Identifying Notable News Stories

Antonia Saravanou[1(✉)], Giorgio Stefanoni[2], and Edgar Meij[2]

[1] National and Kapodistrian University of Athens, Athens, Greece
antoniasar@di.uoa.gr
[2] Bloomberg, London, UK
giorgio.stefanoni@gmail.com, emeij@bloomberg.net

Abstract. The volume of news content has increased significantly in recent years and systems to process and deliver this information in an automated fashion at scale are becoming increasingly prevalent. One critical component that is required in such systems is a method to automatically determine how notable a certain news story is, in order to prioritize these stories during delivery. One way to do so is to compare each story in a stream of news stories to a notable event. In other words, the problem of detecting notable news can be defined as a ranking task; given a trusted source of notable events and a stream of candidate news stories, we aim to answer the question: "Which of the candidate news stories is most similar to the notable one?". We employ different combinations of features and learning to rank (LTR) models and gather relevance labels using crowdsourcing. In our approach, we use structured representations of candidate news stories (triples) and we link them to corresponding entities. Our evaluation shows that the features in our proposed method outperform standard ranking methods, and that the trained model generalizes well to unseen news stories.

1 Introduction

With the rise in popularity of social media and the increase in citizen journalism, news is increasing in volume and coverage all around the world. As a result, news consumers run the risk of either being overwhelmed due to the sheer amount of news being produced, or missing out on news stories due to heavy filtering. To deal with the information overload, it is crucial to develop systems that can filter the noise in an intelligent fashion. Due to the highly condensed language used in news, automated systems have been developed to process them and generate well-defined structured representations from their content [9]. Each structure is a so-called triple that represents an event in the form of *who* did *what* to *whom*, with additional metadata information about *when* and *where* this happened. Such representations (triples) form a *knowledge graph* (KG). There are multiple computational benefits when searching, labeling, and processing KGs due to their clean and simple structure [11,14].

A. Saravanou—This work was done whilst interning at Bloomberg.

J. M. Jose et al. (Eds.): ECIR 2020, LNCS 12036, pp. 352–358, 2020.
https://doi.org/10.1007/978-3-030-45442-5_44

Table 1. Example of a query q_0 and two candidate events c_0 and c_1.

Query q_0	**Tagged Query**
A suicide bomber detonates a vehicle full of explosives at a military camp in Gao, Mali, killing at least 76 people and wounding scores more in Mali's deadliest terrorist attack in history. *Date*: 17 January 2017	A [WIKI: Suicide_attack] detonates a [WIKI: Vehicle] full of [WIKI: Explosive] at a military camp in [WIKI: Gao], [WIKI: Mali], [WIKI: Murder] at least 76 people and wounding scores more in Mali's deadliest [WIKI: Terrorism] in history. *Date*: 17 January 2017

	Subject	**Predicate**	**Object**	**Date**	**Location**
c_0	Armed Gang	Carry out suicide bombing	Armed rebel	17 Jan. 2017	Gao, Mali
c_1	Armed Gang	Carry out suicide bombing	Military	17 Jan. 2017	Bamako, Mali

A common approach to measure notability of a news event is to track it through a proxy metric. For example, Naseri *et al.* [7] decide whether an article describes a notable event by counting the user interactions, while Setty *et al.* [10] cluster together similar news articles and then use the cluster size to decide if the common theme is notable. Wang *et al.* [12] propose a recommendation framework that takes as input a stream of news and predicts the user's click-through rate.

In this paper, we approach the problem of identifying notable news stories as a ranking task, i.e., we rank structured news stories represented as triples against notable events. We use *Wikipedia's Current Events Portal* (WCEP) [2] as curated notable events and, using a combination of textual and semantic features, we build a learning to rank (LTR) model to solve the ranking problem.

2 Problem Statement

Let $\mathcal{Q} = [q_0, \ldots, q_k]$ denote a stream of events, where each *query event* $q_i \in \mathcal{Q}$ is a notable event composed of a textual description and of a publication date. Let $\mathcal{C} = [c_0, \ldots, c_l]$ denote a stream of *candidate* events. Each $c_j \in \mathcal{C}$ is a structured representation of a news story that consists of a *triple* of the form (s, p, o), where s is the subject, p is the predicate, and o is the object, together with information about the *location (city, country)* and the *date* of the news story.

Given a query $q_i \in \mathcal{Q}$ and a stream of candidates $\mathcal{C}$, we aim to rank each candidate $c_j \in \mathcal{C}$ by its relevance to the query q_i. A pair (q_i, c_j) is considered as *very relevant* when the information from q_i and c_j matches completely; it is considered as *relevant* when some of the information matches; otherwise, it is considered as *not relevant*. Table 1 shows a query q_0 and two candidates c_0 and c_1. The pair (q_0, c_0) is very relevant because c_0 matches q_0 completely; in contrast, the pair (q_0, c_1) is relevant because q_0 and c_1 disagree only on the location of the event.

3 Method

In this section we present our method to identify notable news stories which consists of three steps: (1) creating (query, candidate) pairs, (2) extracting textual and semantic features, and (3) training a learning to rank (LTR) model.

(1) Creating Pairs. We create the set of all possible (query, candidate) pairs where (i) the query and the candidate have the same publication date, and (ii) the query and the candidate have at least one word in common as a pair is unlikely to be relevant if they share no words.

(2) Extracting Features. We extract a set of features for each constructed pair. Our features can be classified into three groups as follows.
(i) ***Features related to a component.*** We compute the size of the query or the candidate (i.e., the number of terms in the query/candidate).
(ii) ***Features related to the pair.*** We calculate the Okapi *BM25* score, the term frequency (*TF*) and the term frequency–inverse document frequency (*TF–IDF*) for the query/candidate in the pair. We calculate these scores using the stemmed versions of the query/candidate (using the Porter Stemmer [8]). We further define a similarity score, *element match*, $EM(q_i, ele_{c_j}) = |q_i \cap ele_{c_j}|/|ele_{c_j}|$ where an element ele_{c_j} is one of the: subject, predicate, object, description of the predicate, location, and the date in the candidate c_j. For each of those, we calculate the fraction of the number of common terms between the element ele_{c_j} and the query q_i to the total number of terms in ele_{c_j}. In addition, we compute all *EM* scores using the stemmed versions of the pair components. We also extract similarity scores for combinations of elements, as for example $EM(q_i, subject \cap predicate \cap object)$ and $EM(q_i, city \cap country)$.
(iii) ***Semantic features.*** An entity is a well-defined, meaningful and unique way to characterize a word/phrase. We therefore apply entity linking using the TagMe API [5] to identify entities (an example is shown in Table 1). Given the tagged query and the tagged candidate, we calculate the number of common entities using the Jaccard similarity.

(3) Ranking Pairs. We then use our features to train a learning to rank model in order to obtain a ranking of pairs. More details on the training and the tuning can be found in Sect. 4.

4 Experimental Setup

For the candidate news stories, we use the Integrated Crisis Early Warning System (ICEWS) [1] dataset which contains events that are automatically extracted from news articles using TABARI [3,9]. This system uses grammatical parsing to identify events (*who* did *what* to *whom*, *when* and *where*) using human-generated rules. The events are triples consisting of coded actions between socio-political actors. The actors refer to individuals, groups, sectors and nation states. The actions are coded into 312 categories. Geographical and temporal metadata are also associated with each triple (examples are shown in Table 1).

In our experiments, we use the same two weeks of data from ICEWS and WCEP. We remove triples with the generic action type "Make statement" as they do not convey any meaningful information. We then create pairs as described in Sect. 3. We build a crowdsourcing task (see below) to get golden truth labels. From the resulting annotated dataset, we only keep queries with at least one

Table 2. Distribution of the relevance labels in the dataset.

	Train	Validate	Test	Total
Very Relevant	220 (4%)	73 (4%)	47 (3%)	340 (3%)
Relevant	106 (2%)	20 (1%)	9 (1%)	135 (1%)
Not Relevant	5219 (94%)	1959 (95%)	1475 (96%)	8653 (96%)

relevant ICEWS triple as there are, e.g., sports events in the WCEP dataset but not in the ICEWS dataset. In total, the resulting dataset contains 9.1K pairs; 74 queries and 123 candidates per query on average. To evaluate our method in a real-world setting we split the dataset by date and use the first ten days for training, the next two days for validation, and the last two for testing.

Golden Truth. We employ crowdsourcing on the Figure-eight platform and ask annotators to judge the relevance of each pair on a 3-point scale (very relevant, relevant, not relevant).[1] Each pair (q_i, c_j) is annotated by at least three annotators and we use majority voting to obtain the gold labels. Our task obtains a inter-annotator agreement of 96.57%. Table 2 shows the distribution of relevance labels among pairs. The resulting dataset is highly skewed; with 3% annotated as *very relevant*, 1% as *relevant*, and 96% as *not relevant*.

Models. We explore various LTR algorithms and include results from RankBoost (RB) [6], lambdaMART (LM) [13], and Random Forest (RF) [4]. We experiment using different sets of features: *all* features (ALL), *all except entity-related* features (ALL$^-$), *selected* features (SEL) and *baseline* features (B). SEL features include BM25 and TF–IDF scores calculated from the original/stemmed versions, *EM* scores for subject, predicate, object and location, and the number of entities in common and Jaccard similarity between the query and the candidate. For B features, we only consider BM25 and TF–IDF scores calculated from the original/stemmed versions of the query and the candidate. We evaluate using MAP, Precision@k, NDCG@k and MRR.

5 Results and Discussion

In this section we discuss our experimental results and answer the following research questions. How does our method compare against the baselines? Does the performance vary with different parameter settings? Does the number of *relevant* pairs affect performance? Do we benefit from tagging entities?

5.1 Overall Performance

We compare the three LTR models on the ALL and B feature sets and show the results in Table 3. Our method (using ALL features) achieves better results than using just the baseline B features. For each model and feature set, we only show the best tuned model on the validation set. Our method consistently

[1] https://www.figure-eight.com/.

Table 3. Main results of the LTR models on our dataset.

	MAP	P@5	P@10	NDCG@5	NDCG@10	MRR
$\mathbf{RB_{All}}$	0.53	0.42	0.3	0.6	**0.62**	**0.75**
$\mathbf{LM_{All}}$	0.44	0.38	0.31	0.51	0.56	0.65
$\mathbf{RF_{All}}$	**0.56**	**0.47**	0.32	**0.64**	0.61	**0.75**
$\mathbf{RB_{B}}$	0.37	0.33	0.29	0.37	0.45	0.6
$\mathbf{LM_{B}}$	0.34	0.31	0.29	0.36	0.44	0.6
$\mathbf{RF_{B}}$	0.44	0.4	**0.33**	0.42	0.57	0.62

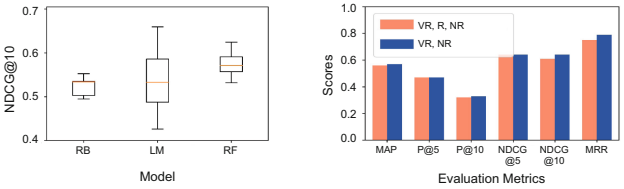

Fig. 1. (Left) Results for each model on the validation set. Each box shows the median and upper/lower quartiles. (Right) Performance using RB with selected features on two datasets.

outperforms all baselines, achieving 5–12% improvements on NDCG@10. These improvements are statistically significant with $p \leq 0.01$ using paired t-test.

We tune the parameters for each model on the validation set using NDCG@10. Figure 1 (left) shows the performance quartile plot using different parameter settings. RB and RF models show less sensitivity in the parameters tuning compared to LM. We evaluate the models when ranking pairs using all annotations (*VR*, *R*, and *NR*). We perform the same experiment using only the *VR* and *NR* labeled pairs. Figure 1 (right) shows that the model achieves better results when excluding the *R* labeled pairs. This is expected as the relevant label is very rare (only 1%, see Table 2) and the models tend to consider it as noise.

Our next step is to evaluate different combinations of features (All, All$^-$, Sel, B). We show our findings in Table 4. First, we compare our method using All$^-$ and B feature sets. We show that using the proposed features (Sect. 3) we achieve better performance for all LTR models. Second, we evaluate the performance of the models when we add the entity features by comparing All and All$^-$. In Table 4, we show that there is a statistically significant improvement ($p \leq 0.01$) on MRR (+7%) when we add the entity–related features.

5.2 Analysis

In this section, we show examples of the output from our best performing setting, i.e., RF with All features using the *VR* and *NR* labeled pairs. We show our best and worst per–query NDCG@10 performance. The best one achieves a score of 1, which indicates that our method was able to rank all pairs in the right order. The top–1 ranked pair is the query "*At least 15 children are killed and 45 more are injured after a school bus collides with a truck in Etah, India.*

Table 4. Results using binary relevance labels.

	MAP	P@5	P@10	NDCG@5	NDCG@10	MRR
$\mathbf{RB_{All}}$	**0.57**	0.42	0.3	0.61	**0.65**	0.69
$\mathbf{LM_{All}}$	0.53	0.4	0.3	0.56	0.61	0.71
$\mathbf{RF_{All}}$	**0.57**	**0.47**	**0.33**	**0.64**	0.64	**0.79**
$\mathbf{RB_{All-}}$	0.52	**0.47**	**0.3**	0.62	0.62	0.68
$\mathbf{LM_{All-}}$	0.44	0.33	0.28	0.47	0.54	0.65
$\mathbf{RF_{All-}}$	**0.53**	0.44	**0.3**	**0.64**	**0.65**	**0.72**
$\mathbf{RB_{Sel}}$	**0.61**	0.44	0.28	**0.67**	**0.67**	**0.81**
$\mathbf{LM_{Sel}}$	0.53	0.4	0.27	0.56	0.6	0.75
$\mathbf{RF_{Sel}}$	0.55	**0.47**	**0.31**	0.62	0.65	0.62
$\mathbf{RB_{B}}$	**0.44**	**0.38**	0.28	**0.47**	0.51	0.6
$\mathbf{LM_{B}}$	0.38	0.33	0.28	0.39	0.47	0.54
$\mathbf{RF_{B}}$	0.42	0.31	**0.3**	0.42	**0.58**	**0.63**

Date: 20 Jan. 2017" and the candidate <*Attacker (from India)*, *Kill by physical assault*, *Children (from India)*> with metadata <*Etah*, *India*, *20 Jan. 2017*>. The item with the worst per–query NDCG@10 performance is "*Mexican drug lord Joaquin Guzman is extradited to the USA, where he will face charges for his role as leader of the Sinaloa Cartel. Date: 20 Jan. 2017*" paired with the candidate <*USA*, *Host a visit*, *Narendra Modi*> with metadata <-, *USA*, *20 Jan. 2017*>. This query is about the extradition of a drug lord, while the candidate is about a visit of the Prime Minister of India. However, among the top–10 ranked candidates, the most relevant one is the triple <*USA*, *Arrest, detain, or charge with legal action*, *Men (from Mexico)*> with metadata <*Kansas City*, *USA*, *20 Jan. 2017*>, ranked 9th. This shows that even in the worst ranking per–query, our method ranks a relevant candidate in the top–10.

In summary, we provide quantitative and qualitative performance analyses of our proposed method and we conclude that learning to rank is a viable method to determine notability of news stories. Among the key steps of our method are: (i) the extraction of textual and semantic features, and (ii) the exclusion of the pairs that do not convey strong signal, i.e., the ones labeled as '*relevant*'. The RF model outperforms all baselines and it is also more robust with respect to hyperparameter settings. These findings show that our approach to detect notable news through ranking is a promising one. Although our method obtains high performance (MRR = 81%), we believe we can attain further improvements by leveraging relations of the identified entities to discover implicitly relevant ones, such as <*Narendra_Modi*, *isPrimeMinisterOf*, *India*>.

6 Conclusion and Future Work

In this paper, we present a method to rank notable news representations which leverages textual and semantic features. Our evaluation on labeled pairs from WCEP and the ICEWS shows that our method is effective. In the future, we intend to include features based on the relations of the tagged entities from external KGs, such as DBPedia and Freebase.

References

1. Integrated Crisis Early Warning System (ICEWS). https://dataverse.harvard.edu/dataverse/icews. Accessed 17 Jan 2020
2. Wikipedia's Current Events Portal (WCEP). https://en.wikipedia.org/wiki/Portal:Current_events. Accessed 17 Jan 2020
3. Boschee, E., Lautenschlager, J., O'Brien, S., Shellman, S., Starz, J.: ICEWS automated daily event data. Harvard Dataverse (2018)
4. Breiman, L.: Random forests. Mach. Learn. **45**(1), 5–32 (2001). https://doi.org/10.1023/A:1010933404324
5. Ferragina, P., Scaiella, U.: TAGME: on-the-fly annotation of short text fragments (by Wikipedia entities). In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, CIKM 2010. ACM (2010)
6. Freund, Y., Iyer, R., Schapire, R.E., Singer, Y.: An efficient boosting algorithm for combining preferences. J. Mach. Learn. Res. **4**, 933–969 (2003)
7. Naseri, M., Zamani, H.: Analyzing and predicting news popularity in an instant messaging service. In: ACM SIGIR Conference on Research and Development in Information Retrieval. ACM (2019)
8. Porter, M.F.: An algorithm for suffix stripping. In: Sparck Jones, K., Willett, P. (eds.) Readings in Information Retrieval. Morgan Kaufmann Publishers Inc., Burlington (1997)
9. Schrodt, P.: Automated coding of international event data using sparse parsing techniques. In: Annual Meeting of the International Studies Association (2001)
10. Setty, V., Anand, A., Mishra, A., Anand, A.: Modeling event importance for ranking daily news events. In: ACM International Conference on Web Search and Data Mining, WSDM 2017. ACM (2017)
11. Voskarides, N., Meij, E., Tsagkias, M., de Rijke, M., Weerkamp, W.: Learning to explain entity relationships in knowledge graphs. In: ACL-IJCNLP. Association for Computational Linguistics (2015)
12. Wang, H., Zhang, F., Xie, X., Guo, M.: DKN: deep knowledge-aware network for news recommendation. In: World Wide Web Conference, WWW 2018 (2018)
13. Wu, Q., Burges, C.J., Svore, K.M., Gao, J.: Adapting boosting for information retrieval measures. Inf. Retrieval **13**, 254–270 (2010). https://doi.org/10.1007/s10791-009-9112-1
14. Yang, S., Han, F., Wu, Y., Yan, X.: Fast top-k search in knowledge graphs. In: 2016 IEEE 32nd International Conference on Data Engineering (ICDE). IEEE (2016)