TREC 2013 Temporal Summarization

Metrics

October 24, 2013 Aslam, Diaz, Ekstrand-Abueg, Pavlu, Sakai

1 Temporal Summarization

Each event will be retrospectively analyzed for important sub-events or 'nuggets', each with a precise timestamp and text describing the sub-event. Our evaluation metrics will measure the degree to which a system can generate these nuggets in a timely manner.

1.1 Notation

A system/run *update* is a timestamped short text string comparable in length to a sentence. Colloquially, an update can be thought of as an SMS message or tweet. We generally denote an update as the pair (string, time): $u = (u.string, u.t)$. For example $u =$ ("`The hurricane was upgraded to category 4`", 1330169580) represents an update describing the hurricane category, now 4, pushed out by system $\mathcal{S}$ at UNIX time 1330169580 (i.e. 1330169580 seconds after 0:00 UTC on January 1, 1970). In this year's evaluation, the update string is chosen from the set of segmented sentences in the corpus as defined in the guidelines.

Two updates are semantically comparable using a text similarity measure or a manual annotation process applied to their *string* components; if two updates u and u' refer to the same information (semantically matching), then we write this as $u \approx u'$, irrespective of their timestamps. Because two systems might deliver the same update *string* at different times, it is generally not the case that $u.string = u'.string$ implies $u.t = u'.t$.

Given an event, our manual annotation process generates a set of gold standard updates called *nuggets*, extracted from wikipedia event pages and timestamped according to the revision history of the page. Editorial guidelines recommend that nuggets be a very short sentence, including only a single sub-event, fact, location, date, etc, associated with topic relevance. We refer to the canonical set of updates as $\mathcal{N}$. This manual annotation process is retrospective and subject to error in the precision of the timestamp. As a result we might encounter situations where the timestamp of the nugget is later than the earliest matching update.

In response to an event's news coverage, a system/run broadcasts a set of timestamped updates generated in the manner described in the Guidelines. We refer to a system's set of updates as $\mathcal{S}$. The set of updates received before time τ is,

$$\mathcal{S}_\tau = \{u \in \mathcal{S} : u.t < \tau\} \tag{1}$$

2 Evaluation

Our goal in this evaluation is to measure the precision, recall, timeliness, and novelty of updates provided by a system.

2.1 Preliminaries

Our evaluation metrics are based on the following auxiliary functions.

- **Nugget Relevance** Each nugget in $\mathcal{N}$ has an associated relevance/importance grade,

$$\mathbf{R} : \mathcal{N} \rightarrow [0, 1] \tag{2}$$

  This measures the importance of the update content (information) in a summarization system.

- **Relevance Discounts** In order to capture timeliness and conciseness of our system, we introduce two ways in which relevance might be discounted.
  - **Latency Discount** Given a reference timestamp of a matching nugget, t^*, a latency penalty function $\mathbf{L}(t^*, t)$ is a monotonically *decreasing* function of $t - t^*$.
  - **Verbosity Normalization** Our problem definition assumes that a user receives a stream of updates from the system. Consequently, we want to penalize systems which include unreasonably long updates which can be used to introduce, for example, higher reading effort. It can be defined as a string length penalty function, monotonically increasing in the number of words of the update *string*. We will refer to this normalization function as $\mathbf{V}(u)$.

- **Discounted Gain** Given an update u and a matching nugget n (i.e. $u \approx n$), we can define the discounted gain as,

$$\mathbf{g}(u, n) = \mathbf{R}(n) \times \text{discounting factor} \tag{3}$$

  Given the previously defined discounts, we have the following family of discounted gains,

$$\mathbf{g_F}(u, n) = \mathbf{R}(n) \qquad \text{discount-free gain} \tag{4}$$
$$\mathbf{g_L}(u, n) = \mathbf{R}(n) \times \mathbf{L}(n.t, u.t) \qquad \text{latency-discounted gain} \tag{5}$$
$$\tag{6}$$

- **Update-Nugget Matching** We also define a very important *earliest matching function* between a nugget and an update set,

$$\mathbf{M}(n, \mathcal{S}) = \text{argmin}_{\{u \in \mathcal{S} : n \approx u\}} u.t \tag{7}$$

  or $\emptyset$ if there is no matching update for n. $\mathbf{M}$ should be interpreted as "given n, the earliest matching update in $\mathcal{S}$."

  We also define the set of nuggets for which u is the earliest matching update as,

$$\mathbf{M}^{-1}(u, \mathcal{S}) = \{n \in \mathcal{N} : \mathbf{M}(n, \mathcal{S}) = u\} \tag{8}$$

  Note that an update can be the earliest matching update for more than one nugget.

2.2 Metrics

2.2.1 Update Gain

Since an update can be the earliest to match several nuggets ($u \approx n$), we define the gain of an update with respect to a system/run $\mathcal{S}$ as the sum of latency-discounted relevance of the nuggets for which it is the earliest matching update:

$$\mathbf{G}(u, \mathcal{S}) = \sum_{n \in \mathbf{M}^{-1}(u, \mathcal{S})} \mathbf{g}(u, n) \tag{9}$$

where the gain can be any of the discounted gains described earlier. Note that for an appropriate discounting function, $\mathbf{G}(u, \mathcal{S}) \in [0, 1]$.

2.2.2 Expected Gain

One way to evaluate an update system is to measure the expected gain for a system update. This is similar to traditional notions of precision in information retrieval evaluation.

Over a large population of system updates, we can estimate this reliably. In order to compute the expected update gain for system $\mathcal{S}$ by time τ, we

$$\begin{aligned}\mathbf{EG}(\mathcal{S}) &= \frac{1}{|\mathcal{S}|}\sum_{u\in\mathcal{S}}\mathbf{G}(u,\mathcal{S}) && (10)\\ &= \frac{1}{|\mathcal{S}|}\sum_{u\in\mathcal{S}}\sum_{n\in\mathbf{M}^{-1}(u,\mathcal{S})}\mathbf{g}(u,n)\\ &= \frac{1}{|\mathcal{S}|}\sum_{\{n\in\mathcal{N}:\mathbf{M}(n,\mathcal{S})\neq\emptyset\}}\mathbf{g}(\mathbf{M}(n,\mathcal{S}),n) && (11)\end{aligned}$$

Additionally, we may penalize "verbosity" by normalizing not by the number of system updates, but by the overall verbosity of the system

$$\mathbf{EG_V}(\mathcal{S}) = \frac{1}{\sum_{u\in\mathcal{S}}\mathbf{V}(u)}\sum_{\{n\in\mathcal{N}:\mathbf{M}(n,\mathcal{S})\neq\emptyset\}}\mathbf{g}(\mathbf{M}(n,\mathcal{S}),n) \quad (12)$$

We will adopt a definition of $\mathbf{g}$ such that evaluation that,

- does not penalize for large a update matching several nuggets, as opposed to a few small updates matching a nugget each due to verbosity weighting,
- penalizes for late updates (against matched nugget reference timestamp), and
- penalizes "verbosity" of updates text not matching any nuggets.

Furthermore, we will guarantee that $\mathbf{G}(u,\mathcal{S}_\tau)\in[0,1]$. Over a set of events, the mean expected gain is defined as,

$$\mathbf{MEG} = \frac{1}{|\mathcal{E}|}\sum_{\epsilon\in\mathcal{E}}\mathbf{EG}(\mathcal{S}^\epsilon) \quad (13)$$

where $\mathcal{E}$ is the set of evaluation events and $\mathcal{S}^\epsilon$ is the system submission for event ϵ.

Because a user interest may be concentrated immediately after an event and because a system's performance (in terms of gain) may be dependent on the time after an event, we will also consider a time-sensitive version of expected gain for the first τ seconds,

$$\mathbf{EG}_\tau(\mathcal{S}) = \mathbf{EG}(\mathcal{S}_\tau) \quad (14)$$

with $\mathbf{MEG}_\tau$ defined similarly.

2.2.3 Comprehensiveness

In addition to good expected gain, we are interested in a system providing comprehensive updating. That is, we would like the system to cover as many nuggets as possible. This is similar to traditional notions of recall in information retrieval evaluation.

Given a set of system updates, $\mathcal{S}$, we define the comprehensiveness of the run as,

$$\mathbf{C}(\mathcal{S}) = \frac{1}{\sum_{n \in \mathcal{N}} \mathbf{R}(n)} \sum_{\{n \in \mathcal{N}: \mathbf{M}(n,\mathcal{S}) \neq \emptyset\}} \mathbf{g}(\mathbf{M}(n,\mathcal{S}), n) \tag{15}$$

$$= \frac{1}{\sum_{n \in \mathcal{N}} \mathbf{R}(n)} \sum_{u \in \mathcal{S}} \sum_{n \in \mathbf{M}^{-1}(u,\mathcal{S})} \mathbf{g}(u, n)$$

$$= \frac{1}{\sum_{n \in \mathcal{N}} \mathbf{R}(n)} \sum_{u \in \mathcal{S}} \mathbf{G}(u, \mathcal{S}) \tag{16}$$

where $\mathbf{g}$ may or may not be the same gain function used for the expected update gain.

We also define a time-sensitive notion of comprehensiveness,

$$\mathbf{C}_\tau(\mathcal{S}) = \mathbf{C}(\mathcal{S}_\tau) \tag{17}$$

with an aggregated measure defined as,

$$\int_{t_s}^{t_e} \mathbf{C}_\tau(\mathcal{S}) d\tau \tag{18}$$

which measures how quickly a system captures nuggets.

2.3 Values used in Evaluation

$n.t$ - Nugget time (time of Wikipedia edit from which nugget was extracted)
$n.i$ - Nugget importance (assigned by assessor)
$|u|$, $|n|$ - Length of update, or nugget, (in words)

2.3.1 Normalized Nugget Relevance

Nugget importance was provided on a 0-3 scale by assessors (no importance to high importance). For graded relevance, we normalize on an exponential scale, since high importance nuggets are described as "of key importance to the query", whereas low importance nuggets are "of any importance to the query". When binary relevance is needed, everything of any relevance is relevant (0 is the only non-relevant grade).

$$\mathbf{R}_{\text{graded}}(n) = \frac{e^{n.i}}{e^{\max_{n' \in \mathcal{N}} n'.i}} \qquad \text{Graded relevance} \tag{19}$$

$$\mathbf{R}_{\text{binary}}(n) = \begin{cases} 1 & \text{iff } n.i > 0 \\ 0 & \text{otherwise} \end{cases} \qquad \text{Binary relevance} \tag{20}$$

$$\tag{21}$$

Note that for graded relevance, returning exactly the nugget set as the system output updates and nothing more ("perfect system"), would usually *not* result in an expected gain of 1. However, using binary relevance, the perfect system would score an expected gain of 1.

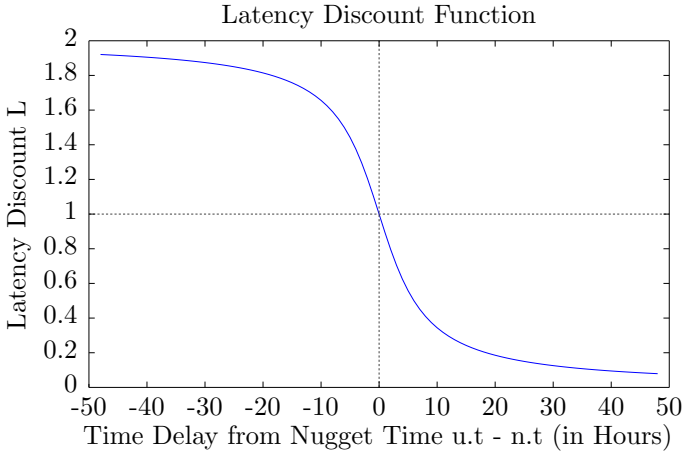
2.3.2 Latency Discount Factor

Given that a system may return an update matching Wikipedia information before the Wikipedia information exists, we use a function that is smooth and decays on both the positive and negative sides. The current

parameters allow the latency discount factor to vary from 0 to 2 (1 means nugget time equal to update time), and flattens at around one day(± 24 hours). Note that as a result, gain and expected gain can be greater than 1.

$$\mathbf{L}(n.t, u.t) = 1 - \frac{2}{\pi}\arctan(\frac{u.t - n.t}{\alpha}) \quad \text{latency-discount} \tag{22}$$

$$\alpha = 3600 * 6 \quad \text{latency-step (6 hours)} \tag{23}$$



2.3.3 Verbosity Normalization Factor

For verbosity, we approximate the number of extra nuggets worth of information in a given update. This is done by finding all text which did not match a nugget (as defined by the assessors), and dividing the number of words in the text by the average number of words in a nugget for that query.

$$\mathbf{V}(u) = 1 + \frac{|all\_words_u| - |nuggetmatching\_words_u|}{AVG_n|words_n|} \tag{24}$$

$$= 1 + \frac{|u| - |\cup_{n \in \mathbf{M}^{-1}(u,\mathcal{S})} \mathbf{M}(n, \mathcal{S})|}{\text{avg}_{n \in \mathcal{N}}|n|} \quad \text{verbosity-normalization} \tag{25}$$

Note that if an update has all its words part of some match to a nugget, the verbosity is $V(u)$=1; otherwise $V(u) - 1$ is an approximation of the "extra non-matching words" in terms of equivalent number of nuggets.

3 Value Tracking

3.1 Notation

Let $\mathcal{V}$ be the set of possible values deliverable to the user. A value is either a real number or a pair of numbers, depending on the type. An value update u refers to a timestamped value indexed such that $u_v \in \mathcal{V}$ and u_t is the timestamp of the update.

Given a set of value updates $\mathcal{U}$ for an event, we assume the predicted value at time τ is the value of the most recent update before τ. We represent this as $f_{\mathcal{U}}(\tau) \in \mathcal{V}$. In order for this function to be well-defined, we request that participants provide a baseline or prior prediction before having seen any evidence.

Given an event and an attribute type, our manual annotation process generates a set of timestamped target attribute values $\mathcal{U}^*$. In most cases, the annotation process will generate a single target value (i.e. $f_{\mathcal{U}^*}(\tau)$ is constant for all τ).

3.2 Metrics

We evaluate a system according to the expected error with respect to the true attribute value $f_{\mathcal{U}^*}$,

$$\mathbf{EE}(\mathcal{U},\mathcal{U}^*,\tau) = \frac{1}{t_e - t_s}\int_{t_s}^{t_e} \mathbf{Err}(\mathcal{U},\mathcal{U}^*,\tau)d\tau \tag{26}$$

where, for scalar attributes, the error is defined as

$$\mathbf{Err}(\mathcal{U},\mathcal{U}^*,\tau) = |f_{\mathcal{U}^*}(\tau) - f_{\mathcal{U}}(\tau)| \tag{27}$$

and for geographic attributes, we use the Vincenty distance.