy accessibility and wide event coverage, but any document collection could be used. After removing duplicates and discarding non-crawlable Web pages, plain text is extracted by using BoilerPipe[1]. Finally, events and documents are logically coupled and stored as *(event, document)* pairs for later use.

3.2 Feature Extraction

The features described in [4] are extracted from pairs to form the input to the validation model. These are logically split into three groups: *event features* describe the event without coupling it with any document; *document features* are extracted from the plain text of each document, independent of the event to be validated; *pair features* are extracted from pairs to give information about the extent to which a document contains evidence of the event. Stanford CoreNLP[2] is used for POS tagging, named entity recognition, and dates extraction. The features extracted from each pair p are concatenated in a feature vector $\boldsymbol{f}_p$ as input to the validation.

3.3 Document-level Validation

The last step of event validation consists of assessing the validity of pairs, i.e. whether the document in the pair contains *evidence* of the event. Note that we do not aim at stating whether an event is true or false in general, but whether it occurs in the retrieved documents. Formally, an event is said to be *valid* (i.e. to have evidence of its occurrence in a document) with a threshold $\tau, 0 < \tau \leq 1$, iff at least $\tau\%$ of the event participants conform together in an event reported in the document, strictly within the timespan of the event. This implies that an event can exhibit different degrees of evidence within the same document, depending on the imposed evidence threshold. For instance, if 75% of the participants relate together in a document and within the event timespan, then the event would have evidence in the document if a threshold $\tau = 50\%$ is considered, but would not have evidence for $\tau = 100\%$. Imposing an evidence threshold makes the validation a classification task and allows to have a more intuitive notion of document-level validity: a document (and then the corresponding pair) is judged as *valid* if the number of event participants that conform together within it is equal to or greater than the threshold, and *invalid* otherwise.

Given the feature vectors $\boldsymbol{f}_p$ extracted from (event,document) pairs p and an evidence threshold τ, an SVM with RBF Kernel trained considering τ as threshold is used to predict the validity $\gamma_p = SVM\left(\boldsymbol{f}_p, \tau\right)$ of each pair, where the validation judgment γ_p is binary and can assume the values $\{valid; invalid\}$. In our system, the user can choose among three evidence thresholds (50%, 65%, 100%) and the SVM trained with the chosen value is used accordingly. Refer to [4] for further details on the training process.

4. SYSTEM DEMONSTRATION

4.1 Implementation

We implemented a publicly available Web application[3,4] using the JSF framework. For a given input event, candidate documents are retrieved through the Bing Search API and then posed as input to the back-end validation model [4], implemented in Java. It uses the Stanford CoreNLP parser for feature extraction and SVM classifiers trained with the LibSVM library to validate each (event,document) pair. The user can review the documents and possibly revise the validity judgments given by the system. This feedback is stored in a MySQL database for future re-training of the validation model.

The user interface of our application is shown in Figure 2. The user can specify an input event by filling the form on the left-hand side. The required information consists in the event participants, it start and end dates, the validity threshold, and the number of candidate documents to retrieve. On the right-hand side is the list of retrieved documents along with their validation judgments (either *valid* or *invalid*), given with respect to the input event and evidence threshold. For each document, the user can inspect the plain textual content extracted with BoilerPipe and actually used for validation (Figure 3), as well as visit its original URL. When reviewing the text, the user has the chance of changing the original validation judgment, possibly marking cases of uncertainty. To simplify the review process of the user, the occurrences of event participants and dates detected by the Stanford CoreNLP parser are highlighted.

4.2 Scenarios

The main usage scenario that we consider consists of a user willing to retrieve information related to a given event, possibly provided by a certain news agency or an event detection algorithm, to either corroborate or deny its occurrence. Given the possibility for the users to provide and store their own validation judgments, the application can also be used to acquire ground truth data for a set of input events. The judgments about event validity are not absolute: they relate to the Web as ground truth and to the documents that have been thus retrieved. For instance, an event having no evidence in any retrieved Web pages might still occur in other documents

[1] http://code.google.com/p/boilerpipe/

[2] http://nlp.stanford.edu/software/corenlp.shtml

[3] http://forgetit.l3s.uni-hannover.de:8081/eventvalidation

[4] Video available here: www.l3s.de/~ceroni/SIGIR2016/Demo.mp4

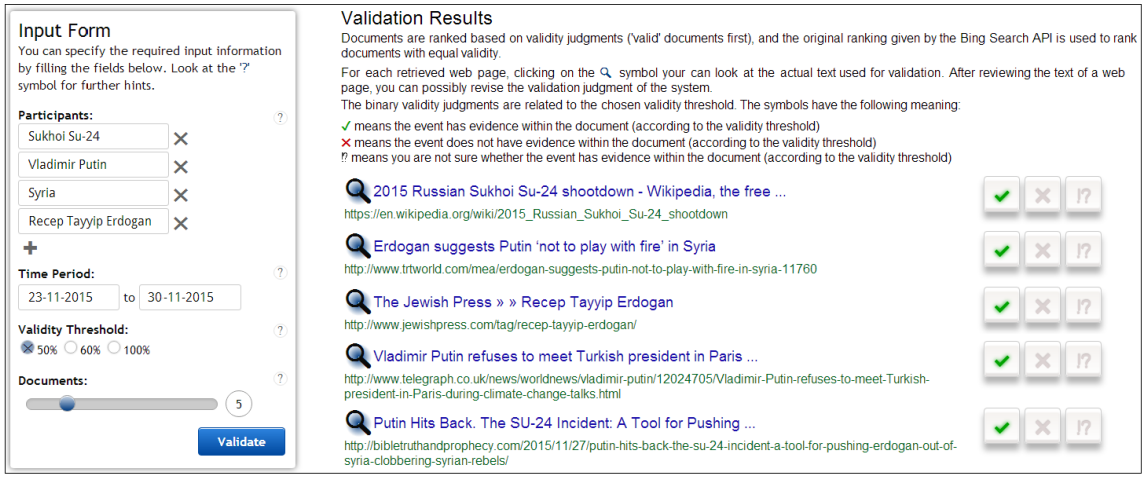


Figure 2: Example of a valid input event (left-hand side) and corresponding validation results (right-hand side).

in the Web that were not retrieved, or in corpora disjoint from the Web. In the rest of this section we discuss three examples of usages of the application, namely in presence of true, false, and less newsworthy input events. Since the system relies on the Bing Search API for retrieval and the set of documents populating the Web is continuously evolving, the set of documents shown in the figures and discussed in this section might be different from the ones got when performing the same queries at future points in time.

The validity judgments given for pairs always refer to the specified evidence threshold τ (Section 3.3). An (event, document) pair is said to be *valid* with a threshold τ if at least $\tau\%$ of the event participants relate together in the document, strictly within the event timespan. For instance, if 3 out of 4 participants conform together in a document, the occurrence of the event in it will be *valid* for $\tau = 50\%$ and $\tau = 65\%$, but *invalid* for $\tau = 100\%$ (all participants are required). We consider $\tau = 50\%$ in the rest of this section.

4.2.1 *True Events*

Let us assume that a user wants to check whether the entities *Sukhoi Su-24*, *Vladimir Putin*, *Syria*, and *Recep Tayyip Erdogan* have been related to each other within the period from 23-11-2015 to 30-11-2015. Actually they were, since on 24^{th} November 2015 a Russian Sukhoi Su-24M bomber aircraft flying at the Syria-Turkey border was shot down by a Turkish fighter jet, leading to tensions between the two leaders Putin and Erdogan. The input form as well as the retrieved and validated Web pages representing this scenario are shown in Figure 2. The top-5 retrieved documents have been all judged as valid, which means that all of them are declared to contain evidence of the input event. Inspecting their content, one can observe that they report the description of the actual event as well as political discussions generated from it. An excerpt of the first document reporting the event is shown in Figure 3, where matching event participants and dates are highlighted in the application for sake of inspection and review.

Note that the mutual conformation of the event participants to the same event in the document is not a sufficient condition for the (event,document) pair to be declared as valid (i.e. the event occurs in the document). It has to be integrated by the temporal validity, which means that the event participants relate together strictly within the specified timespan of the event. As we will show in Section 4.2.2, (event,document) pairs exhibiting mutual occurrence and relation of event participants not explicitly within the event timespan are judged as *invalid*.

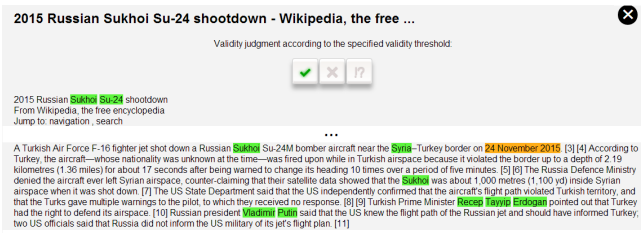


Figure 3: Excerpt of a retrieved document reporting the input event.

4.2.2 *False Events*

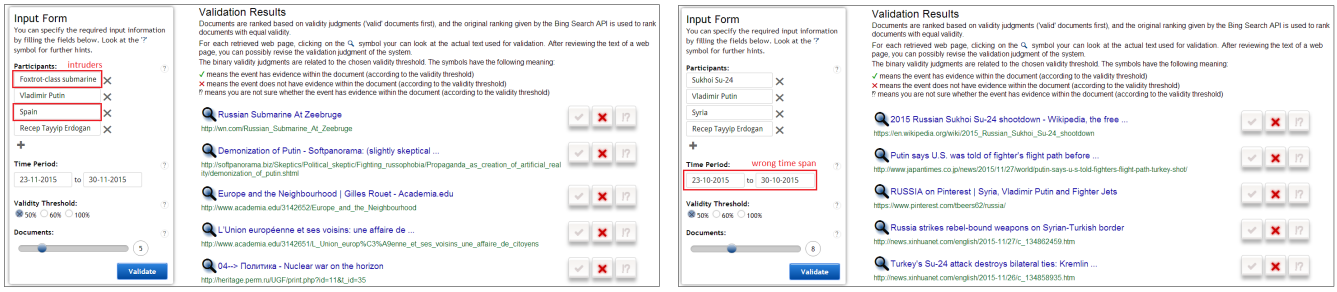
We focus on two reasons of event invalidity: (i) when event participants are not reported in the retrieved documents as they do not relate to each other, and (ii) when the participants did relate to each other but not within the specified event timespan.

Starting from the event mentioned in Section 4.2.1, let us modify the set of participants by replacing *Syria* and *Sukhoi Su-24* with two "intruders", namely *Spain* and *Foxtrot-class Submarine* (patrol submarines built in the Soviet Union). The retrieved documents along with their validation judgments are reported in Figure 4a, which shows that all retrieved documents have been judged as not containing the event. The input entities did not participate in a common event, and consequently, the system did not find any evidence.

To show temporal invalidity, let us keep the original participant set of the event in Section 4.2.1 and let us shift the event timespan one month in advance, namely from 23-10-2015 to 30-10-2015. The output of the system is shown in Figure 4b, where the result list differs from the one in Figure 2 because a different month has been used to formulate the query. All the retrieved documents have been declared as not containing the event, although participants actually occur in them, because no signal about temporal validity was found. As stated by the problem definition itself (Section 3.3), event validation strictly takes the temporal dimension of events into account when making decisions about their occurrence within documents.

4.2.3 *Unpopular Events*

Besides newsworthy events with a large amount of corroborating documents, as the one considered in Section 4.2.1, event validation also has to cope with less popular events, whose occurrences in documents might be less frequent or unclear. As an example, let us consider an event with participants *SIGIR*, *Chile*, *Ricardo Baeza-*



(a) Invalid event because of unrelated participants.

(b) Invalid event because of wrong time span.

Figure 4: Examples of invalid input events along with their validation results.

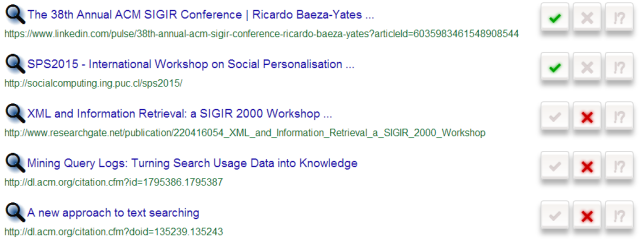


Figure 5: Top-5 retrieved documents for a low newsworthy event.

Yates from 04-08-2015 to 16-08-2015. This validation scenario is different from the previous ones because (i) the event has a high impact only for a particular community of researchers and (ii) the periodicity of the SIGIR conference might introduce documents referring to other venues outside the specified period. The top-5 retrieved documents and their validation judgments are shown in Figure 5, which shows that only a part of the documents contain evidence of the event. The others refer either to previous SIGIR venues, or contain works of Ricardo Baeza-Yates, or even completely unrelated documents.

4.2.4 *User Feedback*

Automatically determining whether an event occurs in a document collection is a challenging task. The usage of the Web itself as ground truth poses further challenges; due to its ubiquitous accessibility and wide event coverage, it can be regarded as a noisy source owing to the presence of unstructured and potentially untrustworthy sources (like blogs and forums) with questionable verity. Therefore, our application keeps users in the loop by allowing them to review the plain text extracted from the retrieved Web pages. By highlighting the event participants and dates detected in the text, our system simplifies the inspection and presents the user with the possibility of revising a potentially incorrect validity judgment deduced by the system (Figure 3). Such feedback is stored and can be used for future re-training of the validation model. Besides errors committed by the back-end classifier, other possible misclassifications can be caused by errors introduced during boilerplate removal and named entity or temporal expression extraction (e.g. important content of a Web page might not be included in the plain text or temporal expressions might not be detected by the NLP parser). Background information known to a user but not explicitly present in the text might be a further source of judgment discrepancy.

5. CONCLUSION

In this paper, we presented a system to automatize the event validation process by predicting whether a given event has evidence within a set of unannotated documents, thus simplifying the task of manually searching for event-related information to confirm or deny its verity. The developed system allows to specify an event, retrieves candidate web documents, and assesses what are the documents (if any) where it occurs. The validation method relies on a state of the art model for event validation. The user can review the documents and revise the validation judgments given by the system. Given the possibility for users to provide their own validation judgments, the application can also be used to acquire ground truth data for a given set of input events. By coupling our system with the powerful crowdsourcing paradigm, one can build large event corpora with corresponding ground truths in a scalable and cost-effective manner. Although we chose the Web as a source for documents, due to its easy accessibility and wide event coverage, other document collections could be explored and included in the system.

Acknowledgments This work was partially funded by the European Commission in the context of the FP7 project QualiMaster (grant No: 619525) and H2020 project AFEL (grant No. 687916).

6. REFERENCES

[1] J. Allan, R. Papka, and V. Lavrenko. On-line new event detection and tracking. In *SIGIR*, 1998.

[2] J. Araki and J. Callan. An annotation similarity model in passage ranking for historical fact validation. In *SIGIR*, 2014.

[3] A. Ceroni and M. Fisichella. Towards an entity-based automatic event validation. In *ECIR*, 2014.

[4] A. Ceroni, U. Gadiraju, and M. Fisichella. Improving event detection by automatically assessing validity of event occurrence in text. In *CIKM*, 2015.

[5] A. Das Sarma, A. Jain, and C. Yu. Dynamic relationship and event discovery. In *Proc. of WSDM '11*, 2011.

[6] Q. He, K. Chang, and E.-P. Lim. Analyzing feature trajectories for event detection. In *Proc. of SIGIR '07*, 2007.

[7] E. Kuzey and G. Weikum. Evin: Building a knowledge base of events. In *Proc. of WWW '14*, 2014.

[8] A. J. McMinn, Y. Moshfeghi, and J. M. Jose. Building a large-scale corpus for evaluating event detection on twitter. In *CIKM*, 2013.

[9] A. J. McMinn, D. Tsvetkov, T. Yordanov, A. Patterson, R. Szk, J. A. Rodriguez Perez, and J. M. Jose. An interactive interface for visualizing events on twitter. In *Proc. of SIGIR '14*, 2014.

[10] V. Milicic, G. Rizzo, J. L. Redondo Garcia, R. Troncy, and T. Steiner. Live topic generation from event streams. In *WWW '13*, 2013.

[11] A. Mishra and K. Berberich. ExposÉ: Exploring past news for seminal events. In *Proc. of WWW '15*, 2015.

[12] J. B. Vuurens, A. P. de Vries, R. Blanco, and P. Mika. Online news tracking for ad-hoc queries. In *Proc. of SIGIR '15*, 2015.