Exploring Entity-centric Networks in Entangled News Streams

Andreas Spitz
Institute of Computer Science, Heidelberg University
Heidelberg, Germany
spitz@informatik.uni-heidelberg.de

Michael Gertz
Institute of Computer Science, Heidelberg University
Heidelberg, Germany
gertz@informatik.uni-heidelberg.de

ABSTRACT

The increasing number of news outlets and the frequency of the news cycle have made it all but impossible to obtain the full picture from online news. Consolidating news from different sources has thus become a necessity in online news processing. Despite the amount of research that has been devoted to different aspects of new event detection and tracking in news streams, solid solutions for such entangled streams of full news articles are still lacking. Many existing works focus on streams of microblogs since the analysis of news articles raises the additional problem of summarizing or extracting the relevant sections of articles. For the consolidation of identified news snippets, schemes along numerous different dimensions have been proposed, including publication time, temporal expressions, geo-spatial references, named entities, and topics. The granularity of aggregated news snippets then includes such diverse aspects as events, incidents, threads, or topics for various subdivisions of news articles. To support this variety of granularity levels, we propose a comprehensive network model for the representation of multiple entangled streams of news documents. Unlike previous methods, the model is geared towards entity-centric explorations and enables the consolidation of news along all dimensions, including the context of entity mentions. Since the model also serves as a reverse index, it supports explorations along the dimensions of sentences or documents for an encompassing view on news events. We evaluate the performance of our model on a large collection of entangled news streams from major news outlets of English speaking countries and a ground truth that we generate from event summaries in the Wikipedia Current Events portal.

CCS CONCEPTS

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

KEYWORDS

entity network; implicit network; news stream; document indexing

ACM Reference Format:
Andreas Spitz and Michael Gertz. 2018. Exploring Entity-centric Networks in Entangled News Streams. In *WWW '18 Companion: The 2018 Web Conference Companion, April 23–27, 2018, Lyon, France.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3184558.3188726

WWW '18 Companion, April 23–27, 2018, Lyon, France

ACM ISBN 978-1-4503-5640-4/18/04.
https://doi.org/10.1145/3184558.3188726

1 INTRODUCTION

Reading it in the paper in the morning is a common idiom for catching up with the news that is becoming increasingly less applicable. Putting aside the obvious departure from printed news, both the temporal aspect and the grammatical singular are less and less accurate. News are not reported and consumed in the morning but in a constant news cycle throughout the day, published by a multitude of news outlets with varying degrees of reliability, political bias, and overlapping content. It is these *entangled streams of news* that the reader has to wade through to stay informed. Despite similarities between the news cycle and streams of microblogs, social media cannot take on the mantle of investigative journalism, which relies on argumentative texts and is less focused on the instant than it is on the evolution of stories. In this context, the so called *Five Ws* of *Who?*, *When?*, *Where?*, *What?*, and *Why?* are questions of central importance that serve the journalist and the reader in uncovering news. Naturally, these questions put an emphasis on entities as pivotal components of news. In information retrieval, this is reflected in the definition of an event as *something that happens at a given place and time between a group of actors* [2], originating in topic detection and tracking and highlighting the central role of entities for inducing structure in the unstructured texts of news articles.

In large entangled news streams, far more than one news article tends to be required to retrieve the full picture [32]. However, a lot of information is replicated between or even within individual news streams and thus redundant. Intuitively, this motivates two major subtasks in automated news analysis: identifying event mentions in unstructured texts, and aggregating them across documents. These tasks are referred to as *new event detection* and *event tracking* [3], and can be augmented by detecting *topics* [7] that put individual documents into context. To make identified events accessible to users, a central step is thus their aggregation into threads of events along some dimension(s). Many different approaches have been proposed to this end. Some focus on a geographic aggregation and visualization of news sources [33], while others focus on the temporal aggregation [12], or both [38]. Alternative approaches use the participating entities directly [9, 16, 30]. In the case of a temporal aggregation, different temporal dimensions can be considered, such as the dates in the documents [21], or external information such as the publishing date [4] and edit histories [15]. With regard to timelines, another important aspect is then the temporal order, as the SemEval-2015 task for cross-document event ordering shows [20]. Beyond the above dimensions, more recent approaches include aggregation on a topic level [1] or based on word embeddings [22].

When considered for contrastive explorations of content, the above approaches suffer from two critical drawbacks: the limited number of *aggregation dimensions* and the *aggregation granularity* level. No existing approach covers the entirety of available dimensions and it is indeed questionable whether an aggregation along

all dimensions at once is realistically possible. Perhaps even more critically, the results are always coarse structures due to an aggregation either on the document, event, or topic level. However, events are commonly defined as composite mentions of (named) entities, which form the stitching points between individual news streams. After all, we consume news about people, organizations, or locations of interest and follow them over time and in different contexts. Is it then not a more reasonable approach to retain this entity-centric structure of news in a suitable document representation for subsequent analyses, and aggregate only where necessary and in exactly the dimensions that fit the exploratory task?

As a first step towards addressing this shortcoming, we introduce *entity-centric implicit networks* as a representation of entangled news streams. Based on the concept of implicit entity networks for static document collections [27, 30], we include entity relation information, a spatial and two temporal dimensions (temporal expressions and publication metadata), and the context of mentions in a comprehensive framework for entity-centric analyses. On the technical side, our model addresses the inherent scaling issues of entangled news streams by utilizing efficient entity-centric queries to localized graph substructures, and the streaming graph updates take advantage of incremental adjustments to relevance measures for queries against the data [8, 36]. Furthermore, the implicit representation serves as an (inverse) index for retrieval tasks without requiring the storage of proprietary news article content. On the application side, our model provides a more fine-grained and versatile representation of entangled news streams than any previous approach. Instead of utilizing document- or event-centric indexing, we focus on the level of entities and contexts and use them as stitching points between individual news threads. The model supports a wide range of tasks, including entity-centric topic and event extraction and tracking, contextual search, contrastive source comparison, and exploratory visualizations of the underlying streams.

Contributions. Our contributions are fourfold. **(i)** We propose a comprehensive model for entity-centric exploration and retrieval tasks on large entangled news streams. **(ii)** We discuss graph-based context-sensitive clustering of joint entity mentions in both static and streaming applications. **(iii)** We introduce a clustering of entities along naturally evolving topics that does not suffer from a pre-defined number of topics or topic degradation at low ranks like traditional topic models. **(iv)** We evaluate our model on a large entangled stream of news from international outlets. We provide the resulting network data, including links to the original articles[1].

2 RELATED WORK

To our knowledge, no previous work supports the comprehensive, entity-centric exploration of entangled news streams. Thus, we give an overview of works that cover some of these aspects.

Entity-centric Exploration and Analysis. A fundamental result in entity-centric document analysis with strong emphasis on the detection of event descriptions is by Feng and Allan, who formalize the concepts of *incident threading* and *event threading* [11]. While event threading captures the internal structure of news topics by adding causal or temporal relations, incident threading merges mentions of identical entity cooccurrences. Later works utilize similar concepts. Kanhabua et al. assess the importance of temporal expressions based on the cooccurrences of entities and temporal anchor texts within individual sentences [15]. Gupta et al. present EventMiner, a framework for extracting events from collections of documents [13] that is very comprehensive in its use of temporal expressions and named entities, but does not scale. Mishra and Berberich link coarse-grained events from news articles to corresponding Wikipedia pages [21]. Similarly, Ceroni et al. use entity mentions and temporal information to confirm the occurrence of events in a document collection [9]. Although the above works focus on entities, they only consider static document collections and do not support entity-centric exploration or streaming news.

Analysis of Articles in News Streams. A number of frameworks offer comprehensive analyses of streaming news. Lydia is a large-scale aggregation tool for news articles [17] with numerous subsequent publications. The European Media Monitor builds and processes a repository of multilingual European news articles [5]. News Stand monitors and retrieves RSS feeds to extract geographic content from articles for spatial clustering and visualization [33]. The enBlogue system allows the identification of emerging topics from news streams in real time [4], but has so far been applied with a focus on blogs and microblogs. A shortcoming of the above approaches is the lacking support for entity-centric explorations.

Ahmed et al. combine topic modeling, clustering and named entity recognition to distinguish topics, story lines, and entities in streaming news articles [1], but do not include the effects of entity cooccurrences. To support ad-hoc tracing of news streams, Vuurens et al. utilize the clustering and qualification of titles and sentences in news articles [35]. Moran et al. introduce the use of word embeddings to enhance first story detection in microblogs [22].

Many further approaches to streaming news analysis exist, but few of them consider temporal information, and none of them include temporal information along with entities, terms, and topics.

Network-based Document Models. Yang et al. classify news documents into topics and measure topic novelty by using both keywords and named entities with relative weighting for event-level novelty detection [37]. For a similar purpose, Das Sarma et al. build entity dynamic relation graphs to identify entities participating in trending events, but exclude locations [26]. More generally, Blanco and Lioma explore a network-based approach that models terms as nodes in a graph with edges weighted by cooccurrence counts [6]. Rousseau and Vazirgiannis use an unweighted but directed graph to account for term order in a document's text [25]. They focus on the sentence level and do not include entities, thus limiting the capability of the model for entity exploration.

All of these approaches use graphs to answer *specific* questions about a document collection or stream. Here, we focus on a *comprehensive* network representation that supports a multitude of subsequent analyses. A similar approach was recently presented by Spitz and Gertz [27, 30], who construct an implicit network of entity mentions from a static document collection to support exploratory tasks. While their model could be adapted to support efficient streaming updates, an exploration along the dimension of publication dates or the context of entities is not included. In the following, we thus describe how a more general implicit network model can be realized in a streaming environment.

[1]News graph, evaluation data, and code are available for download at our website: https://dbs.ifi.uni-heidelberg.de/resources/newsstream/

3 ENTITY NETWORK MODEL

Based on the intuition that entity relations can be derived from joint entity mentions, we construct a network of entity relations from the named entity classes *locations*, *organizations*, *actors*, and *dates*. Together, these form the basis of the LOAD model for implicit networks [30], which we improve from a static to a streaming model by adding **(i)** term embeddings to encode the context of entity mentions for refined queries, **(ii)** an adaptation to the news domain by considering publication times as a second temporal dimension beyond temporal expressions in the documents, and **(iii)** an adaptation to entangled news streams and concurrent events by using a multigraph model with (partial) edge aggregation schemes.

3.1 Entity Multigraph Model

Let N be a collection of news articles. Each document $n \in N$ consists of sentences $s \in n$. We denote the set of all sentences in all documents as $S := \bigcup_{n \in N} \{s \in n\}$. To consecutively number the sentences, let $\sigma : S \to \mathbb{N}$ map a sentence to its index in the document in which it occurs. We then consider each sentence to be a collection of words, which we partition into entity classes.

Graph Nodes. In the following, we consider words to be units within a text that has been tagged for named entities. We distinguish between the named entity classes *locations* L, *organizations* O, *actors* A, and *dates* D, according to the LOAD model. All remaining words constitute the set of terms T, which is defined as

$$T := \bigcup_{s \in S} \{w \in s \mid w \notin L \cup O \cup A \cup D\}, \tag{1}$$

Thus, a sentence is a multiset of entities $s \in (L \cup O \cup A \cup D \cup T)^*$. In the following, we refer to the set of actors, locations, organizations, dates and terms as *entities*. We define the union of the aforementioned seven classes $V := L \cup O \cup A \cup D \cup T \cup S \cup N$ as the nodes of the graph and employ a function $\eta : V \to \{L, O, A, D, T, S, N\}$ that maps each node $v \in V$ to the corresponding class $\eta(v)$.

Graph Edges. To obtain a graph representation $G = (V, E)$, we construct a set of edges $E := E_C \cup E_P$ based on two criteria: *containment* and *proximity*. Containment represents edges E_C between entities and sets, such that an entity is connected to a set that contains the entity. This type of edge provides provenance and context information for entities. Proximity edges E_P encode the cooccurrence of entities within at least one common document and represent implicit entity relations. Edges of the proximity type introduce parallel edges in the graph since one edge is induced whenever two entities cooccur. To distinguish between parallel edges, we rely on *instances* $I \subseteq \mathbb{N}$ of entity cooccurrences, along with an injective mapping $\iota : V \times V \to I$. Here, $i = \iota(v, w) \in I$ represents an instance of the cooccurrence of two entities v and w in some unique ordering, such that the tuple $e = (v, w, i)$ denotes an edge between v and w. For example, if entities v and w cooccur in a document, this induces an edge (v, w, i). If they later cooccur again, we obtain a new edge (v, w, j). Note that a document may contain multiple instances of the same entities. Formally, we obtain

$$E_C := \{(v, w, i) \mid v \in w \wedge i = \iota(v, w)\} \tag{2}$$

$$E_P := \{(v, w, i) \mid \exists n \in N : v, w \in n \wedge i = \iota(v, w)\} \tag{3}$$

Thus, containment edges occur between entities and sentences or sentences and documents, while proximity edges connect entities. Since there is a surjection from S to N, edges between entities and documents can be reconstructed from edges between entities and sentences. The resulting graph is undirected and we therefore do not distinguish between edges (v, w, i) and (w, v, i) where this is clear from context. We denote with $\mathcal{N}(v)$ the neighbourhood of a node v, i.e., all nodes that are connected to v by at least one edge.

3.2 Edge Weights and Edge Attributes

To assign weights and attributes, we distinguish between edges of the containment type E_C and edges of the proximity type E_P.

Set Containment Edges. Edges of the containment type are binary relations. Therefore, the resulting edges are essentially unweighted, although parallel edges may occur in rare cases. To simplify the subsequent notation, we define a distance function $\delta : E_C \to \mathbb{N}$ for edges between an entity v and a sentence s as $\delta(v, s, i) := 0$ if $v \in s$, and $\delta(v, s, i) := \infty$ if $v \notin s$. The distance between sentences and documents is defined analogously.

Occurrence Proximity Edges. Edges of the proximity type are more complex due to more finely nuanced distances and parallel edges caused by multiple cooccurrences. Since it is this entity cooccurrence information that encodes the relevant information for later analyses, we want to preserve these multiple edges and enrich them with additional information for later aggregation (see Section 3.4). We consider three fundamental concepts, namely (1) the publication time, (2) the textual distance between the mentions of two entities, and (3) the context of the mentions.

Publication Time. We assume that a publication time or retrieval date is known for each news article. Let $\tau : N \to \mathbb{N}$ map each document $n \in N$ to its publication time $\tau(n)$. Let n_i be the document that contains an instance i inducing an edge $e = (v, w, i)$ between two entities. We then assign $\tau(e) := \tau(n_i)$ to edge e.

Textual Distance. By overloading the function σ, we can map each entity of an instance to the index of the sentence in which this entity occurs. Thus, let $\sigma(v, i)$ denote the number of the sentence in which entity v occurs in instance i. For example, if entity v in instance i occurs in the first sentence of a document, then we have $\sigma(v, i) = 1$. In analogy to the LOAD model, the textual sentence distance $\delta : E_P \to \mathbb{N}$ of two entities can then be written as

$$\delta(v, w, i) := |\sigma(v, i) - \sigma(w, i)| \tag{4}$$

For example, if entities v and w cooccur in a document such that v is contained in the first sentence and w is contained in the fourth sentence, then $\delta(v, w, i) = 3$. Thus, if two entities occur in the same sentence, their distance is 0. If v and w never occur in the same document, we set $\delta(v, w) := \infty$. To include the distance of entity cooccurrences in the graph, we assign to each edge $e = (v, w, i)$ the corresponding distance $\delta(v, w, i)$ as an edge attribute $\delta(e)$.

Context Embeddings. To conserve the context of joint entity mentions, we use a vector embedding of terms in the context window of two entities. Formally, an embedding is a function $\varepsilon : T \to \mathbb{R}^k$ that maps a term to a point in a k-dimensional vector space. To obtain the context of two entities in a cooccurrence instance, we define a context window as a function of those entities. Let $\omega : E_P \to 2^S$, such that ω maps an instance to a set of sentences. Specifically, let n_i be the document containing an instance i, then

$$\omega(v, w, i) := \{s \in S \mid s \in n_i \wedge \sigma(v, i) \leq \sigma(s) \leq \sigma(w, i)\} \tag{5}$$

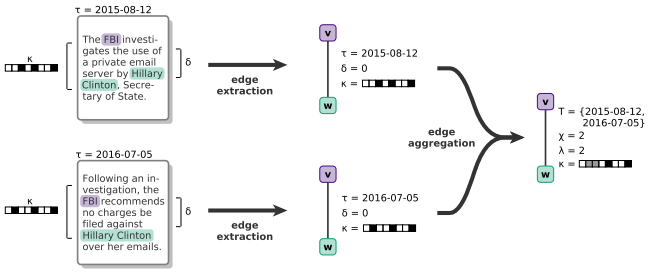

Figure 1: Schematic view of the model. Edges between entities v and w are extracted with context κ, distance δ and timestamp τ. If edges with a similar context between the same entities re-occur, they may be aggregated.

where w.l.o.g. $\sigma(v,i) \leq \sigma(w,i)$. Thus, $\omega(v,w,i)$ consists of the sentences containing v and w, and all sentences inbetween. Based on this, we define the context of an edge $e = (v,w,i)$ as the normalized sum of all embeddings of the terms in the context window $\omega(v,w,i)$. Thus, let $\kappa : E_P \to \mathbb{R}^k$ denote the context function

$$\kappa(v,w,i) := \sum_{s \in \omega(v,w,i)} \sum_{t \in s} \frac{\varepsilon(t)}{|\omega(v,w,i)|} \tag{6}$$

where $|\omega(v,w,i)|$ denotes the number of terms in the context window. The removal of stop words and the limitation to content words is feasible in this step to reduce noise. For each edge $e = (v,w,i)$, we store $\kappa(e)$ as an attribute to identify pairs of entities that appear in similar contexts. For an overview of the model, see Figure 1.

3.3 Aggregated Graph Attributes

We now lay the foundations for the entity-centric exploration of news in their context. A shortcoming of the LOAD model is the aggregation of all parallel edges to obtain a simple graph. While such an aggregation makes graph representations of large document collections feasible, it does not distinguish between mentions in different contexts. In news analysis, however, the number of contexts in which two entities cooccur is limited. Thus, aggregating edges by context still results in a stark reduction of the number of edges, while also preserving the context of entity cooccurrences for later analyses. Here, we argue that such an approach should be flexible enough to handle arbitrary numbers of contexts. Furthermore, an aggregation by context partially preserves the multiplicity of edges, while simultaneously collapsing unjustifiably duplicate edges to enable a more focused extraction of information from the resulting graph. In particular for entangled streams of news articles with redundant information, such an approach is clearly beneficial.

To obtain an aggregated graph $G_A = (V, A)$, we require a new set of aggregated edges A with aggregated attributes. Let v and w denote two entities and let I_a denote a set of instances that induce parallel edges $E_a := \{(v,w,i) \in E \mid i \in I_a\}$ between them. In the following, we discuss how to derive the aggregated edge features.

Aggregated Edge Importance. This weight derives an overall strength of the relation between two entities from the sentence distances of individual edges. Here, the dissimilarity of a sentence distance is transformed into a similarity by a decaying exponentiation. The individual similarities are then added over all aggregated edges.

Table 1: Overview of edge attributes in the graphs.

τ	publication time	ω	context window
δ	textual sentence distance	κ	context embedding
σ	sentence index	λ	# aggregated edges
ι	instance of cooccurrence	η	node type
ε	term embedding	χ	edge importance

Thus, we compute a weight for the aggregated edge $a = (v,w,j)$ as

$$\chi(a) := \sum_{e \in E_a} \exp(-\delta(e)) \tag{7}$$

Aggregated Publication Dates. For a temporal analysis, we store the set of all publication dates, which we assume to be distinct as long as the granularity of time is fine enough. For lower granularities, this attribute is effectively a multiset of dates. Formally,

$$T(a) := \bigcup_{e \in E_a} \{\tau(e)\} \tag{8}$$

Aggregated Context. The context is the primary component of the edge aggregation (see Section 3.4). However, once edges are aggregated, a single context vector is sufficient to represent an edge and facilitate context-sensitive queries. Therefore, the contexts of individual edges can be aggregated as the mean of the context vectors. Since the context of two entities whose mentions are separated by a couple of sentences is likely less important than two mentions within the same sentence, we normalize individual contributions by the distance of the mentions δ. Thus,

$$\kappa(a) := \frac{1}{|E_a|} \sum_{e \in E_a} \frac{\kappa(e)}{\delta(e)+1} \tag{9}$$

Number of Aggregated Edges. To maintain the context centroid in the streaming aggregation model, we store for each edge the number of individual edges that were aggregated. Thus, we define an attribute function $\lambda : A \to \mathbb{N}$ with $\lambda(a) := |E_a|$.

Based on these four attributes, parallel edges in $G = (V,E)$ can be combined to create the aggregated graph $G_A = (V,A)$ as we describe in the following. For containment edges, only the importance and the number of aggregated edges are meaningful. For the importance of containment edges, note that the exponentiation turns the distances into a value of 1 for existing edges and 0 for missing edges. An overview of edge attributes is shown in Table 1.

3.4 Edge Aggregation Schemes

For edge aggregation, two settings are possible. If real-time queries on streaming data are of interest, a streaming aggregation can be used to process news articles as they come in, and merge new edges to existing ones. Conceptually, this resembles streaming first story detection for microblogs [23] but retains the entire contextual information. Extracted edges are treated as information fragments that can be merged with existing edges (if the context is sufficiently similar) or treated as new edges (if the context is sufficiently different). We refer to this as the *streaming approach*. Alternatively, all edges of all articles can be stored to retain the unaggregated information. In this case, the edges are aggregated locally between pairs of nodes at query time. We refer to this as the *static approach*.

Streaming Edge Aggregation. Streaming aggregation supports a real-time analysis of news articles as they become available and

Algorithm 1 Addition of edges in the streaming approach.

Input: G_A, document graph $G_n = (V_n, E_n)$, threshold t

$V_A \leftarrow V_A \cup V_n$
for $e = (v, w, i) \in E_n$ **do** ▷ for all new edges
  $E_a \leftarrow \{(v', w', i) \in E \mid v = v' \wedge w = w'\}$
  if $E_a = \emptyset$ **then** ▷ if this is the first edge betw. v and w
    $a \leftarrow (v, w, i)$ ▷ create new aggregated edge
    $\lambda(a) \leftarrow 1$ ▷ set multiplicity to 1
    $A \leftarrow A \cup \{a\}$ ▷ insert as a new edge
  else ▷ otherwise, find candidates for merging
    $a \leftarrow \arg\max_{a' \in E_a} \{sim(e, a')\}$ ▷ find most similar edge
    if $sim(e, a) \leq t$ **then** ▷ if similarity below threshold
      $\lambda(a) \leftarrow 1$ ▷ set multiplicity to 1
      $A \leftarrow A \cup \{a\}$ ▷ insert as a new edge
    else
      $\chi(a) \leftarrow \chi(a) + \chi(e)$ ▷ update importance
      $T(a) \leftarrow T(a) \cup \{\tau(e)\}$ ▷ merge date sets
      $\kappa(a) \leftarrow \frac{1}{\lambda(a)+1}(\kappa(a)\lambda(a) + \kappa(e))$ ▷ update context
      $\lambda(a) \leftarrow \lambda(a) + 1$ ▷ increase multiplicity

Output: G_A

Algorithm 2 Aggregation of edges in the static approach.

Input: Multigraph $G = (V, E)$, clustering algorithm

Initialize $A \leftarrow \emptyset$ and $l \leftarrow 0$
for $(v, w) \in V \times V, v < w$ **do** ▷ for all pairs of nodes
  $E' \leftarrow \{(v', w', i) \in E \mid v = v' \wedge w = w'\}$ ▷ select edges
  $C \leftarrow \text{cluster}(E')$ ▷ cluster edges by context similarity
  for $E_a \in C$ **do** ▷ for each cluster of edges
    $l \leftarrow l + 1$ ▷ increase edge index
    $a \leftarrow (v, w, l)$ ▷ create new aggregated edge
    $\lambda(a) \leftarrow |E_a|$ ▷ set multiplicity
    $\chi(a) \leftarrow \sum_{e \in E_a} \exp(-\delta(e))$ ▷ aggregate importance
    $\kappa(a) \leftarrow \frac{1}{|E_a|} \sum_{e \in E_a} \frac{\kappa(e)}{\delta(e)}$ ▷ combine contexts
    $T(a) \leftarrow \bigcup_{e \in E_a} \{\tau(e)\}$ ▷ merge date sets
    $A \leftarrow A \cup \{a\}$ ▷ insert as a new edge

Output: Aggregated graph $G_A = (V, A)$

utilizes a *similarity threshold* parameter t. As new articles n are added to the collection, multigraph representations $G_n = (V_n, E_n)$ are constructed. Each edge in G_n is inserted into the collection graph G_A by aggregating it with existing edges based on context similarity. Here, any suitable vector similarity measure can be used to compare the embeddings. We distinguish between three cases for a new edge $e = (v, w, i) \in E_n$. (1) If e is a containment edge, it is added to the set of aggregated edges A. (2) If v and w are disconnected in G_A, then e is added to A. (3) Otherwise, if G_A already contains edges between v and w, we check if e is sufficiently similar to the centroid context vector of an existing edge and aggregate it with the existing edge $a \in A$ that is the best fit and update the edge attributes accordingly. If no existing edge is similar enough, e is inserted into A. For a detailed description, see Algorithm 1.

Static Edge Aggregation. The static aggregation of edges is a post-hoc processing of the collected news stream, in which parallel edges are clustered. Here, a clustering approach without a fixed number of clusters is required, as the optimal number of aggregated edges per pair of nodes is unknown and highly varying for different pairs of nodes. Additionally, outliers and noise should be kept separate from the clusters since many news articles do not belong to major news stories. After clustering, edges within each cluster are aggregated into a single edge. See also Algorithm 2.

Complexity. The complexity of the streaming approach is in $O(I \cdot \langle p \rangle)$, where $\langle p \rangle$ is the average multiplicity of parallel *aggregated* edges between node pairs. The number of instances I scales linearly with the number of articles N for a given cooccurrence window size, and $\langle p \rangle$ is small enough to support similar edge detection by linear scans, as we show in Section 5.4. The complexity of the static approach is higher with $O(I \cdot C)$, where C is the complexity of the selected clustering algorithm, which is likely at least quadratic in the edge multiplicity of *unaggregated* edges. However, due to the localized clustering, the approach is parallelizable by node pairs.

Stability. Both approaches produce deterministic results, although this depends on the temporal order of articles in the streaming approach. Obviously, the aggregated graphs differ between the two approaches. In Section 5, we compare their efficacy. In practice, the static approach can be applied in a streaming setting if sufficient memory is available to cluster edges locally at query time.

4 NETWORK CREATION AND EXPLORATION

Based on the above model, we consider application scenarios in which such a representation supports the exploration of news, and show exploratory results on a large stream of news articles.

Exploration Focus. Using an implicit network representation, any task that can be formulated as entity or term rankings or the extraction of weighted entity graph patterns is viable. In particular, events as dyadic or triadic structures of entities can be queried efficiently [27]. Due to the transitivity of edge aggregation (edges can always be aggregated further), all entity-centric exploration methods designed for LOAD also work on our context-enriched model. We thus focus specifically on novel exploration methods that utilize temporal data and the context of entity mentions to extract evolving entity-centric topics from entangled news streams.

4.1 News Document Data

In the following, we describe the acquisition and preparation of the news data as well as the construction of the graph representation.

Data Collection. Since we require entangled news streams from multiple outlets, standard corpora such as the New York Times corpus cannot be used. Instead, we collect articles from the RSS feeds of international outlets with a focus on quality news. For content extraction, we use manually created rules since these allow a clean extraction of article contents (including multi-page articles) at a level that automatic boilerplate removal does not support [29].

Specifically, we use articles from 14 English speaking news outlets located in the U.S. (CNN, LA Times, NY Times, USA Today, CBS News, The Washington Post, IBTimes), Great Britain (BBC, The Independent, Reuters, SkyNews, The Telegraph, The Guardian), and Australia (Sidney Morning Herald). The RSS feeds of these outlets differ, but we focus on feeds related to political news. The time frame for our data collection is June 1 to November 30, 2016. We remove articles that have less than 200 or over 20,000 characters

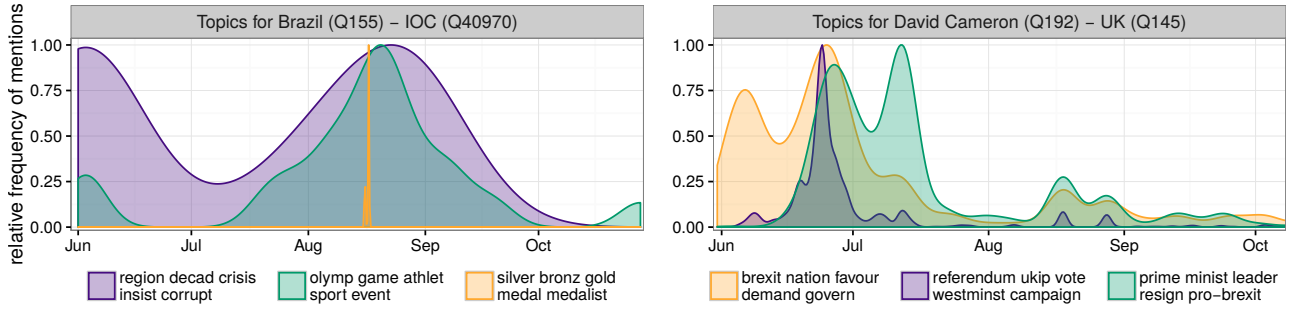

Figure 2: Evolution of contextual topics (i.e., edge contexts) for selected entity pairs with streaming aggregation ($t = 0.65$). Shown is the relative aggregated frequency of publication dates. Contexts are derived from the $k = 5$ terms in the neighbourhood of both entities whose context is most similar to the edge context. Q identifiers denote Wikidata IDs. Left: relation between Brazil and the International Olympic Committee. Right: relation between David Cameron and the United Kingdom.

(due to NER limitations) or more than 100 disambiguated entities per article (i.e., lists). The final collection contains 127,485 articles over a period of six months, with a total of 5.4M sentences.

Data Preparation. Data preparation consists of five steps: recognition of named entities, entity linking, entity classification, part-of-speech and sentence tagging, and temporal tagging. For the recognition and disambiguation of named entities to Wikidata IDs, we use the Ambiverse API[2]. To classify named entities into actors, locations, and organizations, it is possible to use Wikidata hierarchies directly, but this can be problematic due to their constantly evolving structure [28]. Therefore, we map Wikidata IDs to YAGO3 entities [18] and classify them according to the YAGO hierarchy. For actors, we use the class `wordnet_person_100007846`, and for organizations `wordnet_social_group_107950920`. For locations, no comprehensive WordNet class exists, so we use `yagoGeoEntity`, which was designed for this purpose [14]. For the extraction and normalization of temporal expressions we run HeidelTime in the news domain setting [31]. Finally, for sentence splitting and part-of-speech tagging, we use the Stanford POS tagger [34].

Network Construction. We proceed as described in Section 3. Terms are stemmed with the Porter stemming algorithm [24], and we impose a minimum word length of 4 characters for terms. The window size for entity cooccurrence extraction is set to 5. As context embeddings, we use Google's pre-trained 300-dimensional word2vec [19] word embeddings. The resulting networks has 5.7K dates, 27.7K locations, 72.0K actors, 19.6K organizations, and 351K terms, which are connected by 83.4M edges (before aggregation).

4.2 Contextual Topic Evolution

To highlight an exploratory application, we demonstrate the extraction of contextual topics. We extract topics that best describe the individual contexts in which two entities are mentioned together and consider their evolution over time. Naturally, multiple such contexts may exist, which is reflected by the multiple parallel edges.

Contextual Topics. Recall that a context vector $\kappa(a)$ is associated with each aggregated edge $a = (v, w)$. We define a *contextual topic* of edge a as a weighted list of terms that describe the context in which entities v and w occur in instances included in a. To extract the contextual topics for all aggregated edges between these entities, we retrieve all terms $T_x = \mathcal{N}(v) \cap \mathcal{N}(w) \cap T$ in the joint neighbourhood of the two nodes along with all edges that connect them to v or w. We aggregate these edges such that each term x is connected to both v and w by exactly one edge, which we denote with a_v and a_w. Based on these triangular structures, we obtain a ranking score for each term $x \in T_x$ in relation to edge a as

$$r_t(x|a = (v, w)) := \min\{\text{sim}(\kappa(a), \kappa(a_v)), \text{sim}(\kappa(a), \kappa(a_w))\} \quad (10)$$

Intuitively, we are ranking terms by how closely the context in which they occur with an entity matches the context in which the entities occur together. We create such a ranking of terms for all aggregated edges between v and w. For each such edge, we select the k top-ranked terms to describe the topic. Thus, we obtain a natural language description for each of the edges between the two entities. Since edges are aggregated based on context similarity, the assumption is that the terms then describe the context of an edge and that each edge in turn represents a topic.

Results. To demonstrate the expressiveness of contextual topics, we show a timeline visualization of topics for pairs of entities. To extract these, we use a cosine similarity of the context vectors and rank the terms as described above. Then, we assign to each edge between the two entities the $k = 5$ top-ranked terms as descriptors. We select the three top edges by multiplicity (i.e., the aggregated edges with the highest λ values). Since each such edge is associated with a set of publication times, we can plot the evolution of the topics over time. The results for two entity pairs are shown in Figure 2. On the left, we see the evolution of topics for Brazil and the IOC (i.e., the Olympic Games). One can easily identify contexts as dealing with corruption, sports, and the awarding of medals. Specifically, the award topic spikes precisely at the date of the games. The second example shows the relation of David Cameron to the United Kingdom during the Brexit crisis. While all three topics are related to this issue, the referendum topic spikes at the proper date and the shift between the remaining topics towards Cameron's resignation only after the referendum is pronounced.

In summary, the intuitive notion of aggregated edges as contexts corresponds well with our observations. Thus, term-based topic descriptors assign meaning to such edges, and their extraction serves to facilitate exploratory analyses of the news stream. Since the extraction utilizes only a localized substructure of the network around the focus entities, the process is efficient and allows a near real-time exploration of the entire entangled news stream. Alternatively, a subset of news outlets and focus entities can be selected

[2] `https://www.ambiverse.com/`

by the user for a contrastive analysis between outlets, or context terms can be employed as additional input to quantify *how* a given news outlet reports about a specific group of entities.

5 EVALUATION

To demonstrate the validity of our model beyond exploration, we evaluate the streaming and static approach on a set of news events.

5.1 Event Completion Task

We evaluate against LOAD as the only comparable implicit network model. The event completion task can be defined as follows: Given $k-1$ out of k entities participating in an event, predict the remaining entity based on the data. We briefly describe the scheme used by LOAD for this task, before we present our improved version that includes the context. Both schemes rank entities x in the target set $X \in \{L, O, A, D\}$ based on a set of query entities $Q \subseteq L \cup O \cup A \cup D$.

LOAD Ranking (Baseline). This scheme applies a *tf-idf*-like scoring to the edges of the graph to rank entities $x \in X$ based on a query entity q by computing a normalized importance score r_L as

$$r_L(x|q) := \left(\log \frac{|Q|}{|\mathcal{N}(x) \cap Q|}\right) \sum_{e=(x,q,\cdot)\in E} \exp(-\delta(e)) \qquad (11)$$

Note that this scheme aggregates *all* parallel edges into a single edge, which is weighted by the sum of individual edge importances. Thus, the context is lost in this step. To generate a ranking based on multiple input entities, LOAD sums over the contributions of individual query entities. As an added feature, it includes the notion of *coherence*, requiring that each target entity is linked to at least $\min\{coh, |Q|\}$ query entities, with $coh = 2$ as a suggested value [30].

Context-based Ranking. In the context-sensitive model, two entities may be connected by more than one edge, which we use to differentiate between target candidates. Let $E_a(x)$ denote this set of aggregated edges between a query entity q and an entity x in the target set. Furthermore, where available, we can include the context of the event description in the query to match the context of candidate entities. Let $\kappa(q)$ denote the context of query entities in the event, which we include in the improved ranking

$$r_C(x|q) := \max_{a\in E_a(x)} \left[\mathrm{sim}(\kappa(a), \kappa(q)) \left(\log \frac{|Q|}{|\mathcal{N}(x) \cap Q|}\right) \chi(a)\right] \qquad (12)$$

Intuitively, we normalize the importance of a candidate with the similarity *sim* to the query context, before using the best contribution as a ranking score. While any suitable vector similarity function can be used, we use the cosine similarity in the following since it works well for vector embeddings. To obtain a ranking by multiple query entities, we improve the notion of coherence and rank candidates first by the number of neighbours in the query set $|\mathcal{N}(x) \cap Q|$, and break ties by the sum of ranking scores r_C.

5.2 Evaluation Setup

Static Clustering. We require a clustering algorithm without fixed clusters since it is impossible to divine a reasonable number of clusters that applies equally to all pairs of nodes. Thus, we select DBSCAN [10] with cosine as a distance measure. To obtain the necessary parameters ϵ and *minPts*, we conduct a number of preparatory tests and find that the result quality suffers for high values of *minPts*, while $minPts = 5$ works well. Since a value of $minPts > |E_a|$ would be meaningless for edge aggregation, we use the scheme $minPts = \min\{5, \frac{|E_a|}{5}\}$, which performs best in our experiments. We then employ the min-points heuristic to obtain a reasonable value of $\epsilon = 0.3$ as a starting point for the evaluation.

Table 2: Evaluation results of the streaming edge aggregation. Shown is the precision@1 for the complete context embedding as well as a context derived only from verbs.

	aggregation threshold			
	$t = 0.3$	$t = 0.4$	$t = 0.5$	$t = 0.6$
complete	0.218	0.218	0.232	**0.253**
verb	0.225	0.222	0.215	0.208
LOAD	0.157			

Table 3: Performance comparison of the static and streaming ($t = 0.6$) edge aggregation approaches on a subset of the evaluation data. We show the correct predictions at rank one (cor@1), precision@1, and recall.

	LOAD	stream aggr.		static clustering		
		complete	verb	$\epsilon = 0.2$	$\epsilon = 0.3$	$\epsilon = 0.4$
cor@1	44	71	61	35	27	25
prc@1	0.165	**0.266**	0.228	0.131	0.101	0.094
recall	0.655	0.955	0.955	0.955	0.955	0.955

Context Extraction Schemes. To derive contexts for entity cooccurrences, we consider two schemes according to the definition in Section 3.2. For the *complete* context, we use the weighted average embedding of all non-stopwords inside the context window. Based on the importance of verbs for traditional event extraction, we also consider the *verb* context, for which we utilize only the embeddings of verbs inside the context window (we exclude all forms of the auxiliary verbs *be* and *have*). Both schemes are applied separately during network and ground truth construction.

Ground Truth Data. To extract ground truth events, we use the Wikipedia Current Events portal[3], which contains manually maintained summarizations of news events. We crawl the pages for the months of June 2016 to November 2016 to extract each item as a news event. For NER and disambiguation, we use Wikipedia links in the text. Since the Wikipedia summaries contain references to news article sources, we match the references to articles in our input stream. We exclude all events that consist of less than two entities or have no reference to an article in our network. We obtain 97 individual events that correspond to at least one article in our collection. For each such event, we generate a query from each contained entity by using the remaining entities as query input and the removed entity as ground truth (i.e., an event with k entities induces $k(k-1)$ queries). We manually annotate the verbs in these event summaries. In total, we obtain 293 queries for the evaluation.

5.3 Evaluation Results

Each evaluation query has exactly one correct answer. Therefore, suitable evaluation metrics are precision@1, i.e., the fraction of queries in which the top ranked prediction is correct, and recall@k, i.e., the number of correct predictions among the top k predictions.

[3] https://en.wikipedia.org/wiki/Portal:Current_events

Streaming Aggregation. We first compare the two approaches for context generation over varying aggregation thresholds and show the resulting precision in Table 2. Threshold values of $t < 0.3$ are omitted since no further changes occur. Both methods outperform the LOAD baseline by a large margin (up to 61% improvement). The verb context aggregation shows a slight decline in performance with increasing threshold. The precision of the complete context increases with the threshold value and it performs better overall. In Figure 3 (top), we show the corresponding recall values of the complete context approach. Varying thresholds show little influence on recall, which makes low thresholds attractive in settings where a compact representation is important and recall@5 is sufficient.

Static Aggregation. In Table 3, we show the performance of the static aggregation for a subset of 267 evaluation queries (the remaining 26 clusterings did not finish within 48 hours). Due to this smaller evaluation set, the values for the static aggregation vary slightly. For some of the ϵ settings, the clustering performs better than the LOAD baseline, but not by a large margin, and higher values of ϵ decrease the performance. The recall values shown in Figure 3 support this observation. While static aggregation outperforms LOAD, it does not rival the streaming aggregation.

In summary, we find that streaming edge aggregation is superior to static aggregation in this setting. While other clustering algorithms may perform better, our tests were extensive and the ease of use for the streaming method is much higher. While the optimal parameter settings for clustering approaches are usually difficult to obtain, in the streaming approach there is a direct correlation between the threshold and the prediction quality. What remains is the issue of performance depending on the threshold selection, which we discuss in the following.

5.4 Edge Deflation in Streaming Aggregation

The streaming model is designed to reduce the number of aggregated edges that are stored in the graph to a manageable size and avoid redundancy. Especially for entangled news streams, many parallel edges with highly similar context are to be expected. In Figure 4, we show the number of aggregated edges as a function of the number of unaggregated edges for different threshold values applied to the complete context embeddings. We find that for thresholds $t \leq 0.3$, aggregation is almost complete and there are never more than three parallel edges. For higher thresholds, this number increases but is still easily manageable. Since higher thresholds are favorable with regard to the extraction of information from the graph, the threshold thus has to be tuned to the data throughput in an application scenario. For the real-time processing of streams of news articles, this is unproblematic due to the relatively low volume of documents in the news domain. For higher frequency streams such as the entire blogosphere, more sophisticated data structures or similarity approximations may be desirable.

6 CONCLUSION & ONGOING WORK

In this paper, we discussed the problem of entity-centric explorations of large entangled streams of news articles. Based on the intuition that entity mentions can serve as stitching points between potentially biased news streams and as focal points of news retrieval tasks, we introduced contextual implicit entity networks as

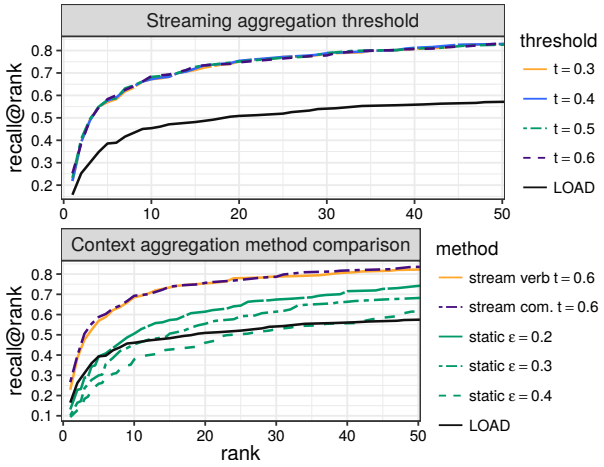

Figure 3: Recall of the edge aggregation methods for the event completion task. Top: values for different thresholds of the streaming aggregation approach with the complete context. Bottom: Comparison of the recall of the static aggregation DBSCAN clustering for $minPts = 5$ and varying ϵ.

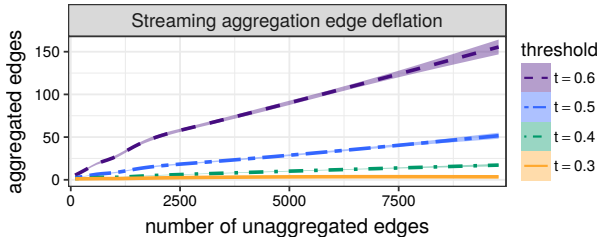

Figure 4: Deflation of parallel edges for streaming aggregation with complete context as a linear fit to the average number of edges after aggregation with a 3σ confidence interval.

a comprehensive and versatile tool for the representation of such entangled news streams. The model can be constructed faster than the publication speed of news articles by several orders of magnitude and thus efficiently facilitates a multitude of subsequent entity-centric information retrieval tasks from the underlying streams in near real-time, such as topic and event extraction and tracking, contextual search, descriptive sentence extraction, or document retrieval. Furthermore, it supports the interactive contrastive exploration and contextual aggregation of news published by multiple news outlets as well as their change over time. We evaluated the model's performance for different parameter settings on a large collection of news streams, and found that the streaming aggregation approach outperforms existing alternatives for the task of entity-centric event completion. Finally, we discussed an application of the model to the extraction of contextual and entity-centric topic detection and tracking as one example of news exploration in entangled news streams. Our implementation of the model along with all used data is available for further studies.

Ongoing Work. We are currently researching the extraction of evolving document topics from entity-centric topics, in a step towards the comparison of contents between news streams. Furthermore, we are working on a generalization of the model to settings with more versatile requirements for (named) entity annotations.

Acknowledgements. The authors would like to thank Satya Almasian for her assistance in preparing the ground truth data, and the Ambiverse Ambinauts for kindly providing access to their named entity linking and disambiguation API.

REFERENCES

[1] Amr Ahmed, Qirong Ho, Jacob Eisenstein, Eric P. Xing, Alexander J. Smola, and Choon Hui Teo. 2011. Unified Analysis of Streaming News. In *WWW*. https://doi.org/10.1145/1963405.1963445

[2] James Allan. 2012. *Topic Detection and Tracking: Event-based Information Organization*. Vol. 12. Springer. https://doi.org/10.1007/978-1-4615-0933-2

[3] James Allan, Ron Papka, and Victor Lavrenko. 1998. On-Line New Event Detection and Tracking. In *SIGIR*. https://doi.org/10.1145/290941.290954

[4] Foteini Alvanaki, Sebastian Michel, Krithi Ramamritham, and Gerhard Weikum. 2012. See What's enBlogue: Real-time Emergent Topic Identification in Social Media. In *EDBT*. https://doi.org/10.1145/2247596.2247636

[5] Martin Atkinson and Erik Van der Goot. 2009. Near Real Time Information Mining in Multilingual News. In *WWW*. 1153–1154. https://doi.org/10.1145/1526709.1526903

[6] Roi Blanco and Christina Lioma. 2012. Graph-based Term Weighting for Information Retrieval. *Inf. Retr.* 15, 1 (2012), 54–92. https://doi.org/10.1007/s10791-011-9172-x

[7] David M Blei. 2012. Probabilistic Topic Models. *Commun. ACM* 55, 4 (2012), 77–84. https://doi.org/10.1145/2133806.2133826

[8] James P. Callan. 1996. Document Filtering With Inference Networks. In *SIGIR*. https://doi.org/10.1145/243199.243273

[9] Andrea Ceroni, Ujwal Gadiraju, Jan Matschke, Simon Wingert, and Marco Fisichella. 2016. Where the Event Lies: Predicting Event Occurrence in Textual Documents. In *SIGIR*. https://doi.org/10.1145/2911451.2911452

[10] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *KDD*. http://www.aaai.org/Library/KDD/1996/kdd96-037.php

[11] Ao Feng and James Allan. 2007. Finding and Linking Incidents in News. In *CIKM*. https://doi.org/10.1145/1321440.1321554

[12] Dhruv Gupta and Klaus Berberich. 2014. Identifying Time Intervals of Interest to Queries. In *CIKM*. https://doi.org/10.1145/2661829.2661927

[13] Dhruv Gupta, Jannik Strötgen, and Klaus Berberich. 2016. EventMiner: Mining Events from Annotated Documents. In *ICTIR*. https://doi.org/10.1145/2970398.2970411

[14] Johannes Hoffart, Fabian M. Suchanek, Klaus Berberich, Edwin Lewis-Kelham, Gerard de Melo, and Gerhard Weikum. 2011. YAGO2: Exploring and Querying World Knowledge in Time, Space, Context, and many Languages. In *WWW Companion*. https://doi.org/10.1145/1963192.1963296

[15] Nattiya Kanhabua, Sara Romano, and Avaré Stewart. 2012. Identifying Relevant Temporal Expressions for Real-world Events. In *TAIA@SIGIR*.

[16] Andrey Kutuzov and Elizaveta Kuzmenko. 2016. Cross-Lingual Trends Detection for Named Entities in News Texts with Dynamic Neural Embedding Models. In *NewsIR@ECIR*. http://ceur-ws.org/Vol-1568/paper5.pdf

[17] Levon Lloyd, Dimitrios Kechagias, and Steven Skiena. 2005. Lydia: A System for Large-Scale News Analysis. In *SPIRE*. https://doi.org/10.1007/11575832_18

[18] Farzaneh Mahdisoltani, Joanna Biega, and Fabian M. Suchanek. 2015. YAGO3: A Knowledge Base from Multilingual Wikipedias. In *CIDR*. http://cidrdb.org/cidr2015/Papers/CIDR15_Paper1.pdf

[19] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *NIPS*. http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality

[20] Anne-Lyse Minard, Manuela Speranza, Eneko Agirre, Itziar Aldabe, Marieke van Erp, Bernardo Magnini, German Rigau, and Ruben Urizar. 2015. SemEval-2015 Task 4: TimeLine: Cross-Document Event Ordering. In *SemEval@NAACL-HLT*. http://aclweb.org/anthology/S/S15/S15-2132.pdf

[21] Arunav Mishra and Klaus Berberich. 2016. Event Digest: A Holistic View on Past Events. In *SIGIR*. https://doi.org/10.1145/2911451.2911526

[22] Sean Moran, Richard McCreadie, Craig Macdonald, and Iadh Ounis. 2016. Enhancing First Story Detection using Word Embeddings. In *SIGIR*. https://doi.org/10.1145/2911451.2914719

[23] Sasa Petrovic, Miles Osborne, and Victor Lavrenko. 2010. Streaming First Story Detection with Application to Twitter. In *NAACL-HLT*. http://www.aclweb.org/anthology/N10-1021

[24] Martin F. Porter. 1980. An Algorithm for Suffix Stripping. *Program* 14, 3 (1980), 130–137. https://doi.org/10.1108/eb046814

[25] François Rousseau and Michalis Vazirgiannis. 2013. Graph-of-word and TW-IDF: New Approach to Ad Hoc IR. In *CIKM*. https://doi.org/10.1145/2505515.2505671

[26] Anish Das Sarma, Alpa Jain, and Cong Yu. 2011. Dynamic Relationship and Event Discovery. In *WSDM*. https://doi.org/10.1145/1935826.1935867

[27] Andreas Spitz, Satya Almasian, and Michael Gertz. 2017. EVELIN: Exploration of Event and Entity Links in Implicit Networks. In *WWW Companion*. https://doi.org/10.1145/3041021.3054721

[28] Andreas Spitz, Vaibhav Dixit, Ludwig Richter, Michael Gertz, and Johanna Geiß. 2016. State of the Union: A Data Consumer's Perspective on Wikidata and Its Properties for the Classification and Resolution of Entities. In *Wiki@ICWSM*. http://aaai.org/ocs/index.php/ICWSM/ICWSM16/paper/view/13200

[29] Andreas Spitz and Michael Gertz. 2015. Breaking the News: Extracting the Sparse Citation Network Backbone of Online News Articles. In *ASONAM*. https://doi.org/10.1145/2808797.2809380

[30] Andreas Spitz and Michael Gertz. 2016. Terms over LOAD: Leveraging Named Entities for Cross-Document Extraction and Summarization of Events. https://doi.org/10.1145/2911451.2911529

[31] Jannik Strötgen and Michael Gertz. 2013. Multilingual and Cross-domain Temporal Tagging. *Language Resources and Evaluation* 47, 2 (2013), 269–298. https://doi.org/10.1007/s10579-012-9179-y

[32] Chenhao Tan, Adrien Friggeri, and Lada A. Adamic. 2016. Lost in Propagation? Unfolding News Cycles from the Source. In *ICWSM*. http://www.aaai.org/ocs/index.php/ICWSM/ICWSM16/paper/view/13011

[33] Benjamin E. Teitler, Michael D. Lieberman, Daniele Panozzo, Jagan Sankaranarayanan, Hanan Samet, and Jon Sperling. 2008. NewsStand: A New View on News. In *ACM-GIS*. https://doi.org/10.1145/1463434.1463458

[34] Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In *NAACL-HLT*. http://aclweb.org/anthology/N/N03/N03-1033.pdf

[35] Jeroen B. P. Vuurens, Arjen P. de Vries, Roi Blanco, and Peter Mika. 2015. Online News Tracking for Ad-Hoc Queries. In *SIGIR*. https://doi.org/10.1145/2766462.2767872

[36] Yiming Yang, Thomas Pierce, and Jaime G. Carbonell. 1998. A Study of Retrospective and On-Line Event Detection. In *SIGIR*. https://doi.org/10.1145/290941.290953

[37] Yiming Yang, Jian Zhang, Jaime G. Carbonell, and Chun Jin. 2002. Topic-conditioned Novelty Detection. In *KDD*. https://doi.org/10.1145/775047.775150

[38] Kaiqi Zhao, Lisi Chen, and Gao Cong. 2016. Topic Exploration in Spatio-Temporal Document Collections. In *SIGMOD*. https://doi.org/10.1145/2882903.2882921