Computers in Industry 123 (2020) 103321

Contents lists available at ScienceDirect

Computers in Industry

journal homepage: www.elsevier.com/locate/compind

A knowledge-graph platform for newsrooms

Arne Berven[a], Ole A. Christensen[b], Sindre Moldeklev[b], Andreas L. Opdahl[b,*], Kjetil J. Villanger[b]

[a] Wolftech Broadcast Solutions, Nøstegaten 72, N-5011 Bergen, Norway
[b] University of Bergen, P.O. Box 7802, N-5020 Bergen, Norway

ARTICLE INFO

Article history:
Received 31 March 2020
Received in revised form 17 August 2020
Accepted 8 September 2020

Keywords:
Computational journalism
Journalistic knowledge platforms
Newsroom systems
Knowledge graphs
Semantic technologies
RDF
OWL
Ontology
Natural-language processing (NLP)
Machine learning (ML)

ABSTRACT

Journalism is challenged by digitalisation and social media, resulting in lower subscription numbers and reduced advertising income. Information and communication techniques (ICT) offer new opportunities. Our research group is collaborating with a software developer of news production tools for the international market to explore how social, open, and other data sources can be leveraged for journalistic purposes. We have developed an architecture and prototype called News Hunter that uses knowledge graphs, natural-language processing (NLP), and machine learning (ML) together to support journalists. Our focus is on combining existing data sources and computation and storage techniques into a flexible architecture for news journalism. The paper presents News Hunter along with plans and possibilities for future work.

1. Introduction

Journalism is in crisis, but information and communication technologies (ICT) offer new opportunities (Ekdale et al., 2015; Machill and Beiler, 2009). Journalists today have access to a wealth of digital information from news aggregators, social media, and open data providers in addition to traditional sources (Diakopoulos, 2017; Heravi and McGinnis, 2015). Computational journalism is "the advanced application of computing, algorithms, and automation to the gathering, evaluation, composition, presentation, and distribution of news." (Thurman, 2019). News platforms, or *journalistic knowledge platforms,* support computational journalism through integrated content and software components that support all stages of news production, thereby creating "a competitive advantage by enabling scale and reducing marginal costs in the gathering, production, and dissemination of their content" (Diakopoulos, 2017). For example, journalistic knowledge platforms can use knowledge graphs (Singhal, 2012) and other semantic technologies (Allemang and Hendler, 2011; Berners-Lee et al., 2001) to automatically harvest, analyse, enrich, organise, and prepare potentially news-related information with increasing semantic precision (Leban et al., 2014; Vossen et al., 2016). They can also leverage theories and techniques from artificial intelligence (Russel and Norvig, 2009) and, in particular, from machine learning (Goodfellow et al., 2016; Müller and Guido, 2016), and natural-language processing (Mikolov et al., 2013; Mitchell et al., 2018), to classify, label, cluster, detect events, and otherwise process journalistic information in new and meaningful ways (Latar, 2015; Miroshnichenko, 2018).

News Hunter is an architecture and a series of proof-of-concept prototypes that: harvests potentially news-related texts and social-media messages from the net; analyses and represents them semantically in a knowledge graph; classifies, clusters, and labels them; enriches them with background information; and presents them in real time to journalists who are working on related reports or as tips about new events. For this purpose, News Hunter combines knowledge graphs and other semantic technologies with techniques from artificial intelligence, such as machine learning, and natural-language understanding.

News Hunter is being developed in collaboration between a university research group and a software developer of news production tools for the international market. Our research goal is to understand whether and how topical information and communication techniques (ICTs) can be leveraged to make social, open, and other data sources more readily available for journalistic work. Our

* Corresponding author.
E-mail address: andreas.opdahl@uib.no (A.L. Opdahl).

https://doi.org/10.1016/j.compind.2020.103321

industrial goal is to develop and evaluate proof-of-concept prototypes of such a journalistic knowledge platform. These two goals mutually reinforce one another, because work on the research goal supplies theories and ideas for development, whereas work on the industrial goal returns working prototypes and evaluations of the research.

The paper starts with the following research question: how can existing heterogeneous data sources and processing and storage techniques be combined to support news journalism? We first review the most similar existing systems described in the literature. We then outline our research and development approach, before we present and evaluate the News Hunter architecture along with its proof-of-concept prototype and components. We also compare News Hunter with the existing systems and discuss future plans and possibilities.

2. Background

Neptuno (Castells et al., 2004) is an early semantic newspaper archive system that aims to give archivists and reporters more expressive ways to describe and annotate news materials and to give reporters and readers better search and browsing capabilities. It uses an ontology for classifying archive content along with modules for semantic search, browsing and visualisation of content, but is limited to organising and disseminating already published news.

News Engine Web Services (NEWS) (Fernández et al., 2006) is another early example of a journalistic knowledge platform, which uses natural-language processing (NLP) techniques in combination with a domain-specific ontology (Fernández et al., 2006; Fernandez et al., 2010; García et al., 2006; Garcia et al., 2008) to annotate news reports and photography descriptions precisely in order to streamline news retrieval, production, and dissemination. In contrast to our platform, NEWS is limited to semantic annotation of existing news items. It represents metadata as RDFS and uses concepts from SUMO/MILO, but does not build a knowledge graph for detecting and representing news events.

Global Database of Events, Language, and Tone (GDELT) is a project and a live web service[1] that monitors the world's news media in over 100 languages and provides a real-time open data global graph of news items and events that is updated every 15 min. GDELT uses newspaper texts and TV and radio news as input, and aims to increasingly analyse social-media sources as well. The data are provided in tables that can be transformed into a graph, but not all the data are disambiguated with IRIs that point into the LOD cloud. Also, GDELT's primary focus is on conflicts, not general news, and it represents events as actor1-action-actor2 triplets that are more fine-grained than news events.

EventRegistry (Leban et al., 2014) is a live web site that continuously harvests news items from RSS feeds in many languages; uses named-entity recognition and wikification to represent the items as RDF in a knowledge graph; and groups the items according to event (a significant happening reported several times). Each group is then categorised according to the DMOZ taxonomy; linked to related information about locations, involved people, and organisations; and used to track events and trending topics in real time. In contrast to News Hunter, EventRegistry only monitors events that are already reported in RSS streams and, although it uses RDF, linking is limited to Wikipedia articles and DMOZ categories.

NewsReader (Vossen et al., 2016; Rospocher et al., 2016) analyses web-news texts written in four different languages semantically and enriches them with information from general and linguistic reference knowledge bases in order to build event-centric knowledge graphs (ECKG). ECKGs are defined by an OWL ontology and describe events in terms of what has happened, who was involved, and where and when it took place. Like EventRegistry, NewsReader differs from our platform by only considering published news reports.

Reuters' *Tracer* (Liu et al., 2017) harvests and analyses tweets in real time to detect new events and trending topics. Tracer can thereby detect newsworthy events several minutes before they are reported by news media (Liu et al., 2017). However, Reuters Tracer does not represent messages and news events semantically in knowledge graphs. It is limited to harvesting tweets and thus focusses on fine-grained events.

Scalable Understanding of Multilingual MediA (SUMMA) (Germann et al., 2018) records, transcribes, and translates multimedia news items in six languages, driven by the need to support data journalism in organisations like BBC and Deutsche Welle (DW). It then employs NLP and ML techniques to extract named entities, events, and topics and to summarise emerging news clusters. Like GDELT, SUMMA deals with multimedia news. It represents metadata as RDFS and uses labels defined in SUMO/MILO, but does not link externally to the LOD cloud nor build a knowledge graph for detecting and representing news events. Also unlike our platform, it focusses on organising and summarising already reported news, rather than on detecting new events.

Acquisition de Schémas pour la Reconnaissance et l'Annotation d'Événements Liés (ASRAEL) (Rudnik et al., 2019) harvests English and French news texts; annotates them semantically with items and statements from Wikidata; and links them to Wikidata events using IPTC's rNews vocabulary for metadata annotation. The resulting knowledge graph can be queried using SPARQL. Unlike our platform, the focus is on contextualising and enriching existing news items and ASRAEL can only identify events that have already been represented in Wikidata.

Each existing system demonstrates new information services that can be highly useful for journalists and in newsrooms, but none of them yet combine the full strength of externally linked knowledge graphs and ontologies to harvest both pre-news messages from social media and already published news reports. Dealing with *both pre- and post-news information* in the same platform would improve support for central tasks, such as identifying which social-media events that are really pre news and following up how unfolding news events spread and perhaps develop further in social media. Also, the literature on existing systems does little to address *architecture for journalistic knowledge platforms* and, when it is mentioned at all, it is limited to simpler processing pipelines. In light of this, we sharpen our starting research question as follows: how can *pre-and post-news* data sources and processing and storage techniques be combined into a *flexible architecture* for news journalism? In the following sections, we will explain how our News Hunter architecture and prototype contribute to filling this gap and to advance research on journalistic knowledge platforms.

3. Research approach

3.1. Collaborators

Wolftech Broadcast Solutions is a software company that develops integrated news systems for making live news production simpler and more efficient. Wolftech News is their system for effective news production with current focus on television news. It aims to help journalists and other newsroom workers collaborate effectively and efficiently on creating, managing, and publishing media to a variety of platforms. It supports and improves the workflows in a newsroom by integrating mobile solutions for fieldwork with central systems for news monitoring, resource management, news

[1] http://gdeltproject.org

editing, and multi-platform publishing (on live and internet TV, web, social media, etc.).

The group for Intelligent Information Systems (I2S) at the University of Bergen studies information systems (IS) that employ artificial intelligence (AI) related techniques, such as knowledge graphs (Singhal, 2012; Allemang and Hendler, 2011), machine learning (Goodfellow et al., 2016; Müller and Guido, 2016), and natural-language understanding (Mikolov et al., 2013; Mitchell et al., 2018).

News Hunter (Opdahl et al., 2016; Berven et al., 2018) is a collaboration between Wolftech and I2S that aims to extend Wolftech News to harvest, organise and leverage social media streams and other big-data sources for journalistic and newsroom purposes, using topical ICTs such as knowledge graphs, machine learning, and natural-language processing.

3.2. Method

Because our research involves explorative technology development, we have organised and reported it as design science (Hevner et al., 2004; Hevner, 2007), where the aim is to advance theory and improve practice on journalistic knowledge platforms by incrementally developing and evaluating two research artefacts: an *architecture* (a high-level structure of system components) and a *prototype* that instantiates the architecture as a proof-of-concept (Vaishnavi et al., 2004).

Following Hevner's three-cycle model of design-science research (Hevner, 2007), we synchronise research and development iterations in continuous dialogue with Wolftech, in order to get feedback on results and ideas for new features. For each iteration, we attempt to define clear goals for both research and development, based on an explicit development process that involve selected tools and technologies and where each step produces well-defined and verifiable results. We seek short, XP-like (Beck, 1999) iterations, typically of 1–3 weeks' length, inspired by the minimum viable product (MVP) idea: build the minimum number of features that is required for the platform to work as intended, and then evolve from there. We align with technologies that Wolftech already use in their development and runtime environments.

Throughout development, we use proof-of-concept evaluations after each major development iteration (typically every 1–3 weeks) along with component evaluations to gauge the quality of the most central components in our architecture. Section 5 will present evaluations of central prototype features with human participants.

3.3. Process

The first simple News Hunter prototype harvested posts from Facebook's public API and ran non-English posts through Google's Translate API. The English texts were then analysed using IBM's online Alchemy API (today part of the IBM cloud) to extract metadata about topics, named entities, and sentiments. The Alchemy metadata, along with the message itself and additional metadata from Facebook's API (date, title, location, etc.), were then loaded into a graph database (triple store). The first prototype was written in C# as an ASP.NET application using BrightstarDB (a graph database/triple store for the. NET platform) as triple store. Having demonstrated the viability of our idea, we then started developing a more elaborate and clearly architected, second News Hunter prototype from scratch, which the rest this paper will present and discuss.

4. Architecture

Fig. 1 places News Hunter in its usage context, whereas Fig. 2 shows the News Hunter architecture with access and serving relations between its components. The prototype reported in this paper implements central functionality of all these components, although a few of them only in rudimentary form. This section and the next present and discuss the overall architecture and the more developed prototype components in further detail.

4.1. Harvesters

The harvesters continuously download potentially news-related information items (articles, messages, posts, tweets...) from relevant sources such as Facebook and Twitter, RSS feeds, and major English-, Norwegian- and Spanish-language news sites. The harvesters are implemented as Python scripts that use the Tweepy library to harvest Tweets, the Feedparser library for RSS, and the Newspaper library to harvest news reports. The pre-processed news items are stored as JSON objects in a source database.

Defining harvesters as a separate type of component in the architecture makes it easier to add new harvesters to future version of the platform, for example to retrieve information from new information sources or to incorporate external harvesting services and components. Commercial harvesting components are already becoming available in the form of *news aggregators* such as datahub.io and newsapi.org. Running multiple harvester components independently in parallel also makes the platform more resilient to changed or unavailable information sources.

4.2. Source DB

The source database stores the original items to make the raw texts available for later retrieval and further analysis. The source database is implemented using Elasticsearch with a Python script that inserts harvested item texts and selected metadata. The source DB also merges duplicates, which is important when multiple harvesters are run in parallel.

Including a source DB as a component in the architecture makes it possible to re-analyse old news items whenever a lifter has been introduced or improved. It also facilitates graded lifting, where seemingly less important items, for example from social media, can first be lifted coarsely and quickly and then re-analysed more deeply later if they turn out to be important. Storing the original texts also makes them available for other NLP techniques should the need arise, and it ensures that a news item remains accessible even if it is later removed by the publisher, for example because of censorship.

4.3. Translator

When necessary, the translator creates canonical-language translations of news items in the source database. The prototype uses English as canonical language. The translator relies on Microsoft's online Translate API, because the standard libraries we have explored do not support small languages like Norwegian.

Including a source translator in the architecture makes it possible to use lifting techniques developed and tuned for richly-resourced languages such as English to lift news items from smaller languages for which NLP pipelines and training materials are less developed or missing completely. Storing translations of texts into the same canonical language also simplifies cross-language retrieval.

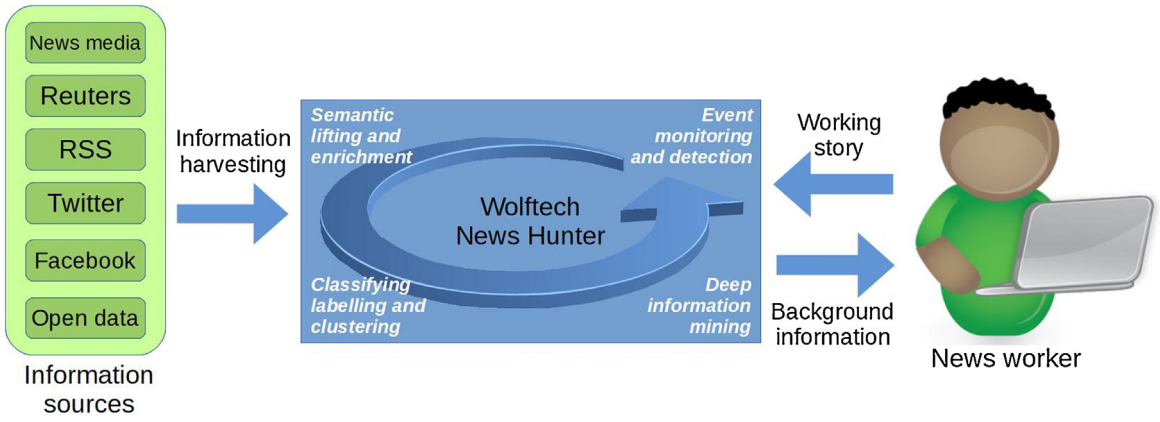

Fig. 1. News Hunter harvests information from social, open, and other sources and presents it as relevant background information to working journalists (Berven et al., 2018).

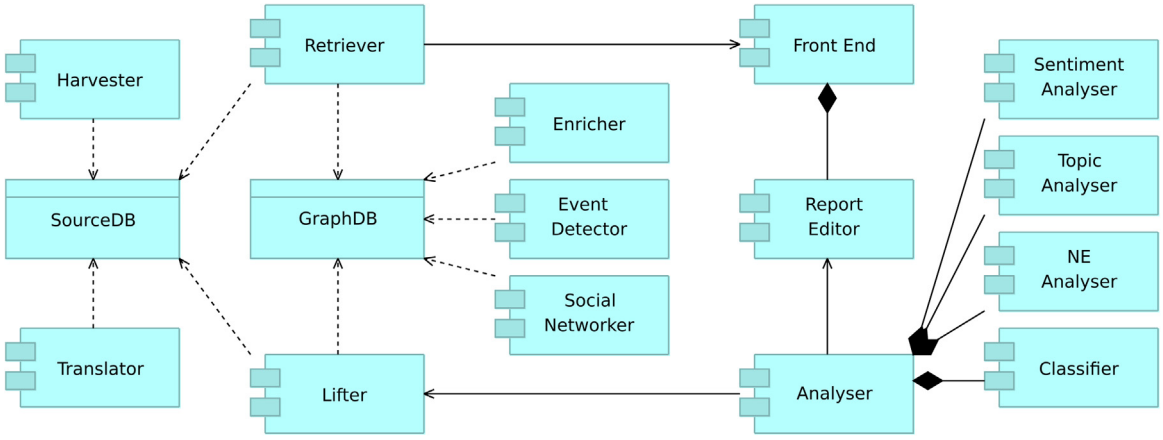

Fig. 2. The News Hunter architecture with access and serving relations between its components.

4.4. Lifter and analysers

The lifter runs the harvested (and perhaps translated) news items through an NLP pipeline to represent them semantically as small knowledge graphs, which can be inserted into a graph database (triple store). The analyser realises the NLP pipeline and invokes sub-components for specific tasks such as sentiment analysis, topic analysis, named-entity (NE) analysis, and classification. The backbone of the analyser is implemented in C#, with most of the components written in Python.

4.4.1. Sentiment analyser

Sentiment analysis is implemented using the AFINN Python library.

4.4.2. Topic analyser

Unsupervised topic extraction is implemented using a variety of tools. Microtexts from Twitter and Facebook, are analysed using the RAKE (Rapid Automatic Keyword Extraction) library, written in C#. RSS items are handled using Textacy, a wrapper library for Spacy (see below). Longer texts are analysed with the Python-implementation of the TextRank library, which also supports automatic topic extraction along with report summarisation. Longer texts introduce a bias because they generate more keywords, making them appear more prominent than shorter texts when querying by topic. The analyser therefore weighs each keyword by its position in the text (because news texts should present its most central themes in the beginning) and number of occurrences, so that infrequent and late-occurring keywords weigh less.

4.4.3. Named-entity (NE) analyser

Part-of-speech (PoS) tagging and named-entity recognition (NER) uses the Python-library Spacy, whereas named-entity linking are done with DBpedia Spotlight (Mendes et al., 2011). Written in Scala, Spotlight tags named entities recognised in the text with DBpedia IRIs in order to provide more precise semantics and facilitate data enrichment with linked open data.

Separating the lifter into a separate component makes it available both downstream for lifting harvested news items and upstream for lifting the stories-being written by journalists. A clearly defined lifting API might also make it easier to incorporate external lifting tools and services seamlessly into the platform, such as those provided by FRED (Gangemi et al., 2017) and PIKES (Corcoglioniti et al., 2016). Furthermore, the lifting component can hide complexity, for example opaquely using different NLP pipelines for different types of news items or using an ensemble of different analysers for the same item, and it makes it easier to add new or improved analysers.

4.5. Graph DB

The graph database (or triple store) combines the resulting semantic representations of news items into a persistent, contiguous journalistic knowledge graph, or news graph. It also stores the titles and first paragraphs of each news item in its original language and canonical translation. The graph database is implemented as a BrightstarDB triple store, which is a DBMS specifically for RDF graphs (Allemang and Hendler, 2011). To upload and query data, we use the Microsoft Entity Framework along with LINQ (Language

Integrated Query), a .NET component for C#-native data querying of domain-specific models that automatically translates datatype-specific queries to SPARQL for processing by BrightstarDB.

The graph database is a central component in the architecture because it is the output target and input source for several other components. Most importantly, it stores the output from the lifter and it provides the information needed by the retriever and thus, indirectly, by the front end in Fig. 2. In addition, it is continuously monitored by the enricher, event detector and social networker components, which thereby become decoupled from one another and from the other components. Defining a single interface to the triple store also ensures scalability as the knowledge graph grows, because the data can be split thematically, chronologically, regionally, or otherwise into several more manageable knowledge graphs without changing other components.

4.6. Ontology

The ontology describes how lifted news items are represented semantically. A clearly defined ontology is crucial to support the graph database as the central knowledge repository in the architecture. It must therefore define all the classes and properties necessary for representing the output of the lifter, as well as the classes and properties needed by the retriever and thus, indirectly, by the front end. As shown in Fig. 3, *Items* are potentially news-relevant harvested items, with *Item Types* such as news article, report-in-writing, RSS item, blog post, and microtext (Facebook message or tweet). *Descriptors* represent the semantic contents of items, with relations to *namedEntities*, *topics*, *locations*, and *sentiments* that are mentioned or expressed in the text. Topics and named entities (such as people, organisations, and places) are *EnrichableResources* with a unique IRI, meaning that they can easily be enriched with additional triples available in the LOD cloud from sources such as DBpedia.

The ontology has been explicitly defined using the Web Ontology Language (OWL), making the graph database interface more clearly defined and making it easier to enrich the knowledge graph with linked-open data from other ontologies. It also facilitates automated reasoning to ensure that the ontology remains consistent. Because the ontology follows good linked-open data practices, such as reusing and interlinking terms from existing ontologies, it can easily be extended with terms from other popular ontologies, for example to represent the source IRIs and creation dates of items, social relations between persons, and semantic relations between topics.

4.7. Classifiers

The classifiers organise the harvested and lifted news items further, using several NLP and ML pipelines.

4.7.1. Single-label classifier

In order to provide additional entry paths into the knowledge graph, NLP and ML techniques are used to classify (label) news items. A single-label classifier is implemented as a pipeline that uses Spacy for part-of-speech (PoS) tagging, stop-word removal, and lemmatisation. scikit-learn is then used to create a term-document matrix and for feature selection, assigning higher weights to potentially important words in a text using TF-IDF. The final classification step uses both a support-vector machine (SVM) implemented using scikit-learn and a multi-layer perceptron (MLP) neural network implemented using Keras. Out of 2225 pre-labelled RSS items from BBC's Insight dataset, 70 % were used for training and 30 % held out for validation (Müller and Guido, 2016).

4.7.2. Multi-label classifier

Multi-label classification relaxes the single-label requirement and has the potential to represent news content even more flexibly and precisely, in particular when combined with standard news taxonomies such as the IPTC Media Topics. Multi-label classification is implemented by the same pipeline used for single-label classification. Out of 544 820 pre-labelled news articles from The Guardian's public API, 70 % were used for training and 30 % held out for validation. String equivalence supported by WordNet synsets (Miller, 1995) were used to match the Guardian labels to IPTC news codes.

Similarly to the lifter component, a classification component with a clearly defined API makes it easier to incorporate external classification tools and services into the platform. Many primary and secondary news sources already label the news items they provide, sometimes even with semantically disambiguated IRIs. Defining single- and multi-label classifiers as sub-components inside the classifier makes it easier to run several alternative components in parallel as part of ensembles, and it makes it easier to add new or improved specific classification components.

4.8. Event detection

The event detector clusters news items according to named entities, topics, and location in order to identify potentially newsworthy events. Multiple near-simultaneous news items related to the same entity (including places) and/or topic suggests that a potentially newsworthy event is unfolding. The event detector is implemented by clustering news items represented in the knowledge graph, selecting items from recent days and pre-processing them to calculate TF-IDFs. Scikit-learn's DBSCAN algorithm is then used for clustering, because it offers scalability and focus on neighbourhood size at the expense of uneven cluster sizes.

Defining event detection as a separate component type makes it possible to develop specialised event detectors running in parallel for different event types and makes it easier to introduce new detectors without changing other components.

4.9. Enricher

Enrichers augment the knowledge graph by inserting related information from external data sources. An example enricher component is implemented using pre-written SPARQL queries that use IRIs returned by DBpedia Spotlight to retrieve related background information on demand from the live DBpedia endpoint (Bizer et al., 2009a). Another example enricher is implemented using the Twitter API to retrieve tweets on demand from athletes represented in the knowledge graph. The front end can also retrieve additional background information on demand from social, open, and other data sources on the web.

Separating out the enricher as a separate type of component avoids redundancy, because specific enrichment tasks such as adding external LOD data about, for example, athletes is carried out in a single place in the architecture, instead of being replicated across several different components that all somehow deal with athletes. Instead of adding different types of information about athletes for different purposes in several different components, possibly creating both factual inconsistencies and terminology conflicts, all athlete information is thereby added in a coordinated manner by a single enricher component that can collaborate with other enricher, such as a more general person-information enricher. These enrichers can work independently of the lifters and other components, continuously scanning the input queues to the knowledge graph for new resources to enrich.

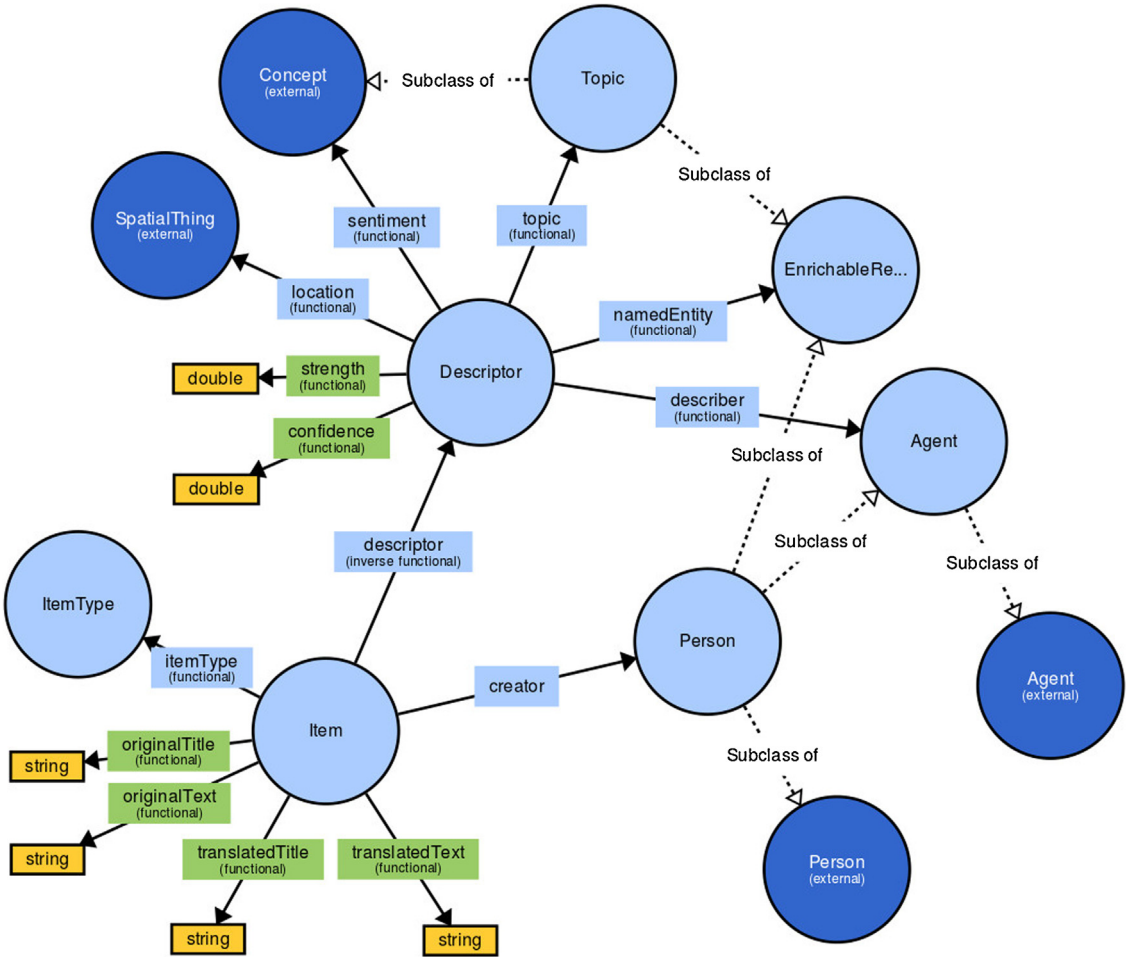

Fig. 3. The revised News Hunter ontology, adapted from (Berven et al., 2018).

4.10. Social networking

The social networker conducts affinity analysis to identify whether people in the knowledge graph are on friendly terms or not, a useful feature for journalists when selecting informants and planning interviews. Who-knows-who graphs help journalists prepare for interviews outside their usual areas, for example to avoid saying something wrong to the subjects. News Hunter also supports invitation of interview objects, giving journalists and other newsroom workers a standard form for creating invitations and saving invitations in the knowledge graph so they can be revised and reused in the future.

Similarly to the enricher, defining a specific social-networking component in the architecture makes it easier to incorporate existing social and other network analysis tools and services in the platform, and it avoids redundancy because all network analysis are carried out in a single place.

4.11. Report editor

The report editor lets the journalist type in a report which is analysed in real time. The report editor is implemented as a Froala WYSIWYG-editor plug-in. Whenever the journalist pauses writing, the text is sent asynchronously via the lifter to the same NLP pipeline that is used to analyse harvested news items. This has several benefits. The journalist gets instant feedback on the sentiment of their writing and under which category they should save and publish their reports. Other journalists in the same organisation that are working on similar new items can be identified to foster collaboration and avoid duplicate work. And the knowledge graph can be queried for relevant background information to present to the journalist. The latter is implemented using simple algorithms based on semantic distance, so that the relevance of a harvested new item is proportional to the number of related topics and named entities, weighted by the strength of each relation. Exact word-sense match is the strongest, followed by synonyms, hyper-/hyponyms, and then other semantic relations.

Defining the report editor as a distinct component with a clearly defined API is important because most news organisations will already have report editing tools in place, which should fit as seamlessly as possible into the News Hunter platform.

4.12. Front end

The front end embeds the report editor in a working environment. It receives the result of semantically analysing the report-in-writing and invokes the retriever component to present similar news items and other relevant background information to the journalist. The report editor is implemented as an interactive HTML page as shown in Fig. 4, developed using AngularJS, HTML and CSS, Sketch, and Marvel. The user can click on the identified topics and named entities to retrieve summaries of related news items. The front end also displays the most recent item cluster detected in the knowledge graph. The News Hunter prototype thus provides a full application stack, from the web-based GUI down to the web interfaces and persistent data storage.

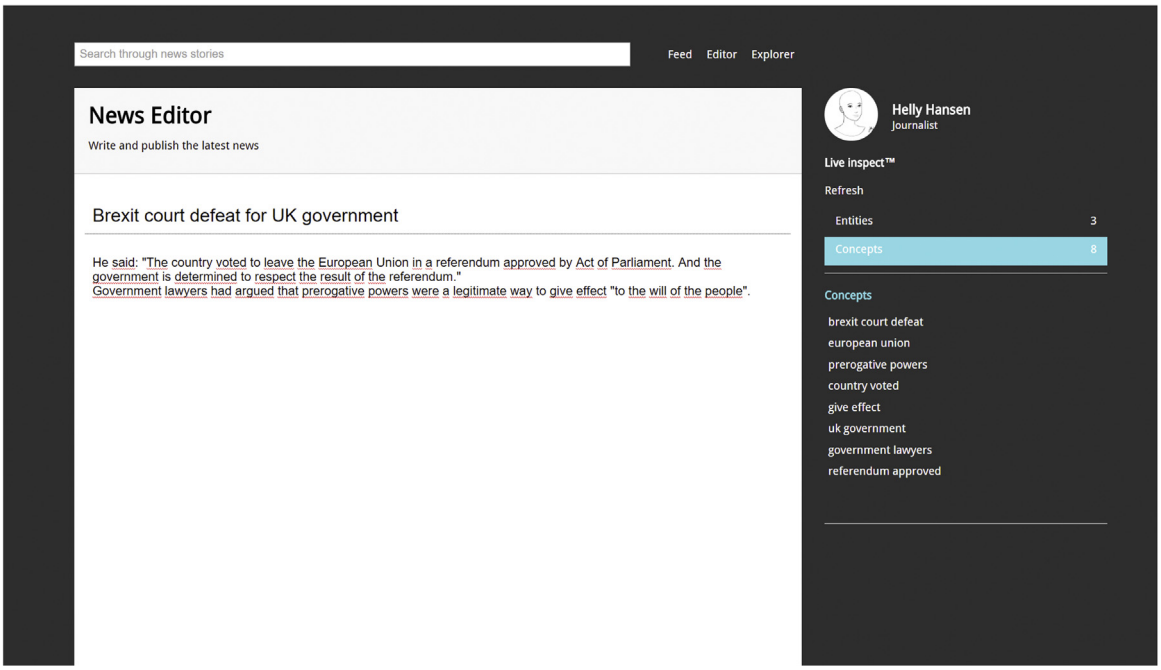

Fig. 4. The web front end of the News Hunter prototype. The left side shows an early version of the report editor. The right part shows the results of analysing the report-in-writing.

Isolating the front end as a separate component inside a well-defined API is beneficial in the same way as the report-editor component. Many news organisations already have newsroom tools in place with which the News Hunter front end should integrate as seamlessly as possible.

4.13. 13 Microservice framework

The microservice framework leverages REST endpoints in order to: decouple the architecture, make it more flexible, and facilitate APIs written in different languages. The microservices are implemented using Flask, a framework written in and for Python, but also usable from C#.

A message-exchange component is an essential part of a distributed architecture such as News Hunter, and Flask is a natural light-weight starting point for a prototype. Using a standardised message-exchange framework makes the component interfaces clearer and prepares for introducing a more comprehensive, big-data ready message-exchange framework such as Kafka in future versions of the prototype.

5. Evaluation

This section reviews the results of our final evaluation of the central News Hunter features with human participants. In addition to the evaluations presented here, we conducted functional tests and standard performance evaluations at the end of each development iteration.

5.1. Procedure

Six subjects participated in the final evaluation: two with background as journalists and four with background as software developers. They all had experience with newsroom systems through the development of Wolftech News system, but they had not participated in developing News Hunter. Their experience with the News system was beneficial because they thus understood the users' needs and were able to asses the prototype in its intended context as a component of News. Our evaluation focussed on identifying the relatively most and least useful News Hunter features (see the questions below) in order to identify important paths for further work. Importantly, we did *not* attempt to compare News Hunter to existing platforms, when the participants' connection to Wolftech might have biased their responses.

We first introduced the purpose of the evaluation briefly and demonstrated the prototype. The participants were then presented with three English-language news reports, each 350–400 words long, from different domains: the Reuters article "Trump gives nod to Republican tax-credit proposal on Obamacare" (389 words), the BBC article "UKIP burka ban policy' misguided' says party's MEP" (491 words), and the CNN article "Paul Pogba: Is he worth $120 million?" (449 words). The participants were instructed to read the news reports and explore the following six features of News Hunter in order: inspecting published news reports through the front end; displaying named entities for reports-in-writing; displaying topics for reports-in-writing; generating summaries of longer reports; retrieving clusters of recent reports ("top stories"); and retrieving reports related to the report-in-writing. For each feature, we asked three questions: what is useful about this feature?, what is missing from or is problematic with this feature?, and is the feature useful overall? In general, the respondents were positive to all the features, but they also pointed to many areas of improvement, which we now turn to discuss.

5.2. Inspecting published news reports

All participants considered this feature useful for getting an overview of existing news. They suggested several possible improvements: an option to skip the inspect menu to go straight to the report text; a flag indicating when a report has been updated; access to the report's update history; more prominent display of report texts compared to entities and keywords; and displaying the most relevant named entities as a word cloud.

5.3. Displaying named entities

We used the prototype to extract named entities and types for each report and select entities with background information available in DBpedia Most of the participants agreed that associating named entities with reports was useful, in particular for retrieving background information without having to leave the tool to search the web, and that the named entities provided helpful context for understanding a report. They found that the term "Named Entities" in the user interface was confusing and suggested that entities should instead be organised according to type, such as "Person", "Organisation", "Place", or even more specifically as "Football player". However, a few named entities present in the reports were not found by the tool, and the participants had issues with a few of the suggestions. The respondents found between 73 % and 89 % of the entities appropriate for each report, having smaller or larger issues with 11%–27% of them. For example, for the BBC article ("burka ban"), "Commonwealth" and "EU" had been typed as countries and the wrong election was linked. The respondents also noted that the named-entity feature could benefit from more filtering options; from triangulating information from multiple sources; and from using verifiable sources. Filtering could be done, for example, by entity relevance, by type (people, organisations, places.. .), and by map.

5.4. Displaying topics

We also used the prototype to extract topics for the three reports. TextRank tended to describe topics using single keywords or shorter 2–4 word phrases, whereas RAKE suggested many longer topic phrases of 4–9 words. Before showing these auto-generated topics to the participants, we asked them to suggest their own suitable topics for each report. We then let them assess the two topic lists from TextRank and RAKE. Most participants preferred the shorter topics generated by TextRank, which were more similar to their own suggestions. But the RAKE topics were sometimes preferred, and it was pointed out that the two topic-labelling styles were useful each in their own way.

Most participants agreed that the topics were useful for finding related content and for saving as metadata. The topic labels also offered a quick and easy way to find out what a report is about. Both journalists found the feature to be useful overall, but the domain experts were divided. They pointed out that the quality of the keywords still needed improvement, e.g., through tuning and word-sense disambiguation. It should also be possible to remove bad keyword suggestions from a report. In addition to keywords used in the text, higher-level topics could also be suggested.

Although our research aim is to validate the overall News Hunter architecture, and not to optimise individual components through extensive tuning, we compared the TextRank topics quantitatively with the set of user-suggested topics. F1 scores were low (0.4−0.5), with better recall (0.55−0.75) than precision (0.3−0.4) but, of course, these measures cannot be compared to evaluations that use established and validated gold standards. Nevertheless, the evaluation results suggest that keyword extraction with TextRank needs improvement or at least tuning to become useful for journalistic work. We did not compare the RAKE topics quantitatively in the same way, because the longer topic phrases could not be meaningfully compared with the participants' mostly single-keyword topics.

5.5. Report summaries

Two of the domain experts said the summaries were an effective way of gaining a quick overview of a report. One of the journalists added that the summaries could be published already while the main report was being written. However, the quality of the summaries generated by the prototype needed improvement, and several of the respondents were therefore negative to the presented version of this feature.

5.6. Top stories

The top-story feature shows clusters of recent reports that have been detected as reporting the same event. The respondents found this useful for identifying multiple reports about the same event and for identifying unfolding events that need to be covered. However, only named entities and keywords that apply to all the reports in a group should be displayed, and the numbers of named entities and keywords should be restricted. Respondents also wanted categories and a slider to limit the time frame. Ability to search through groups, control which sources to include, and explain group membership criteria were also called for. Two participants saw it as problematic to make the top-story feature too prominent, because it might drown less prominent but nevertheless important events.

5.7. Related reports

All the respondents appreciated the possibility to retrieve related previous news articles. One of them pointed out that the usefulness of this feature would increase with better word-sense disambiguation. Some respondents would like more metadata about the previous reports, such as their date and source. Another suggested feature was being able to see if other journalists were working on similar or related reports.

5.8. Report editor

The respondents found the automatic identification of named entities and keywords in the text useful, because they could be used both as metadata and for retrieving relevant background information. In-editor access to related news reports was also considered beneficial. The respondents missed the ability to remove unwanted named entities and keywords.

5.9. Classification

We also used the prototype to categorise the three reports. The participants agreed that the suggested categories were correct, but too general. More precise and descriptive ones would be needed to make them useful in practice. To evaluate single-label classification more precisely, we had held out 30 % of the pre-labelled RSS items from BBC's Insight dataset used for training to be used for evaluation (Müller and Guido, 2016). Both classifiers were tuned to reach an overall F1 score of 0.89. On the nine news categories used for evaluation, they both performed excellently for sports (F1 = 0.98.. . 0.99) but less well for education (F1 = 0.68.. . 0.69), most likely because the training set contained fewer education articles. To evaluate multi-label classification, we had also held out 30 % of the pre-labelled news articles from The Guardian used for training to be used for evaluation. String equivalence and WordNet synsets (Miller, 1995) were used to match The Guardian's labels to IPTC news codes before inspecting the matches manually. The F1 scores were 0.84 for the Scikit-learn SVM and 0.72 for the Keras MLP.

5.10. Event detection

To evaluate event detection, we analysed 1292 news reports harvested by our prototype from a variety of newspapers. We compared the resulting clusters with the top stories listed in Google News. Two out of the six identified clusters were deemed correct

after manual inspection, and the four remaining ones were also among the top events in Google News.

6. Discussion

6.1. Research question

The paper has presented an architecture for and a prototype of a journalistic knowledge platform. The platform is able to harvest data from both pre-news and post-news data sources, exemplified by Twitter, RSS, and online newspapers. It combines the full strength of externally linked knowledge graphs and ontologies with topical AI techniques for natural-language processing and machine learning. It has been designed to be able evolve and grow, using decoupled components and subcomponents that exchange information and services through a microservice framework and a central knowledge graph. The evaluation with domain experts suggest that they found the platform potentially useful. The paper thereby answers our research question positively.

6.2. Design goals

The News Hunter architecture and prototype satisfy central design goals we have established in collaboration with our industrial partner. It makes state-of-the-art techniques for semantic analysis of natural-language texts and for managing and enriching knowledge graphs available to journalists embedded in their newsroom environment. It is designed to support live harvesting of potentially news-relevant items from the net and for real-time analysis of these items and of news reports that journalists are working on. It offers both push and pull provision of information to journalists, although the latter is currently more developed. It accepts multi-language input through the use of translators, and it is language neutral through its use of language-agnostic IRIs to represent entities and concepts that can have multiple language-tagged labels to describe their names.

6.3. Novelty

The background section has presented the most similar existing systems described in the literature, showing that each one demonstrates information services that are potentially highly useful for journalists and newsrooms. However, only NewsReader (Vossen et al., 2016) uses the full strength of externally linked knowledge graphs defined by an OWL ontology, and only Reuters Tracer (Liu et al., 2017) harvests social media in order to detect events that are not yet reported in the news. The other systems focus on textual and other news reports that have already been published and do not fully leverage knowledge graphs, ontologies, and linked open data. Hence, News Hunter stands out by leveraging both pre-news (such and Twitter and Facebook) and post-news (such as RSS and web news) information sources in combination with a central knowledge graph. It is unique in combining the following features: It targets journalists and newsrooms specifically. It uses knowledge graphs centrally to integrate, organise, and analyse information for journalistic and newsroom use. It harvests, lifts, and ingests news items from multiple sources, both pre-news and post-news. And it enriches the knowledge graph with facts taken from the linked open data (LOD) cloud (Bizer et al., 2009b). To the best of our knowledge, News Hunter is the only platform that combines all these features.

Yet, the most important novelty of the present paper may be that it, for the first time, frames *architecture* for journalistic knowledge platforms as a research issue. The literature on existing systems either does not address architecture at all or it presents processing pipelines without discussing architecture in further depth. Hence, we plan to investigate architecture for journalistic knowledge platforms more in depth in our further work.

7. Conclusion and further work

The paper has shown how pre-and post-news data sources and processing and storage techniques can be combined into a flexible architecture for news journalism. It thereby answers our initial research question positively. The evaluation results are encouraging, and we plan to continue to evolve our architecture and prototype in several directions. We are currently redesigning and reimplementing the prototype into a parallelised big-data ready platform built on top of state-of-art technologies such as Apache Kafka, Blazegraph, Cassandra, Terraform and Ansible. We are also extending and improving the precision of semantic lifting by leveraging recent embedding- and deep-learning based analysis techniques, e.g., (Mikolov et al., 2013; Devlin et al., 2018; Le and Mikolov, 2013; Yang et al., 2020). We expect that our component-based architecture will make it easy to integrate such new lifting and analysis components into the new News Hunter. Furthermore, we are building a larger-scale journalistic knowledge graph and news-item collection for further testing and research. We are currently exploring these possibilities in the ongoing News Angler project,[2] which aims to provide journalists with unexpected angles on and surprising background information about newsworthy events as they unfold (Opdahl and Tessem, 2020). Longer-term research challenges include taking into account journalists' and other newsroom workers' preferences and work styles and moving from pure text to non-textual information types, acknowledging that audio and video are also important news sources.

Declaration of Competing Interest

The authors report no declarations of interest.

Acknowledgement

Early development of News Hunter was supported by NCE Media, a Norwegian Center of Expertise.

References

Allemang, D., Hendler, J., 2011. Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL. Elsevier.
Beck, K., 1999. Embracing change with extreme programming. Computer 32 (October 10), 70–77, http://dx.doi.org/10.1109/2.796139.
Berners-Lee, T., Hendler, J., Lassila, O., 2001. The semantic web. Sci. Am. 284 (5), 34–43.
Berven, Arne, Christensen, Ole A., Moldeklev, Sindre, Opdahl, Andreas L., Villanger, Kjetil, 2018. News hunter: building and mining knowledge graphs for newsroom systems. Svalbard, Norway In: Proceedings of NOKOBIT 2018, vol. 26, p. 11.
Bizer, C., et al., 2009a. DBpedia – a crystallization point for the Web of Data. Web Semant. Sci. Serv. Agents World Wide Web 7 (3), 154–165.
Bizer, C., Heath, T., Berners-Lee, T., 2009b. Linked data - the story so far. Int. J. Semantic Web Inf. Syst. 5 (3), 1–22.
Castells, P., et al., 2004. Neptuno: Semantic Web Technologies for a Digital Newspaper Archive., pp. 445–458.
Corcoglioniti, F., Rospocher, M., Aprosio, A.P., 2016. Frame-based ontology population with PIKES. IEEE Trans. Knowl. Data Eng. 28 (December 12), 3261–3275, http://dx.doi.org/10.1109/TKDE.2016.2602206.
Devlin, J., Chang, M.-W., Lee, K., Toutanova, K., 2018. BERT: pre-training of deep bidirectional transformers for language understanding. ArXiv181004805 Cs, Oct. 2018, (Accessed 27 August 2019) [Online]. Available: http://arxiv.org/abs/1810.04805.
Diakopoulos, N., 2017. Computational journalism and the emergence of news platforms. In: Franklin, B., Eldridge, S.A. (Eds.), The Routledge Companion to Digital Journalism Studies. , 1st ed. Routledge, London; New York, pp. 176–184, 2016.

[2] http://newsangler.uib.no

Ekdale, B., Singer, J.B., Tully, M., Harmsen, S., 2015. Making change: diffusion of technological, relational, and cultural innovation in the newsroom. J. Mass Commun. Q. 92 (4), 938–958.
Fernández, N., et al., 2006. NEWS: bringing semantic web technologies into News agencies. In: The Semantic Web - ISWC 2006., pp. 778–791.
Fernandez, N., Fuentes, D., Sanchez, L., Fisteus, J.A., 2010. The NEWS ontology: design and applications. Expert Syst. Appl. 37 (December 12), 8694–8704, http://dx.doi.org/10.1016/j.eswa.2010.06.055.
Gangemi, A., Presutti, V., Recupero, D.R., Nuzzolese, A.G., Draicchio, F., Mongiovi, M., 2017. Semantic web machine reading with FRED. Semant. Web 8 (6), 873–893, http://dx.doi.org/10.3233/SW-160240.
Garcia, R., Perdrix, F., Gil, R., Oliva, M., 2008. The semantic web as a newspaper media convergence facilitator. Int. J. Web Semant. Technol. 6 (April 2), 151–161, http://dx.doi.org/10.1016/j.websem.2008.01.002.
García, R., Perdrix, F., Gil, R., 2006. Ontological infrastructure for a semantic newspaper. In: Presented at the Semantic Web Annotations for Multimedia Workshop, SWAMM.
Germann, U., Liepins, R., Gosko, D., Barzdins, G., 2018. Integrating multiple NLP technologies into an open-source platform for multilingual media monitoring. In: Proceedings of Workshop for NLP Open Source Software (NLP-OSS), Melbourne, Australia, pp. 47–51, http://dx.doi.org/10.18653/v1/W18-2508.
Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.
Heravi, B.R., McGinnis, J., 2015. Introducing social semantic journalism. J. Media Innov. 2 (March 1), 131, http://dx.doi.org/10.5617/jmi.v2i1.868.
Hevner, A.R., 2007. A three cycle view of design science research. Scand. J. Inf. Syst. 19 (2), 4.
Hevner, A.R., March, S.T., Park, J., Ram, S., 2004. Design science in information systems research. MIS Q. 28 (1), 75–105.
Latar, N.L., 2015. the robot journalist in the age of social physics: the end of human journalism? In: Einav, G. (Ed.), The New World of Transitioned Media. Springer International Publishing, Cham, pp. 65–80.
Le, Q., Mikolov, T., 2013. Distributed representations of sentences and documents. In: Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a Meeting Held December 5-8, 2013, Lake Tahoe, Nevada, United States, p. 9.
Leban, G., Fortuna, B., Brank, J., Grobelnik, M., 2014. Event Registry: Learning About World Events from News., pp. 107–110.
Liu, X., Nourbakhsh, A., Li, Q., Shah, S., Martin, R., Duprey, J., 2017. Reuters Tracer: toward automated news production using large scale social media data. ArXiv171104068 Cs, Nov., (Accessed 14 January 2019) [Online]. Available: http://arxiv.org/abs/1711.04068.
Machill, M., Beiler, M., 2009. The Importance of the Internet for Journalistic Research: a multi-method study of the research performed by journalists working for daily newspapers, radio, television and online. Journal. Stud. 10 (April 2), 178–203, http://dx.doi.org/10.1080/14616700802337768.
Mikolov, T., Chen, K., Corrado, G., Dean, J., 2013. Efficient Estimation of Word Representations in Vector Space. ArXiv13013781, Cs, Jan., (Accessed 27 July 2018) [Online]. Available: http://arxiv.org/abs/1301.3781.
Miller, G.A., 1995. WordNet: a lexical database for English. Commun. ACM 38 (11), 39–41.
Miroshnichenko, A., 2018. AI to bypass creativity. Will robots replace journalists? (The answer is "yes"). Information 9 (July 7), 183, http://dx.doi.org/10.3390/info9070183.
Mitchell, T., et al., 2018. Never-ending learning. Commun. ACM 61 (April 5), 103–115 https://doi.org/10.1145/3191513.
Müller, A.C., Guido, S., 2016. Introduction to Machine Learning with Python: A Guide for Data Scientists. O'Reilly Media, Inc.
Opdahl, A.L., Tessem, B., 2020. Ontologies for finding journalistic angles. Softw. Syst. Model. (June), http://dx.doi.org/10.1007/s10270-020-00801-w.
Opdahl, Andreas L., Berven, Arne, Alipour, Kamal, Christensen, Ole A., Villanger, Kjetil, 2016. Knowledge graphs for newsroom systems (short paper). Bergen, Norway In: Proceedings of NOKOBIT 2016, vol. 24, p. 2.
Rospocher, M., et al., 2016. Building event-centric knowledge graphs from news. Int. J. Web Semant. Technol. 37–38 (March), 132–151, http://dx.doi.org/10.1016/j.websem.2015.12.004.
Rudnik, C., Ehrhart, T., Ferret, O., Teyssou, D., Troncy, R., Tannier, X., 2019. Searching news articles using an event knowledge graph leveraged by wikidata. In: Companion Proceedings of The 2019 World Wide Web Conference on - WWW' 19, San Francisco, USA, pp. 1232–1239, http://dx.doi.org/10.1145/3308560.3316761.
Russel, S., Norvig, P., 2009. Artificial Intelligence: A Modern Approach, third edition. Prentice Hall.
Singhal, A., 2012. Introducing the knowledge graph: things, not strings. In: Google Official Blog (Accessed 7 November 2019) http://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html.
Thurman, N., 2019. Computational journalism. In: Wahl-Jorgensen, K., Hanitzsch, T. (Eds.), The Handbook of Journalism Studies. , second edition. Routledge.
Vaishnavi, V., Kuechler, W., Petter, S., 2004. Design Research in Information Systems. http://desrist.org/design-research-in-information-systems.
Vossen, P., et al., 2016. NewsReader: using knowledge resources in a cross-lingual reading machine to generate more knowledge from massive streams of news. Knowledge Based Syst. 110 (October), 60–85, http://dx.doi.org/10.1016/j.knosys.2016.07.013.
Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., Le, Q.V., 2019. XLNet: generalized autoregressive pretraining for language understanding. ArXiv190608237 Cs, Jun. 2019, (Accessed 4 July 2019) [Online]. Available: http://arxiv.org/abs/1906.08237.