IntoNews: Online News Retrieval using Closed Captions

Roi Blanco[a], Gianmarco De Francisci Morales[a], Fabrizio Silvestri[a]

[a] *Yahoo Labs, Barcelona – Spain*

Abstract

We present IntoNews, a system to match online news articles with spoken news from a television newscasts represented by closed captions. We formalize the news matching problem as two independent tasks: closed captions segmentation and news retrieval. The system segments closed captions by using a windowing scheme: sliding or tumbling window. Next, it uses each segment to build a query by extracting representative terms. The query is used to retrieve previously indexed news articles from a search engine. To detect when a new article should be surfaced, the system compares the set of retrieved articles with the previously retrieved one. The intuition is that if the difference between these sets is large enough, it is likely that the topic of the newscast currently on air has changed and a new article should be displayed to the user. In order to evaluate IntoNews, we build a test collection using data coming from a second screen application and a major online news aggregator. The dataset is manually segmented and annotated by expert assessors, and used as our ground truth. Our evaluation is based on a set of novel time-relevance metrics that take into account three different aspects of the problem at hand: precision, timeliness and coverage. We compare our algorithms against the best method previously proposed in literature for this problem. Experiments show the trade-offs involved among precision, timeliness and coverage of the airing news. Our best method is four times more accurate than the baseline.

Email addresses: roi@yahoo-inc.com (Roi Blanco), gdfm@yahoo-inc.com (Gianmarco De Francisci Morales), silvestr@yahoo-inc.com (Fabrizio Silvestri)

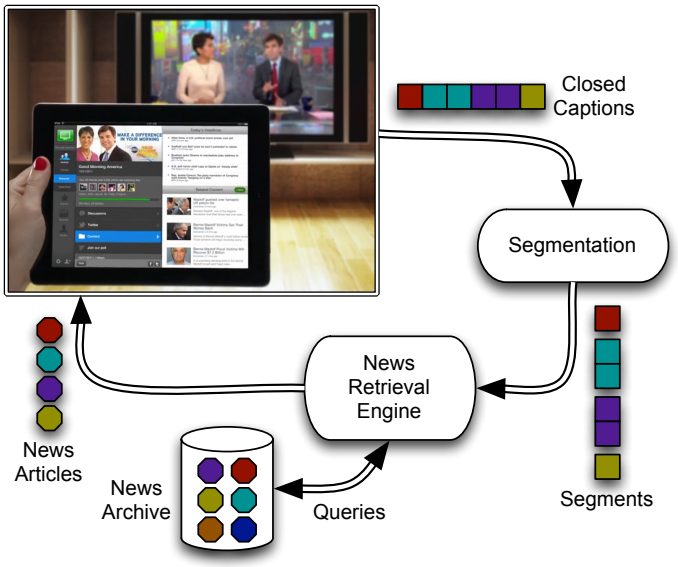

Figure 1: Conceptual schema of IntoNews.

1. Introduction

Television has been the most important communication medium of the last century. However, in the last few years the Web has started to take over this role thanks to a wider offer of content and the possibility of interaction. Recently, a new breed of applications for mobiles and tablets has started appearing on the market, offering the so-called *"second screen"* experience. The goal of these applications is to enhance the TV-watching experience by providing additional content related to the program airing at the moment, thus bridging the TV and Web worlds. By allowing the audience to interact with the program on TV, second screen applications ultimately aim at increasing user engagement. These applications are the natural evolution of a widely recognized trend: between 75% and 85% of TV viewers use another device at the same time.[1]

[1] `http://www.guardian.co.uk/technology/appsblog/2012/oct/29/social-tv-second-screen-research`

second screen[2] from Yahoo! is an example of a second screen application, and the focus of the current work. When second screen launched in 2011, users immediately acclaimed this application as a fun way of watching TV programs. The user experience for people watching TV program is greatly improved. For instance, while watching a football game on TV it can show statistics about the teams playing, or show the title of the song performed by a contestant in a talent show. Other services include forums, episode synopsis, real-time meme generator (CapIt), polls and much more. second screen aims at enhancing the experience of watching TV transforming it into a *"large scale"* social activity. The additional content provided by second screen is a mix of editorially curated and automatically selected one.

From a research perspective, one of the most interesting and challenging use cases for these applications is related to news programs (newscasts). When a user is watching a newscast, they might want to delve deeper into the news airing at the moment. This work presents IntoNews, a system that finds an online news article that matches the piece of news discussed in the newscast currently airing on TV, and displays it to the user in real-time.

The main problem underlying IntoNews is matching different data sources that speak about the same piece of news. On one side we have the text from online news articles. On the other, we obtain the content of the newscast currently airing from the streams of *Closed Captions* (cc) broadcasted along with it by television networks.

The challenges in making IntoNews effective are multiple. The news article we surface to the user must match exactly the news currently airing. The problem is even more challenging given that the matching article has to be selected among the thousands published online every day. The language used on TV and in news articles has different characteristics. Furthermore, the cc tend to be noisy, lack proper capitalization and contain many typos and misspellings. Finally, and most importantly, news articles must be surfaced as soon as possible

[2] `http://www.intonow.com`

to be valuable to the user.

We propose a solution based on techniques from the realm of information retrieval (IR). Figure 1 shows the conceptual schema of the components of our system and how they interact with each others. We decompose the main news matching task into two sub-tasks: find a good segmentation of the stream of CC, and retrieve relevant news for the segment as soon as possible. We model a newscast as a series of contiguous segments, each matching a single cohesive topic. The *segmentation* problem consists in finding the boundaries of these news segments in the stream of CC. The *retrieval* problem consists in formulating a query given a segment, and issuing the query to an underlying IR engine.

While the user is watching the newscast, the system continuously processes the CC and tries to detect a segment boundary ("new story"). If a new segment is detected, it examines the incoming text until it has enough information to retrieve the news article from the IR engine. When enough information has been accumulated, the system submits the query to the IR engine, retrieves the results, and shows them to the user.

There are differences between this problem and those faced by traditional IR systems. Users of a typical IR system issue queries to retrieve a set of top-k most relevant items from a collection. We can identify three distinct phases in a typical IR process: (i) the user formulating the query and issuing it, (ii) the IR system processing the query and retrieving the top-k items, and (iii) the user checking a subset of the resulting items to satisfy their information need.

IntoNews differs in phases (i) and (iii). First, it does not require the user to formulate a query, rather the system "*implicitly*" formulates one for the user by using the content of the newscast airing on TV. In fact, formulating a query by observing only a continuous stream of text without any indication on topic boundaries, query keywords, important concepts or entities is a challenging task, which is fundamentally different from typical IR tasks.

Second, the user sees a small number of results that are continuously changing as new CC arrive. Usually, IR quality assessment only evaluates the amount of relevant documents ranked at the top of the result list. However, IntoNews

has to account for *when* a news item is displayed, as the system should surface a matching item before the newscast has changed topic. Therefore, we evaluate the quality of the system in terms of both relevance and timeliness.

Given that in our setting timeliness directly impacts the relevance of results, we need to design a proper evaluation testbed. We build a ground truth dataset consisting of a day's worth of CC from a news channel. We manually segment the dataset, associate each segment with a set of news items, and annotate the segments with relevance judgements. Furthermore, we propose a family of time-dependent metrics that evaluate the effectiveness of retrieval with respect to timeliness. These metrics discount the value of a relevant result with time, i.e., the sooner the *more relevant*. We experiment with four instances of the metric that use different decay factors.

To the best of our knowledge the only attempt made so far to address the problem presented in this work is discussed by Henzinger et al. [17] and it is summarized in Section 2. The best solution of Henzinger et al. is our baseline which we compare to in Section 6.

Preliminary results of this work were previously presented by the same authors [8]. In this paper we provide a proper formalization of the problem, a novel technique to detect a change of topic in the stream of CC, extensive experimentation for the proposed methods, and a publicly-available dataset used as our testbed.

The research contributions presented can be summarized as follows:

- We investigate the task of matching online news articles to news airing on a newscast;
- We formalize the problem, and present a framework that models the task as two separate sub-tasks:
  - (*i*) Find a topically-homogeneous segmentation of the stream of CC;
  - (*ii*) Retrieve relevant news as soon as possible;
- We design an evaluation testbed for the problem that takes into account the timeliness of the solution;

- We provide the dataset used in our testbed to the research community to foster research on this topic[3];
- We discuss several options to solve the segmentation and retrieval problems and we conduct a thorough experimentation for assessing the performance of our solutions.

The remainder of the paper is structured as follows. Section 2 reviews related literature. We formalize our problem and present a framework to address it in Section 3. Section 4 discusses our evaluation framework and we describe the ground truth dataset in Section 5. Section 6 presents our experimental results, while Section 7 presents our conclusions and future directions for research.

2. Related Work

As mentioned in the introductory section, Henzinger et al. [17] study the same news matching problem. They show how to extract a query from a stream of CC text and submit it to a news search engine. The methods they present are variations of a simple $tf \cdot idf$ scheme and all the methods work by considering non-overlapping segments of the CC stream. When a query has been generated, the current text portion is discarded and a new query is generated from the subsequent terms in the CC stream (in some variations called history-based, they keep terms from previously generated queries). The results are then post-processed to eliminate news articles that have been already shown to the user. The system is evaluated by measuring how relevant a matching news is for a given portion of CC text. While the idea presented by Henzinger et al. is similar to the one presented in this paper, the realization is different. First of all, our goal is the *timely* generation of matching news items. For this reason, our evaluation discounts the relevance of an item by the time needed by the algorithm to find a match. Also, differently from Henzinger et al., we value only perfect matches. Finally, in addition to $tf \cdot idf$-based methods, we generate

[3]Available upon request at `http://webscope.sandbox.yahoo.com`

queries on the basis of entities contained in the news and we decide when to start a new query on the basis of the feedback results from the news retrieval engine.

Castillo et al. [11] describe a system that uses closed captions to find matching news and songs. The system relies on a machine-learned supervised classification model to find matchings, and does not take into account timeliness. However, the system is not available, and it is only described from a functional point of view. Therefore it is not possible to compare against it. Another work that is related to second screen applications is that of Odijk et al. [26] where they describe and approach a task of associating with semantic entities windows of CC text.

A related field of research is *information filtering* (IF) [25], where textual information flows continuously through the system, which has to select the most important documents for a given user's profile. Historically, work on selective dissemination of information started by a 1958 article of Luhn [21]. Even if the term *information filtering* was not used in that paper, the goal of the system described was exactly that of selecting the best document, or documents, according to a user's profile. The term Information Filtering, in fact, became popular after a paper by Denning [14] who argued that emails should be filtered and sorted according to their importance. In the IR community, information filtering approaches have mostly dealt with finding an appropriate model for describing users' profiles [24], and on improving filtering effectiveness [18]. In particular, efficiency of those systems were studied in order to make the IF approaches usable in practice [6, 33]. Belkin and Croft [5] study the relationship between IR and IF in more detail. Some later studies focus more on extracting information from microblogs such as twitter [3].

We remark that this *stream-based news retrieval* task is different from traditional information filtering since we are not given a stream of documents, rather we are given a stream of text from which queries have to be extracted and submitted.

Some of the ideas related to topic segmentation have been drawn from the

Topic Detection and Tracking (TDT) realm. TDT systems discover the topical structure in unsegmented streams of news as they appear across multiple media. One particularly interesting sub-task of TDT is *story segmentation* which detects changes between topically cohesive sections of multimedia or spoken data [16]. Allan [1] provides a good overview on existing approaches to TDT. As in the case of IF, also TDT's scientific literature in the last years has been influenced by the presence of social media and microblogging. Takahashi et al. [32] propose to detect the emergence of new topics by analyzing the anomalies measured through the model. Aggregating anomaly scores from hundreds of users, authors show that we can detect emerging topics only based on the reply/mention relationships in social-network posts. Merlino et al. [23] analyze broadcast news from CNN to find cues of segment boundaries. While their approach relies on multimedia streams, the text cues can be applied to our setting. For example, the presence of the token Hello is a strong indicator of a start-of-story boundary. These features could be embedded in a machine learning approach to augment our sliding-window-based segmentation.

Another related field of research is that of New Event Detection (NED) in streams of text [28]. The goal of NED is to analyze a stream of text documents, possibly coming from different sources, and to "group" documents related to the same event together. The most prevailing approach of NED was proposed by Allan et al. [2], Yang et al. [35], Brants et al. [9], in which documents are processed by an on-line system. Among these approaches, $tf \cdot idf$ is a popular heuristic to build ranking functions over streaming data. Farahat et al. [15] employ a model that updates incrementally term weights using $tf \cdot idf$ on different data sources, and similarly Kumaran and Allan [19] employ an incremental variant of $tf \cdot idf$ to detect news events. The most recent developments of NED research have been mainly driven by the increased popularity of News websites. Wei et al. [34] develop an effective event episode discovery mechanism to organize news documents pertaining to an event of interest. In particular, they propose two novel time-based metrics that they successively use for feature selection and document representation in a temporal-based event episode

discovery technique.

It is worth pointing out that NED is only loosely related to our problem of segmenting the stream of CC. In fact, our problem is to find boundaries of the text that describes an event in a stream of text containing multiple events. The difference is even more evident if we relate this task with the more general task of retrieving associated news articles. In this case, we want to detect an event as soon as possible and to build and issue a query to the underlying IR engine.

3. Problem formulation

In this section we present our notation, provide a statement of the problem, and describe a framework that decomposes the problem in smaller, more manageable tasks.

The primary input is represented by an unbounded stream of CC lines $\mathcal{C} = \langle c_1, c_2, \ldots \rangle$. Each CC line $c = (t, l)$ is composed by a timestamp $t \in \mathcal{T}$ and a short piece of text l. The timestamp t increases monotonically in the stream, and represents the time at which the CC text is available to our system, i.e., $c_i \prec c_j \iff t_i < t_j$.

We assume that at any given time there exist a finite number of topics $\mathcal{N}$, which represent noteworthy news events. We further assume the existence of a function $\mathcal{L}_{\text{CC}} : \mathcal{C} \to \mathcal{N}$ that maps each line of CC in the stream $\mathcal{C}$ to a topic $n \in \mathcal{N}$. IntoNews does not have access to the function $\mathcal{L}_{\text{CC}}$.

The secondary input is a collection of documents $\mathcal{D}$, which may have any format; the only requirement is that they can be indexed and searched via an underlying IR engine. Similarly to CC lines, we assume that each document $d \in \mathcal{D}$ can be mapped to a topic $n \in \mathcal{N}$ by a function $\mathcal{L}_{\mathcal{D}} : \mathcal{D} \to \mathcal{N}$. The system has no access to this function either.

Let us now formally state the problem:

Problem 1. *We are given as input an unbounded stream of closed caption lines $\mathcal{C}$ and a collection of documents $\mathcal{D}$. We assume the existence of a set of topics $\mathcal{N}$, and two functions $\mathcal{L}_{\text{CC}}$ and $\mathcal{L}_{\mathcal{D}}$ that map, respectively, closed caption lines*

and documents to topics. The problem is to find, $\forall c \in \mathcal{C}$*,* k *documents* $\mathcal{R}^k \subset \mathcal{D}$ *such that* $\mathcal{L}_{\text{CC}}(c) = \mathcal{L}_{\mathcal{D}}(d)$, $\forall d \in \mathcal{R}^k$.

Note that Problem 1 does not seek neither to identify the topics nor to approximate the topic functions $\mathcal{L}$. Rather, it only asks to find matching documents for each line of CC, or, equivalently, for each timestamp t.

We avoid defining an optimization objective in the problem formulation, instead we simply state the characterization of an ideal solution. The discussion of issues related to evaluation is deferred to Section 4.

The challenges in Problem 1 originate from two sources: (i) we do not have access neither to the topics $\mathcal{N}$ nor to the functions $\mathcal{L}$, and (ii) we see the input stream line by line (i.e., the solution requires an online algorithm). In practice, a solution needs to deal with unspecified topics that might include loose boundaries, and make online decisions based on local information.

3.1. Proposed Solution

As already mentioned, we take an IR approach in designing IntoNews. In order to employ traditional IR techniques, both for solving the problem and for evaluation, the solution is restricted to finding ranked lists of documents rather then sets; with abuse of notation, we denote with $\mathcal{R}^k$ a ranked list of k documents. Therefore, the system can be regarded as a function $f^{\mathcal{D}}_{\text{IntoNews}} : \mathcal{C} \rightarrow \{\mathcal{D}^k\}$ that matches to each CC line $c_i \in \mathcal{C}$ a document list $\mathcal{R}^k_i \subset \mathcal{D}$, while optimizing a relevance function as defined in Section 4.

In practice, as shown in Section 5, topics arrive in segments in the stream, where contiguous lines of the segment belong to the same topic. Rather than trying to match a list of news items to *each* CC line, we focus on finding the *boundaries* between two different topics in the stream. Therefore the goal becomes to detect the boundaries of the topics as soon as possible, and minimize the duration of the topic mismatch between $\mathcal{C}$ and $\mathcal{R}^k$.

We identify three different sub-problems of Problem 1. First, the system has to identify segments of consecutive lines in $\mathcal{C}$ that belong to the same topic. This can be thought of as identifying a pair of points in time (t_1, t_2) that bound

the segment, with a function $f_{\text{seg}} : \mathcal{C} \rightarrow \{\mathcal{T} \times \mathcal{T}\}$. These bounds implicitly define a sequence of CC lines: $S = \{(t_i, l_i) \mid \ t_1 \leq t_i < t_2\}$. Secondly, we need to construct a query from the sequence of lines with a function $f_{\text{q}} : S \rightarrow S$. The query should adequately represent the topic in order to be matched against the document collection. Here we use a representation based on words, but more generally the query could comprise different units, possibly capturing higher order semantics (e.g., named entities). In its simplest form, f_{q} can be the identity function, but as we discuss next there are benefits in a more compact representations of the query. Lastly, the system needs to rank documents in the collection for the query with a function $f_{\text{rank}} : S \rightarrow \{\mathcal{D}\}$. The former function ($f_{\text{seg}}$) represents the closed caption segmentation component of Figure 1, while the latter two functions (f_{q} and f_{rank}) together represent the news retrieval engine in our schema.

Summarizing, in order to solve Problem 1 we have to tackle two tasks. The first one consists in selecting a segment of CC text such that a retrieval oracle would be able to retrieve the corresponding matching documents for the topic, and we call this problem the *segmentation problem*. The oracle is just a conceptual tool, therefore we also have to design an effective retrieval method. Given an optimal solution to the segmentation problem (i.e., a segment that corresponds to a single topic), a effective retrieval method should return documents associated with the same topic. We call this problem the *news retrieval problem*.

To build the final system we can optimize the two problems independently. However, as we detail in Section 3.4, we study the two problems in a more organic way, by leveraging the feedback of the news retrieval engine to decide on segment boundaries.

3.2. The Segmentation Problem

The system is presented with a continuous stream of CC lines that are added to a buffer $\mathcal{B}$ for building the query. There are several strategies for managing $\mathcal{B}$.

The simplest strategy is to use a windowing approach to build candidate seg-

ments. We explore two different fixed-size variants of the windowing approach:[4] (*i*) a *sliding window* approach (SW_Γ), and (*ii*) a *tumbling window* approach (TW_Γ). The parameter Γ is the size of the window in seconds. SW_Γ trims the oldest CC line from $\mathcal{B}$ when its size exceeds Γ, and therefore builds a new candidate segment of maximum duration Γ for each new CC line in the stream. TW_Γ builds adjacent windows of fixed size Γ, that is, it proposes a new candidate segment and empties $\mathcal{B}$ whenever adding a line to $\mathcal{B}$ would make it exceed Γ. Therefore, it proposes a candidate segment every Γ seconds at most.

Formally, the f_{seg} functions implemented by the two windowing approaches are the following.

$$\text{SW}_\Gamma(\mathcal{C}) = \{(t_i, t_j) \mid t_j - t_i = \Gamma\}$$
$$\text{TW}_\Gamma(\mathcal{C}) = \{(t_i, t_j) \mid t_i = k \cdot \Gamma, t_j = t_i + \Gamma, k \in \mathbb{N}\}$$

The main motivation to choose these approaches is that they are computationally inexpensive and simple to implement. Furthermore, as we shall see in Section 6, they perform well in practice when combined with other methods for generating discriminative queries and retrieving results.

3.3. The News Retrieval Problem

In the remainder of the paper, and for the sake of experimentation, Therefore, whenever referring to the *underlying IR engine* we consider BM25F as the retrieval model in use by f_{rank} [29].

Once the buffer $\mathcal{B}$ has been built by f_{seg}, we need to retrieve the news associated with it. A naïve implementation of f_{q} is the identity function. However, this kind of query is too noisy and lengthy for our dataset. Furthermore, the processing time needed for very long queries may be prohibitive for a real-time retrieval application.

The solution proposed here is to transform $\mathcal{B}$ into a more effective and efficiently-processable query. Therefore, we aim at reducing the buffer $\mathcal{B}$ to

[4] We defer adaptive windowing to a following study.

a more compact version $\tilde{\mathcal{B}}$ while maintaining the same amount of expressiveness. We select the k terms with highest $tf \cdot idf$ to build the query, where k is a parameter of f_{q}. The term frequency is computed in the buffer $\mathcal{B}$, while the inverse document frequency is computed from the document collection $\mathcal{D}$.

3.4. Topic Change Detection

An important issue is *when* to show a new result to the user, i.e., when to submit a new query to the underlying IR engine. Given that candidate queries are generated continuously by the system, the simplest option is to issue each and every query, and we refer to this variant as plain TF-IDF. We evaluate the overhead of this method in Section 6.

A smarter design involves trying to detect when the topic in the buffer has changed. We refer to this technique as *Topic Change Detection* (TCD). Formally, a TCD scheme is a function f_{q} that returns a new buffer $\tilde{\mathcal{B}}$ only if it detects a topic change, otherwise it returns the same buffer $\mathcal{B}$ returned at the previous invocation. The new buffer is initially empty.

There are a number of strategies to implement a TCD scheme, ranging from Natural Language Processing to Machine Learning [4, 12]. Here we explore an IR approach.

Our TCD schemes leverage the underlying IR engine for feedback. They query the underlying IR engine with the current candidate query, and they decide whether there has been a change in the topic by analyzing the results of the query. If the result set has changed "considerably" from the results of the last returned query, the scheme detects a topic change and returns the new query.

We propose three different variants of TCD that use different ways of measuring change in the result set: *Result Jaccard Overlap* (RJO), *Entity Jaccard Overlap* (EJO) and *Entity Jensen-Shannon Divergence* (EJS). All the strategies measure the distance between the candidate query and the last returned query, and are parameterized by a threshold $\theta \in [0, 1]$ that determines their sensitivity.

RJO measures the topic distance by the Jaccard overlap between the result

sets of the queries. In this case, each news article in the result list is considered as an item in a set. It detects a topic change when the overlap between the sets falls below the threshold.

EJO measures the topic distance by the Jaccard overlap between the sets of entities extracted from results of the queries. This methods builds a set from the entities extracted by each news article in the result list. As the previous method, it detects a topic change when the overlap falls below the threshold

EJS measures the topic distance by the JS divergence between the distributions of entities extracted from results of the queries. This method computes the distribution of entities extracted from the news articles in the result list. In this case, it detects a topic change when the divergence is over the threshold.

4. Quality Metrics

The problem defined in Section 3 bears some resemblance to information filtering, recommender systems and traditional information retrieval. In fact, the result of our system is a ranked list of news items that has to be returned with particular efficiency constraints. In fact, time plays an important role in our application scenario and it must be included in the metrics used to assess the quality of our proposed methods.

Given these requirements, we quantify the solution to our problem with a higher-order utility function $\phi(S, f) \rightarrow \mathbb{R}$ that measures the *relevance* of a set of ranked lists of documents for a given segment S. Our problem can then be seen as optimizing the utility function ϕ:

$$f_{\text{IntoNews}} = \arg\max_{f \in \mathcal{F}} \phi\left(S, f(c)\right) \ , \forall c = (t, l) \in S \ ,$$

where $\mathcal{F}$ is the space of possible solutions. In the remainder of this section we present our considerations for ϕ for a single segment S, and assume the evaluation is performed on average across all segments. For simplicity of notation, we assume the boundaries of the segment S to be $[0, \Gamma]$.

To correctly evaluate the system, we need to take into account two conflicting goals: (i) to provide news items that are relevant for the topic of the

current segment, and (ii) to provide these matchings as soon as possible. Providing results sooner means having less data available to create a query for the current segment, which in turn can introduce noise and degrade performance. Conversely, providing relevant results only when the current CC segment is over is of little value to the user since by then the topic has already changed. The function ϕ has to capture this trade-off.

Time-based relevance. We let the value of a match for a single segment depend on two factors: its relevance and the duration for which it is displayed on the screen of the user. For this reason, we define the relevance value of a news match for a segment to be the integral of its point-wise relevance:

$$\phi(S, f) = \int_0^\Gamma \nu\left(f(t, l)\right)\, dt$$

where $\nu(\cdot)$ measures the *value* of a single ranked list of documents $\mathcal{R}^k$ for the segment S, independent of time.

However we want to capture the notion that a match given at an earlier time is more valuable than the same match given at a later time. Therefore, we use a convolution with a *time discount function* $\psi(t)$.

$$\phi(S, f) = \int_0^\Gamma \nu\left(f(t, l)\right) \psi(t)\, dt$$

The time discount function $\psi(t)$ is a positive, monotonically non-increasing function with values between zero and one, that is, it has the following characteristics:

$$\psi(0) = 1; \qquad \psi(t) \geq 0, \quad \forall t; \qquad \frac{d}{dt}\psi(t) \leq 0, \quad \forall t$$

Given that we have different segment durations, we actually want a family of functions parameterized by Γ, and we add an additional constraint:

$$\psi_\Gamma(\Gamma) \leq \varepsilon,\ \varepsilon \ll 1 \tag{1}$$

The results provided by the system change at discrete times, so we can transform the integral into a discrete sum:

$$\phi(S, f) = \sum_{i=0}^{N} \nu_i\left(\mathcal{R}_i^k\right) \psi_\Gamma(t_i)$$

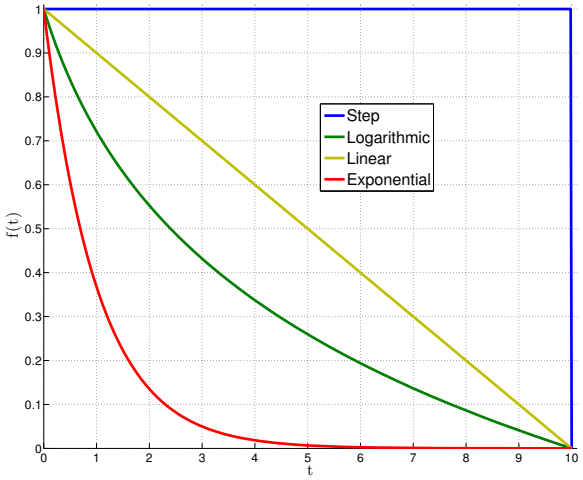

Figure 2: Time discount functions.

where $\mathcal{R}_i^k = f(t_i, l_i)$ is i-th results list $\mathcal{R}_i^k$ provided by our system, and t_i is the time at which it is provided.

We experiment with different options for the functions $\psi_\Gamma(t)$ and $\nu(\cdot)$.

We use four different time discount functions $\psi_\Gamma(t)$. All functions are defined for $0 \leq t \leq \Gamma$ and are zero elsewhere:

$$\begin{aligned} \textit{Step} : \psi_\Gamma(t) &= \text{sgn}(\Gamma - t) \\ \textit{Linear} : \psi_\Gamma(t) &= 1 - \frac{t}{\Gamma} \\ \textit{Logarithmic} : \psi_\Gamma(t) &= 1 - \log_\Gamma(1 + t \times \frac{\Gamma - 1}{\Gamma}) \\ \textit{Exponential} : \psi_\Gamma(t) &= e^{-t\,\frac{10}{\Gamma}} \end{aligned}$$

Figure 2 shows a visual representation of the four functions for $\Gamma = 10$. Given that the area below each curve is different, values computed with different time discount functions are not directly comparable.

We use Mean Average Precision (MAP) as the main measure for the value function $\nu(\cdot)$. We ignore unjudged results, which, as shown by Sakai [30], is a

better alternative than b-pref [10] when dealing with a result list with a high fraction of documents without relevance labels. However, we also explore the use of a measure based on Normalized Discounted Cumulative Gain (NDCG) to make use of unjudged results.

NDCG. As proposed by De Francisci Morales et al. [13], we use the NDCG measure in order to circumvent the limited size of the human judgements available from the ground truth. Rather than having binary relevance judgements, we consider 5 levels of relevance, from 0 to 4.

Our goal is to compute a relevance value for any news article (even unjudged ones) for a given segment. To do so, we use the entities in the news article as a proxy for its content and we bootstrap the procedure by using our ground truth as follows. For each segment S, let RE_S be the set of all the entities of the news articles judged relevant by human assessors (Section 5 explains how entities are extracted). We name RE_S the *relevant entity set* for the segment. Then, we define the relevance of a news article n with entities E_n for a segment S as:

$$Rel(n, S) = \left\lceil \frac{\mid E_n \cap RE_S \mid}{\mid E_n \mid} \times 5 \right\rceil$$

That is, we bin the value of $Rel(n, S)$ in 5 levels (0.2, 0.4, 0.6, 0.8, 1.0) according to the fraction of entities in the news items that are also in the relevant entity set, and assign value 0 to 4 to each level. Finally we compute NDCG for the result list with the relevance values computed as described above.

Coverage. Sometimes the system is not able to provide suggestions in time, mostly because the segment is too short or because the system misses the change in topic. For this reason we also assess the system in terms of how many segments have at least one suggestion. We define *coverage* to be the fraction of segments in the ground truth for which we provide at least one matching.

Suggestion Ratio. We also assess the *suggestion ratio*, the number of different results provided for each segment in the ground truth. Ideally, the suggestion ratio should be one, i.e., the system shows to the user only one result per segment. However, this result is hardly attainable in an online setting. Therefore, we evaluate the suggestion ratio to understand the "overhead" of the system.

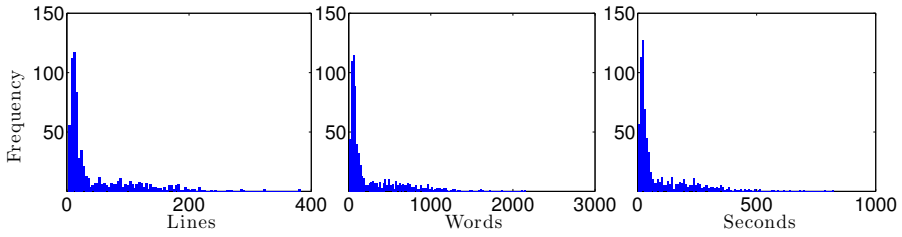

Figure 3: Segment length distribution

5. Dataset

This section describes the dataset we used for testing the IntoNews system. First, we extract from the internal archives of a second screen application a whole day worth of CC data from a very popular news TV network. We manually clean the closed captions by removing advertisements[5] and programs unrelated to news (e.g., talks shows).

We collect the news articles from a major online news aggregator spanning the two weeks preceding the date of the captions. These articles constitute the pool from which we draw the matching news. We represent the news articles by their full-text content and by named entities extracted with SuperSense tagger[6]. Following the approach of Matthews et al. [22], we index these entities in addition to the full text, and process title and body separately.

We then proceed to build a ground truth for the CC segments. Our ap-

[5] Advertisement detection is left as future work.
[6] `http://sourceforge.net/projects/supersensetag`

Table 1: Ground truth dataset characteristics.

Number of lines	$\approx$ 36k
Number of segments	720
Avg. No. of words per segment	$\approx$ 280
Avg. segment duration	$\approx$ 97s
Avg. No. of relevant news per segment	$\approx$ 5.3
No. of news articles	$\approx$ 180k

plication depends on how the stream of text is segmented, thus this step is fundamental. We manually segment the stream of CC lines into coherent pieces of texts that refer to the same *topic*. It is worth noting that the definition of topic is ambiguous and thus prone to errors. How general (or particular) should a topic be? How comprehensive should it be? What if one topic gradually and naturally leads to another? Is it a single topic or should it be split into two different topics?

We opt to be as general as possible, and define a topic as *an event concerning a single subject*. To give an example, consider the following fragment of text:[7]

```
>>> a new celebrity caught up in the chris brown
drake bar fight. tony parker says he suffered a
scratched retina in the fight and now has to put
off training with the french olympic basketball
team. also new, the new york city club where the
fight started has been shut down. police say eight
people were injured including singer chris bro
brown. witnesses told officers the fight started
when drake's entourage confronted brown as he was
leaving that club.
```

The text refers to a bar fight between singers Chris Brown and Drake, in which professional basketball player Tony Parker suffered a scratched retina; because of the fight, the club was also shut down. In this case, we judge the text as belonging to a single topic with a single subject, the fight. A finer segmentation could divide the fragment above into two different parts. The first one on Parker's injuries and the second one on the causes and effects of the fight at the disco. These seemingly ambiguous segments are common in our dataset and it is this ambiguity that makes the task harder than it appears.

We manually segmented the CC data and resolved any disagreement internally after discussions, or by majority voting if an agreement could not be reached. The sequence of segments so created constitutes our ground truth for the segmentation problem.

The next step is to find matching news in the pool of articles for each seg-

[7] This is a real piece of text extracted from our dataset.

ment. Given the size of the document collection (see Table 1), a fully-manual approach is not a viable option. Therefore, we automatically create queries for each segment and submit them to the IR engine to retrieve a set of 10 candidate news articles to be judged. We create the queries by following the process described in Section 3.3.

Each segment-news pair is individually assessed by expert human raters. In total about twenty expert human assessors evaluate the pairs of CC segments and news articles, and assign binary relevance judgements to them. We ask assessors to give a positive judgement only if the news matches exactly, and to give negative judgement when in doubt.

We compute the inter-rater agreement by drawing a random sample of judgments and repeating the assessment independently. Cohen's Kappa coefficient on the sample for the binary judgments between the authors on one side and the assessors on the other side is approximately 0.63. This value indicates *substantial agreement*, and is higher than the agreement found in other domains such as entity retrieval (0.57) [7], sentence retrieval (0.55) [31], or opinion retrieval [27] (0.34). This result shows that the task is less subjective than, for instance, assessing relevant opinions.

As an example of a segment-news pair, consider the following text:

```
>>> a giant leap for china. a chinese spacecraft
successfully docked with a orbiting space
laboratory this morning. this makes china to
complete a manned space docking behind the
united states and russia. the mission also sent
the country's first female astronaut into space.
```

The news article in Figure 4 is considered a match by the assessors. Indeed, the content of the news item describes exactly the same event as the one described by the captions.

Table 1 reports the characteristics of our dataset. Figure 3 shows the distribution of segment lengths measured in number of lines, words and seconds. It is evident that the distribution is skewed and heavy tailed. Many segments are short, but a significant fraction is longer than average.

Figure 4: A matching news article with respect to the fragment in the example.

6. Experiments

In order to evaluate our system, we experiment with the variants described in Section 3, and we explore the effects of the window size parameter Γ and of the TCD threshold parameter θ. We select the ranking model's parameters in a separate validation set (this is, setting weights for entities, title and body) and fix the parameter values of BM25F for the rest of the study, given that the ranking function is not the main focus of the paper. Similarly, for all the experiments in this paper we use $k = 10$ as the parameter for TF-IDF (number of terms per query). The underlying IR engine we use is Apache Solr,[8] modified to adopt the BM25F model, and we retrieve the top-100 results.

We first observe that given the constraint in Eq. 1 for the time function $\psi_\Gamma(t)$, a system that uses a segmentation oracle (ORACLE) always gets a value close to zero on any evaluation function. Indeed, the ORACLE selects the words that compose the query for the IR engine from all the words in the segment, and thus cannot possibly submit the query before all the words in the segment

[8] `http://lucene.apache.org/solr`

Table 2: Linear MAP score of sliding vs. tumbling window.

Variant	TF-IDF	$EJO_{0.2}$	$EJS_{0.8}$	$RJO_{0.2}$
SW_{10}	**0.195**	**0.172**	**0.195**	**0.183**
TW_{10}	0.185	0.145	0.185	**0.183**
SW_{30}	**0.251**	**0.307**	**0.252**	**0.240**
TW_{30}	0.208	0.217	0.208	0.205
SW_{60}	**0.253**	**0.175**	**0.261**	**0.256**
TW_{60}	0.195	0.066	0.195	0.187

have appeared, i.e., $t = \Gamma$ and, consequently, the segment is already over. Nevertheless, we use the result obtained by the ORACLE as an upper bound on the performance of our retrieval scheme. In the rest of the section, we normalize results with respect to the MAP score obtained by the oracle, i.e., 0.658.

Baseline. We compare against the best algorithm proposed by Henzinger et al. [17], i.e., the one attaining the highest score of news "exactly on topic" (R+) on both datasets. The authors call this algorithm $A1\text{-}BASE_{15}$. In our terminology this corresponds to a TW_{15}-TF-IDF strategy with queries of two terms, followed by a filtering step to remove near duplicates. We refer to this algorithm as BASELINE.

Segmentation strategies. We compare the two segmentation methods, the tumbling window approach (TW) and the sliding window approach (SW). For sake of clarity, Table 2 shows results only for the Linear time function given that all the other measures follow the same pattern. The experiments prove that the sliding window approach gets better results across various window sizes Γ and across most TCD variants. The reason behind this result is that SW keeps all the text and is thus more likely to detect the correct segment boundary. Table 2 also shows that larger values of Γ lead to larger differences in terms of objective function between the two approaches. Therefore, in the rest of the paper we focus on the results obtained with SW.

Topic Change Detection. We evaluate the effectiveness of the more sophisticated *Topic Change Detection* (TCD) schemes compared to the naïve TF-IDF one

Table 3: Linear MAP score of TCD variants.

Variant	First	Linear
BASELINE	0.114	0.072
SW_{10}-TF-IDF	0.108	0.195
SW_{60}-TF-IDF	0.064	0.253
SW_{30}-$EJO_{0.2}$	**0.593**	**0.307**
SW_{10}-$EJS_{0.2}$	0.116	0.194
SW_{60}-$EJS_{0.2}$	0.101	0.262
SW_{30}-$RJO_{0.2}$	0.296	0.240
SW_{60}-$RJO_{0.8}$	0.099	0.260

and the BASELINE. Table 3 shows the best variant for each TCD scheme we were able to obtain for different values of θ and Γ and compares it to plain TF-IDF. We show two results for each variant, the one with the highest *First* MAP to assess the detection of topic change, and the highest *Linear* MAP as a global quality indicator. *First* MAP measures the Mean Average Precision by evaluating only the first new match $\mathcal{R}^k$ for each segment, that is, it evaluates the behavior of the algorithm near the boundaries. Although this evaluation measure lies outside our evaluation framework, it is nonetheless useful to understand the behavior of the different variants.

EJO is clearly the best performing variant as it gets results as high as 60% of the ORACLE. Surprisingly, EJO has also the highest MAP value for both categories. However, as we shall see, such a high precision comes at the expense of a lower coverage. Finally, the EJS variant performs similarly to plain TF-IDF. Given that the EJO uses entities as well, the cause is the different way by which they measure topic distance. Our hypothesis is that by using a multi-set measure (JS divergence) rather than a set measure (Jaccard overlap) the results get influenced by repetitions of similar articles in the result list which decreases the detection performance.

The RJO variant seems to perform well compared to TF-IDF. Using larger windows increases the overall performance but decreases the responsiveness in detecting the topic change. In fact, all the methods with higher Linear MAP

Table 4: Coverage analysis.

Variant	Coverage	Suggestion ratio
BASELINE	0.904	5.389
SW_{30}-$EJO_{0.2}$	0.135	0.150
SW_{30}-$EJO_{0.4}$	0.489	0.863
SW_{30}-$EJO_{0.6}$	0.829	3.440
SW_{30}-$EJO_{0.8}$	0.933	6.818
SW_{30}-$RJO_{0.2}$	0.933	6.568
SW_{30}-$RJO_{0.4}$	0.981	13.133
SW_{30}-$RJO_{0.6}$	0.994	19.861
SW_{30}-$RJO_{0.8}$	0.999	24.149
SW_{30}-TF-IDF	1	39.153

score apart from EJO have a large window size $\Gamma = 60$. As with many other real-time applications, there is a tradeoff between being responsive and filtering out noise. We point out that BASELINE is the worst performing strategy in terms of Linear MAP. The technique of Henzinger et al. [17] does not consider timeliness among the goals. Queries are issued only when the window is full and thus it may be too late with respect to the beginning of the topic segment.

Coverage analysis. Table 4 shows the coverage and suggestion ratio for several methods. Ideally, one would like to have both measures as close to one as possible, as this would mean that we have identified the segment exactly. However, there is a trade-off involved: by being too conservative, as in the case of the EJO strategy, we get high precision and low overhead but low coverage as well. On the other hand, TF-IDF obtains perfect coverage but at the expense of a very high suggestion ratio of nearly forty.

For the TCD methods it is possible to tune the similarity threshold in order to get the desired coverage trade-off, and the EJO method seems more sensitive to its parameter, compared to RJO. In either case, it is possible to achieve a coverage around 93% with a number of suggestion ratio as low as seven. This result is more than acceptable for our envisioned application scenario. Note that BASELINE performs acceptably at the price, though, of a lower relevance of

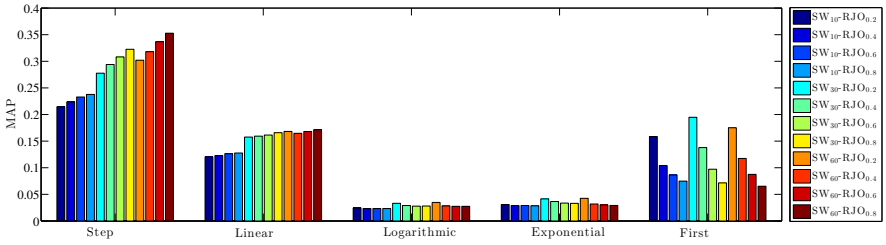

Figure 5: MAP values for various window sizes Γ and TCD thresholds θ.

retrieved results.

Time Functions. The relative performance of the different variants depends on the time discount function in use. Table 5 shows MAP scores for all time discount functions for EJO and RJO with different values of the threshold θ. By increasing the threshold, the algorithm becomes more aggressive (i.e., only very similar text is considered belonging to the same topic). While the Step and Linear scores increase with the parameter, the Logarithmic and Exponential scores decrease with it.

This behavior might seem counterintuitive at first glance, although consistent throughout our experiments. The explanation is that a more aggressive topic detection suffers from high noise levels at segment boundaries, when transitioning from a topic to the next one. At these points, it pays off to refrain from submitting a query until most of the window overlaps with the new segment to get accurate results. Segment boundaries are also the most profitable regions for the Logarithmic and Exponential functions, thus these functions favor getting a correct result right at the onset of a segment. On the other hand, a more aggressive topic detection is able to recover faster from incorrect guesses made at the beginning. Therefore the higher values for the Step and Linear functions for larger θ.

To test our hypothesis, we also show the *First* MAP score, which is computed by taking into account only the first suggestion per segment in the ground truth. Results are shown in the rightmost column in Table 5, where the score is raw,

Table 5: Performance with different time functions.

Variant	Step	Linear	Log.	Exp.	First
BASELINE	0.134	0.072	0.012	0.015	0.114
SW_{30}-$EJO_{0.2}$	0.419	**0.307**	**0.083**	**0.124**	**0.593**
SW_{30}-$EJO_{0.4}$	0.479	0.295	0.058	0.074	0.565
SW_{30}-$EJO_{0.6}$	0.464	0.260	0.047	0.056	0.373
SW_{30}-$EJO_{0.8}$	**0.482**	0.259	0.045	0.052	0.274
SW_{30}-$RJO_{0.2}$	0.422	0.240	**0.050**	**0.062**	**0.296**
SW_{30}-$RJO_{0.4}$	0.447	0.242	0.044	0.056	0.210
SW_{30}-$RJO_{0.6}$	0.468	0.245	0.043	0.052	0.147
SW_{30}-$RJO_{0.8}$	**0.491**	**0.252**	0.043	0.050	0.109

i.e., not weighted by any time function. It is evident that less aggressive topic detection performs as much as three times better.

Additionally, Table 5 shows how a more aggressive topic detection tends, in the limit, to the same behavior as TF-IDF. It is important to remark that MAP values weighted with different time discount functions are not directly comparable to each other, because of the difference in the area below each curve (i.e. their integral). EJO is an exception for Linear, because this result is influenced by variation in coverage of the method due to its sensitivity to θ.

As we already pointed out in the previous section, BASELINE does not take into account timeliness among the constraints to meet. For this reason, the quality of BASELINE is consistently worse than the methods we propose in this work.

Γ and θ. Figure 6 shows how the window size Γ and the TCD threshold θ interact with each other. We plot the Linear and Exponential MAP scores for EJO with two values for the threshold ($\theta = 0.2$ and $\theta = 0.8$) while varying Γ from zero to 90 seconds. When using a more aggressive threshold (0.8) the MAP score is far less sensitive to variations of Γ.

This behavior stems from the fact that with a less aggressive threshold less queries are fired, thus the content of the buffer $\mathcal{B}$ can change considerably between two consecutive queries. Therefore, the size of the buffer has a greater influence on the content of the query, and thus on the results. On the other

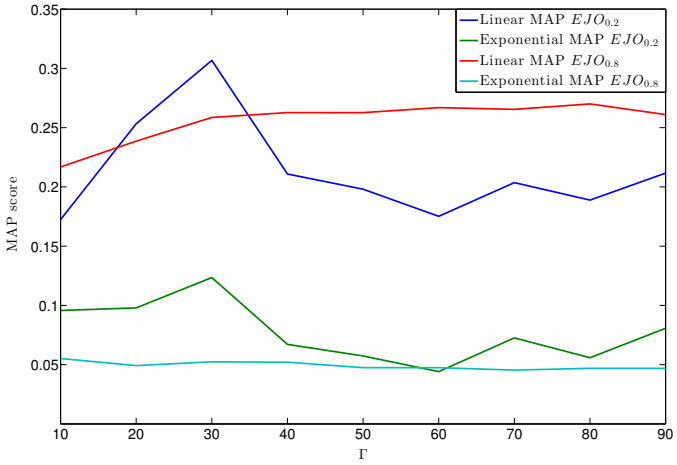

Figure 6: Interaction between window size Γ and TCD threshold θ.

hand, with a more aggressive threshold the method issues more queries, and thus the content of $\mathcal{B}$ often has a large overlap between two consecutive queries. Consequently, the size of the buffer becomes less relevant. From the figure it is also apparent that $\Gamma = 30$ is an optimal value for EJO when using a threshold $\theta = 0.2$.

Figure 5 shows the combined effects of enlarging the window size and using a more aggressive TCD threshold. While Linear and Step value increase, Logarithmic and Exponential value decrease. The cause of this behavior is clearly shown in the rightmost group, where we plot the *First* MAP score.

By using a more aggressive (larger) threshold it is easier to correct the query even after the topic has started, thus leading to a better ranking and MAP score. However, only methods that do not penalize late matchings excessively, such as Step and Linear, can take advantage of it. On the contrary, an aggressive threshold makes the method very sensitive to small changes and thus increases the noise around topic boundaries, which are the critical part for Logarithmic and Exponential given their sharper decrease, and for First given that they are the only part considered.

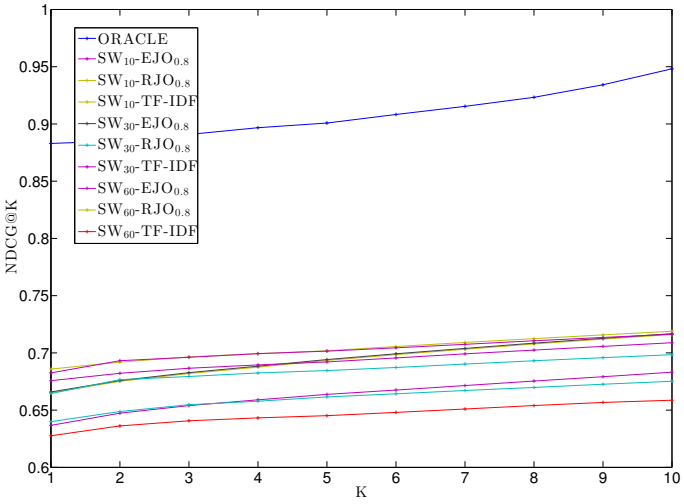

Figure 7: NDCG@K

Note that given that all the methods reach a coverage higher than 0.933 the comparison among them is fair.

NDCG. Figure 7 shows the NDCG values for a subset of the variants we experimented with. In this case, for the sake of clarity, we report only the results obtained by using the Step time function, although using other functions does not change the qualitative results. The best variants achieve a value of 0.70 while the ORACLE gets up to 0.95. For the ORACLE this result only means that the results are quite consistent and we have few outliers. On the other hand, the best variants achieve a value as high as 70% of the ideal one when suggesting related news.

Most of the variants are quite close to each other, so there is no clear winner here. This suggests that the system is able to find a large fraction of related news for most of the segments, and this behavior is consistent across different variants and parameters.

7. Conclusions and Future Work

In this paper we tackled the task of matching articles from a repository of news to a stream of closed captions coming from a newscast. We defined and formalized the problem and proposed a range of solutions, mainly borrowing techniques from information retrieval. We performed extensive experiments by using editorially assembled ground truth out of real-world data. The best performing strategy for segmenting closed captioning data is to use a fixed-width *sliding window*. On the other hand the best strategy to decide whether to issue a query to retrieve news articles, was based on EJO (Entity Jaccard Overlap) between result sets. Evaluation of the final retrieval quality has to trade-off two competing metrics: coverage and precision. When the main goal is coverage, then the best strategy results to be a sliding window of 30 seconds over the stream of CC text with a RJO topic change detection strategy when 40% overlap between new and old result set is detected, i.e., SW_{30}-$\text{RJO}_{0.4}$. In fact, this strategy has a high coverage (more than 98%) with a relatively low suggestion ratio (i.e., about 13.1 suggestions per chunk) and a high MAP score. If the goal is high precision, the best strategy is SW_{30}-$\text{EJO}_{0.2}$, which obtains a coverage of 0.135 with a very low suggestion ratio of 0.15 (i.e., approx. 87 times smaller) and a high quality in terms of linear MAP of 0.307. In all the experiments, we are able to improve consistently over the state-of-the-art baseline described in Henzinger et al. [17]. This behavior is consistent thorough all the time-dependent relevance metrics, which account for the timeliness of retrieved news items using different discount functions.

Future work. The proposed approach has a few shortcomings. First, the method used to detect boundaries is based on a simple sliding window approach. This approach might underperform when the number of discriminative words in the CC is low, and when the language between adjacent topics is too similar. Recent works on topic detection [20] in online newscasts might be helpful, although they are computationally heavy. We will exploit more sophisticated techniques based on machine learning for ranking functions and TCD.

In addition, considering the notion of popularity as a ranking factor for online news page to associate with CC text is another promising direction to explore. For example, considering scores similar to PageRank for online news will allow the system take into account not only textual features but also link-based features. Moreover, these popularity scores might be included from sources other than hyperlink structure, e.g., from online usage and social media. Topic change detection is a research line that deserves a greater deal of attention, specially in the case of continuous streams of closed captions. We plan to investigate further this aspect in order to optimize the amount of information displayed to users. Close captioning data analysis using more sophisticated techniques could lead to a new breed of second screen applications, such as displaying more fine-grained information about topics or people (entities) mentioned on newscasts or other TV programs. Assessing through a user study to what extent the time-discount functions are representative of the utility of the retrieved news items, and under what conditions one function would be preferred over another.

References

[1] James Allan, editor. *Topic detection and tracking: event-based information organization*. Kluwer Academic, 2002.

[2] James Allan, Ron Papka, and Victor Lavrenko. On-line new event detection and tracking. In *SIGIR*, pages 37–45, 1998.

[3] Andreas Bauer and Christian Wolff. An event processing approach to text stream analysis: Basic principles of event based information filtering. In *Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems*, DEBS '14, pages 35–46, 2014.

[4] Doug Beeferman, Adam Berger, and John Lafferty. Statistical models for text segmentation. *Mach. Learn.*, 34(1-3):177–210, 1999.

[5] Nicholas J. Belkin and W. Bruce Croft. Information filtering and informa-

tion retrieval: two sides of the same coin? *Comm. ACM*, 35(12):29–38, 1992.

[6] Timothy A. H. Bell and Alistair Moffat. The design of a high performance information filtering system. In *SIGIR*, pages 12–20, 1996.

[7] Roi Blanco, Harry Halpin, Daniel M. Herzig, Peter Mika, Jeffrey Pound, Henry S. Thompson, and Thanh Tran Duc. Repeatable and reliable search system evaluation using crowdsourcing. In *SIGIR*, pages 923–932, 2011.

[8] Roi Blanco, Gianmarco De Francisci Morales, and Fabrizio Silvestri. Towards leveraging closed captions for news retrieval. In *Proceedings of the 22Nd International Conference on World Wide Web Companion*, WWW '13 Companion, pages 135–136, 2013.

[9] Thorsten Brants, Francine Chen, and Ayman Farahat. A system for new event detection. In *SIGIR*, pages 330–337, 2003.

[10] Chris Buckley and Ellen M. Voorhees. Retrieval evaluation with incomplete information. In *SIGIR*, pages 25–32, 2004.

[11] Carlos Castillo, Gianmarco De Francisci Morales, and Ajay Shekhawat. Online Matching of Web Content to Closed Captions in IntoNow. In *SIGIR*, 2013.

[12] Freddy Y. Y. Choi. Advances in domain independent linear text segmentation. In *NAACL*, pages 26–33, 2000.

[13] Gianmarco De Francisci Morales, Aristides Gionis, and Claudio Lucchese. From Chatter to Headlines: Harnessing the Real-Time Web for Personalized News Recommendation. In *WSDM*, pages 153–162, 2012.

[14] Peter J. Denning. Electronic junk. *Comm. ACM*, 25(3):163–165, 1982.

[15] Ayman Farahat, Francine Chen, and Thorsten Brants. Optimizing story link detection is not equivalent to optimizing new event detection. In *ACL*, pages 232–239, 2003.

[16] Alexander G. Hauptmann. Story segmentation and detection of commercials. In *ADL*, page 24, 1998.

[17] Monika Henzinger, Bay-Wei Chang, Brian Milch, and Sergey Brin. Query-free news search. In *WWW*, pages 1–10, 2003.

[18] David A. Hull, Jan O. Pedersen, and Hinrich Schütze. Method combination for document filtering. In *SIGIR*, pages 279–287, 1996.

[19] Giridhar Kumaran and James Allan. Text classification and named entities for new event detection. In *SIGIR*, pages 297–304, 2004.

[20] Xiaoming Lu, Lei Xie, Cheung-Chi Leung, Bin Ma, and Haizhou Li. Broadcast news story segmentation using manifold learning on latent topic distributions. In *ACL (2)*, pages 190–195, 2013.

[21] H. P. Luhn. A business intelligence system. *IBM J. of Research and Development*, 2(4):314–319, 1958.

[22] Michael Matthews, Pancho Tolchinsky, Peter Mika, Roi Blanco, and Hugo Zaragoza. Searching through time in the new york times. *HCIR*, 2010.

[23] Andrew Merlino, Daryl Morey, and Mark Maybury. Broadcast news navigation using story segmentation. In *Proceedings of the Fifth ACM International Conference on Multimedia*, MM '97, pages 381–391, 1997.

[24] Masahiro Morita and Yoichi Shinoda. Information filtering based on user behavior analysis and best match text retrieval. In *SIGIR*, pages 272–281, 1994.

[25] Douglas W. Oard. The state of the art in text filtering. *User Modeling and User-Adapted Interaction*, 7(3):141–178, 1997.

[26] Daan Odijk, Edgar Meij, and Maarten de Rijke. Feeding the second screen: Semantic linking based on subtitles. In *Proceedings of the 10th Conference on Open Research Areas in Information Retrieval*, OAIR '13, pages 9–16, 2013.

[27] Deanna Osman, John Yearwood, and Peter Vamplew. Automated opinion detection: Implications of the level of agreement between human raters. *Inf. Process. Manage.*, 46(3):331–342, 2010.

[28] Ron Papka. *On-line New Event Detection, Clustering, and Tracking*. PhD thesis, University of Massachusetts, 1999.

[29] M. Taylor S. Robertson, H. Zaragoza H. Simple BM25 extension to multiple weighted fields. In *CIKM*, pages 42–49, 2004.

[30] Tetsuya Sakai. Alternatives to bpref. In *SIGIR*, pages 71–78, 2007.

[31] Ian Soboroff and Donna Harman. Novelty detection: the trec experience. In *HLT*, pages 105–112, 2005.

[32] T. Takahashi, R. Tomioka, and K. Yamanishi. Discovering emerging topics in social streams via link-anomaly detection. *Knowledge and Data Engineering, IEEE Transactions on*, 26(1):120–130, Jan 2014.

[33] Christos Tryfonopoulos, Manolis Koubarakis, and Yannis Drougas. Information filtering and query indexing for an information retrieval model. *ACM Trans. Inf. Syst.*, 27(2):10:1–10:47, 2009.

[34] Chih-Ping Wei, Yen-Hsien Lee, Yu-Sheng Chiang, Chun-Ta Chen, and Christopher C. Yang. Exploiting temporal characteristics of features for effectively discovering event episodes from news corpora. *JASIST*, 65(3): 621–634, 2014.

[35] Yiming Yang, Tom Pierce, and Jaime Carbonell. A study of retrospective and on-line event detection. In *SIGIR*, pages 28–36, 1998.