Online Sentence Novelty Scoring for Topical Document Streams

Sungjin Lee
Yahoo Labs
229 West 43rd Street, New York, NY 10036, USA
junion@yahoo-inc.com

Abstract

The enormous amount of information on the Internet has raised the challenge of highlighting new information in the context of already viewed content. This type of intelligent interface can save users time and prevent frustration. Our goal is to scale up novelty detection to large web properties like Google News and Yahoo News. We present a set of light-weight features for online novelty scoring and fast nonlinear feature transformation methods. Our experimental results on the TREC 2004 shared task datasets show that the proposed method is not only efficient but also very powerful, significantly surpassing the best system at TREC 2004.

1 Introduction

The Internet supplies a wealth of news content with a corresponding problem: finding the right content for different users. Search engines are helpful if a user is looking for something specific that can be cast as a keyword query. If a user does not know what to look for, recommendation engines can make personalized suggestions for stories that may interest the user. But both types of systems frequently represent content that the user has already consumed, leading to delay and frustration. Consequently, identifying novel information has been an essential aspect of studies on news information retrieval. Newsjunkie (Gabrilovich et al., 2004), for instance, describes a system that personalizes a newsfeed based on a measure of information novelty: the user can be presented custom tailored news feeds that are novel in the context of documents that have already been reviewed. This will spare the user from hunting through duplicate and redundant content for new nuggets of information. Identifying genuinely novel information is also an essential aspect of update summarization (Nenкова and McKeown, 2012; Gao et al., 2013; Guo et al., 2013; Wang and Li, 2010; Bentivogli et al., 2011). But the temporal dynamics of a document stream are not generally the focus. Novelty detection has also been studied in Topic Detection and Tracking field for the First Story Detection task (Allan, 2002; Karkali1 et al., 2013; Karkali et al., 2014; Tsai and Zhang, 2011) where the aim is to detect novel documents given previously seen documents. In this paper, we examine a slightly different problem: we perform novelty detection at the sentence level to highlight sentences that contain novel information.

The novelty track in TREC was designed to serve as a shared task for exactly this type of research: finding novel, on-topic sentences from a news stream (Harman, 2002). There were four tasks in the novelty track but we only focus on task 2 in this paper: “given relevant sentences in all documents, identify all novel sentences.” The track changed slightly from year to year. The data of the first run in 2002 (Harman, 2002) used old topics and judgments which proved to be problematic due to the small percentage of relevant sentences. TREC 2003 (Soboroff and Harman, 2003) included 50 new topics with an improved balance of relevant and novel sentences and chronologically ordered documents. TREC 2004 (Soboroff and Harman, 2005) used the same task settings and the same number of topics, but made a major change through the inclusion of irrelevant documents.

Although the participants in the novelty track of TREC and many followup studies have investigated a wide ranging set of features and algorithms (Soboroff and Harman, 2005), almost none were specifically focused on scalability. However, modern news aggregators are usually visited by millions of unique users and consume millions of stories each day. Moreover, every few minutes item churn takes place and the stories of interest are likely to be the ones that appeared in the last couple of hours. As real-time processing on a large scale gains more attention (Osborne et al., 2014), we investigate features that are both effective and efficient, and so could be used in a scalable online novelty scoring engine for making personalized newsfeeds on large web properties like Google News and Yahoo News.

To achieve this goal, our contributions are two-fold. First, we present a set of effective, light-weight features: KL divergence with asymmetric smoothing, nonlinear transformation of unseen word count, relative sentence position and word embedding-based similarity. Note that we restrict ourselves to only surface-level text features and algorithms that have time complexity of $O(W)$ where $W$ is the number of unique words seen so far (previous studies often employed quite expensive
features and algorithms that have time complexity of at least \(O(WT)\) where \(T\) is the number of sentences so far. To fully comply with the online setting, we also exclude very popular methods for measuring similarity such as tf-idf, since we are not allowed to see the entire corpus. Second, we propose efficient feature transformation methods: recursive feature averaging and Deep Neural Network (DNN)-based nonlinear transformation. We evaluate our system on task 2 of the 2004 TREC novelty track. Interestingly, our experiment results indicate that our light-weight features are actually very powerful when used in conjunction with the proposed feature transformation; we obtain a significant performance improvement over the best challenge system.

The rest of this paper is structured as follows: Section 2 presents a brief summary of related work. Section 3 describes our algorithm and features. Section 4 outlines the experimental setup and reports the results of comparative analysis with challenge systems. We finish with some conclusions and future directions in Section 5.

2 Related Work

There were 13 groups and 54 submitted entries for the 2004 TREC novelty track task 2. The participants used a wide range of methods which can be roughly categorized into statistical and linguistic methods. Statistical methods included traditional information retrieval models such as tf-idf and Okapi, and metrics such as importance value, new sentence value, conceptual fuzziness, scarcity measure, information weakness, unseen item count with a threshold optimized for detecting novel sentences (Blott et al., 2004; Zhang et al., 2004; Abdul-Jaleel et al., 2004; Eichmann et al., 2004; Erkan, 2004; Schiffman and McKeown, 2004). Thresholds are either learned on the 2003 data or determined in an ad hoc manner. Some groups also used machine learning algorithms such as SVMs by casting the problem as a binary classification (Tomiyma et al., 2004). Many groups adopted a variety of preprocessing steps including expansion of the sentences using dictionaries, ontologies or corpus-based methods and named entity recognition. Graph-based analysis has also been applied where directed edges are established by cosine similarity and chronological order. After this graph is constructed, the eigenvector centrality score for each sentence was computed by using a power method. The sentences with low centrality scores were considered as new (Erkan, 2004). Graph-based approaches were further pursued by Gamon (2006) that drew a richer set of features from graph topology and its changes, resulting in a system that ties with the best system at TREC 2004 (i.e. Blott et al. (2004)). On the other hand, deep linguistic methods included parsing, coreference resolution, matching discourse entities, searching for particular verbs and verb phrases, standardizing acronyms, building a named-entity lexicon, and matching concepts to manually-constructed ontology for topic-specific concepts (Amrani et al., 2004). The difficulty of the novelty detection task is evident from the relatively low score achieved by even the best systems at TREC 2004 (Soboroff and Harman, 2005). The top scoring systems were mostly based on statistical methods while deep linguistic approaches achieved the highest precision at the cost of poor recall.

3 Method

For the purpose of this paper, we formulate task 2 of the TREC novelty detection track as an online probabilistic inference task. More specifically, we compute the novelty score as the posterior probability of a binary novelty random variable \(N_t\):

\[
p(N_t|S_t, C_{t-1}) = \frac{1}{Z} \exp \sum_i w_i f_i(N_t, S_t, C_{t-1})
\]

in which the \(f_i\) are feature functions, \(w_i\) model parameters, \(S_t\) the sentence in focus and \(C_{t-1}\) a context containing information about previously seen sentences \(S_i\) through \(S_{t-1}\) across documents.

The overall procedure is listed in Algorithm 1. The algorithm takes as input documents which have been clustered by topic and chronologically ordered. For each sentence \(S_t\) in each document, basic preprocessing is performed (e.g. simple tokenization, stopword filtering and stemming (Porter, 1980)), then the inference is made whether \(S_t\) is novel given the context \(C_{t-1}\). Without the use of the context, the time complexity of our algorithm would depend on the number of sentences so far. Thus, the features and the model for the context are important for efficiency. Note that our method only takes time complexity of \(O(W)\) for

---

**Algorithm 1:** Novelty scoring for a topical document stream

**Data:** a document stream

**Result:** a document stream with novelty annotation

Initialize a context \(C_0\);

while not at end of the document stream do
  read a document;
  split the document into sentences;
  while not at end of the document do
    read a sentence \(S_t\);
    perform preprocessing on \(S_t\);
    compute novelty score as the posterior probability of a binary novelty random variable \(N_t\), \(p(N_t|S_t, C_{t-1})\);
    update the context \(C_t\) with \(C_{t-1}\) and \(S_t\);
  end
  compute a document-level score (e.g. average out all sentence-level scores)
end
both context update and feature generation.

3.1 Features

**KL divergence with asymmetric smoothing.** KL divergence has been successfully adopted to measure the distance between a document and a set of documents (Gabrilovich et al., 2004; Gamon, 2006). We use it to measure the distance between context $C$ and sentence $S$:

$$\sum_w p_C(w) \log \frac{p_C(w)}{p_S(w)}$$

The intuition is that the more distant the distributions are, the more likely it is that the sentence is novel. Since KL divergence is asymmetric, both directions are used as features, with and without scale normalization. The computation of KL divergence requires both $p_C$ and $p_S$ to be non-zero; while simple add-one smoothing is employed in previous work, we adopt novel asymmetric smoothing. We add a larger smoothing factor $s$ for already seen words than the factor $u$ for unseen words. The rationale behind this is that we intensify the difference caused by unseen words and attenuate the difference caused by seen words (Figure 1.). Asymmetric smoothing with various smoothing factors consistently showed better performance than symmetric smoothing in our experiments.

![Figure 1: KL divergence with symmetric (left) and asymmetric (right) smoothing. Pink and blue correspond to two distributions while light yellow and orange to smoothing factors.](image)

**Nonlinear transformation of unseen word count.** One of the simplest metrics to measure novelty is the plain count of unseen words. This measure, however, does not necessarily reflect human perception of novelty given the prevalence of nonlinearity in human perception (Kingdom and Prins, 2009). Thus, we explored the use of a simple nonlinear transformation of unseen word counts instead of the plain count (Figure 2):

$$T(n) = (\alpha n + \beta)^\gamma$$

where $n$ is the number of new words and $\alpha$, $\beta$ and $\gamma$ are parameters. In our experiments, the use of a nonlinear transformation helped yield better results.

![Figure 2: Nonlinear transformation of unseen word count with parameters set via cross-validation on the TREC training data: $\alpha = 0.5, \beta = 0$ and $\gamma = 1.5$.](image)

**Relative position in a document.** Relative position of a sentence in a document is simple yet has been proven effective for summarization. Relative position is also closely related to novelty detection as follows: 1) There is in general a good chance that earlier sentences are more novel than the later ones. 2) We found a pattern that news articles coming in later are apt to present novel information first and then a summary of old information.

**Word embedding-based similarity.** Neural word embedding techniques can be effective in capturing syntactic and semantic relationships, and more computationally efficient than many other competitors (Socher et al., 2012; Mikolov et al., 2013). As reported in (Tai et al., 2015), a simple averaging scheme was found to be very competitive to more complex models for representing a sentence vector. These observations lead us to adopt the following additional features derived from word embeddings: 1) cosine similarity between the mean vectors of the context $C$ and sentence $S$, 2) sigmoid function value for the dot product of the mean vectors of the context $C$ and sentence $S$. The mean vectors of $C$ and $S$ are computed by taking the average of the word vectors of each unique word in $C$ and $S$, respectively. We use word embedding with 100 dimensions trained on Wikipedia using the word2vec toolkit (https://code.google.com/p/word2vec).

3.2 Feature transformation

**Recursive feature averaging.** A large portion of the novel sentences in the TREC 2004 data appear in consecutive runs of two or more (Schiffman and McKeown, 2004). Sequential labeling would be a natural approach to take advantage of this characteristic of the problem, but the use of sequential labeling will make time complexity depend on the number of sentences $T$. Thus we came up with another way to exploit this characteristic, recursively averaging over previous feature vectors and augmenting the current feature vector with
the average:

\[ R_t = \eta F_{t-1} + (1 - \eta) R_{t-1} \quad (4) \]

\[ F'_t = F_t :: R_t \quad (5) \]

where \( F \) is a feature vector, \( R \) the average vector of previous ones, \( F' \) the augmented feature vector, \( \eta \) the weight of the last feature vector in averaging and :: means concatenation.

**DNN-based feature transformation.** In order to better capture non-trivial interactions between the features described above, we adopt a DNN with a bottleneck. DNNs with a bottleneck have been successfully explored for nonlinear feature transformation (Grézl et al., 2007). The feature transformation is normally achieved from narrow hidden layers that retain only the information useful to classification. This leads us to introduce bottleneck hidden layers between the input layer and the Logistic Regression output layer (Figure 3.)

![Flowchart for a bottleneck DNN. The dotted-box represents bottleneck generating hidden layers.](image)

**4 Experiments and Results**

Following the guidelines of task 2 for the TREC 2004 novelty detection track, we used the TREC 2003 dataset as training data and the TREC 2004 dataset as test data. The training data includes 10,226 novel sentences out of 15,557 sentences. The test data includes 3,454 novel ones out of 8,343 sentences. We trained a DNN-based classifier and several logistic regression classifiers (which are the same model with the DNN model except without the hidden layers) using the Theano toolkit (Bergstra et al., 2010) to verify the effectiveness of each feature and feature transformation. We optimized all models by minimizing logloss with the stochastic gradient decent algorithm with momentum. We classified a sentence as novel if the posterior probability is greater than 0.5. We performed a search based on five-fold cross validation to identify optimal values for the parameters defined in Section 3, and obtained the following values: \( s = 10, u = 0.1, \alpha = 0.5, \beta = 0, \gamma = 1.5 \) and \( \eta = 0.5 \). For the DNN classifier, we used a set of five bottleneck hidden layers. The number of nodes for each hidden layer were set to 10, 5, 3, 5 and 10, respectively.

Comparative evaluation results in F-score (following the TREC protocol) are shown in Table 1. In Table 1, the first four entries refer to the best top systems from TREC 2004 and followup studies, KLdiv to a system using only KL divergence features, TransCount to a system using only nonlinear transformation of unseen word count features, RelPos to a system using only relative position features, Word2Vec to a system using only word embedding features, All to a system using all features, All + Recursive to All with recursive feature averaging applied, All + DNN to All with DNN-based feature transformation applied and All + Recursive + DNN to All + Recursive with DNN-based feature transformation applied. The best result (in bold) is significantly better than the best system results from TREC 2004, while still being very computationally efficient and therefore scalable. In terms of individual features, KLdiv (ties for 5th place at TREC 2004) and TransCount (outperforms the 6th entry) showed very strong results. Although RelPos and Word2Vec did not yield good results, we found them complementary to other features; performance was degraded to 0.621 and 0.624, respectively, when they were excluded from All + Recursive + DNN. The DNN-based feature transformation generally yielded better results. In particular, it becomes very effective in conjunction with recursive feature averaging. This result indicates that the DNN-based transformation allows the system to capture the non-trivial interactions between previous sentences and the current one.

<table>
<thead>
<tr>
<th>Systems</th>
<th>F-score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blott et al. (2004) / Gamon (2006)</td>
<td>0.622</td>
</tr>
<tr>
<td>Tomiyama et al. (2004)</td>
<td>0.619</td>
</tr>
<tr>
<td>Abdul-Jaleel et al. (2004)</td>
<td>0.618</td>
</tr>
<tr>
<td>Schiffman and McKeown (2004)</td>
<td>0.617</td>
</tr>
<tr>
<td>KLdiv</td>
<td>0.614</td>
</tr>
<tr>
<td>TransCount</td>
<td>0.611</td>
</tr>
<tr>
<td>RelPos</td>
<td>0.577</td>
</tr>
<tr>
<td>Word2Vec</td>
<td>0.577</td>
</tr>
<tr>
<td>All</td>
<td>0.615</td>
</tr>
<tr>
<td>All + Recursive</td>
<td>0.615</td>
</tr>
<tr>
<td>All + DNN</td>
<td>0.617</td>
</tr>
<tr>
<td>All + Recursive + DNN</td>
<td><strong>0.625</strong></td>
</tr>
</tbody>
</table>

Table 1: Performance breakdown. The best result is significantly better than the other configurations \((p < 0.01)\) based on the McNemar test. Since the systems’ output is not available, we are not able to calculate statistical significance against TREC systems.

**5 Conclusions**

We explored the space of light-weight features and their nonlinear transformation with the goal of supporting online web-scale sentence novelty detection. The experiment results show that these features are not only efficient but also very powerful; a combination of these
features with a simple, scalable classification approach significantly surpassed the best challenge system at TREC 2004. For future work, it would be interesting to see if more sophisticated DNN training techniques (e.g. unsupervised pre-training and different optimization algorithms) would yield a better performance.

References


Ahmed Amrani, Jérôme Azé, Thomas Heitz, Yves Kodratoff, and Mathieu Roche. 2004. From the texts to the concepts they contain: a chain of linguistic treatments. In Proceedings of TREC.


