Embedded Entity Representations for Vital Filtering on Large Corpora

ABSTRACT

The task of identifying documents that contain vital information for an entity of interest, known as *vital filtering*, has become increasingly relevant as the corpora available for extraction grow in size. To efficiently explore large corpora of text documents, we need to identify references to entities of interest as well as studying their topics evolution over time. Existing approaches are limited; they are unable to handle streaming data, do not partition entities' references according to topics, and do not accurately estimate temporal vitalness. In this paper we introduce a embedding-based non-parametric representation of entities that addresses the above limitations. To efficiently handle lexical sparsity, we propose using word embedding representation of the contexts. The entity contexts are described by topic clusters that are estimated in a non-parametric manner. Further, we associate a staleness measure to each entity and topic cluster, dynamically estimating their temporal relevance. This approach of using distributed word embeddings, non-parametric clustering, and staleness provides an efficient yet appropriate representation of entity contexts for streaming settings, facilitating accurate vital filtering. Experiments on the TAC KBA vital filtering task evaluate the utility of our contributions, demonstrating significant gains over existing approaches.

1. INTRODUCTION

To extract relevant information from streaming text corpora, we often need to find references to entities of interest, and study their topic trends over time. This is, unfortunately, an incredibly difficult task, and thus a large number of pertinent articles are seldom retrieved by automated approaches. As a consequence, Frank et al. [8] observe a considerable lag between the publication date of articles and the date of their citations in Wikipedia. The median time is over a year, and the distribution has a long and heavy tail. This gap can be drastically reduced if automated systems can accurately and efficiently suggest relevant documents to editors as soon as they are published.

Recent submissions to TREC KBA [16, 4, 6, 26, 2] center their attention on solving the above problems with supervised methods, using mainly document, document-entity and temporal level features. They are, however, somewhat limited; they depend heavily on labeled data, do not handle context sparsity appropriately, and do not identify the topics in the references.

In this submission, we introduce a semi-supervised approach suitable for streaming settings that uses low-dimensional vectors and non-parametric topic clusters to represent entities' contexts. We also include a staleness measure that approximates the relevance of the topic clusters. Further, we update the topic identities, number of topics, and the entities and topics staleness in an online fashion, observing only a single document at a time. To utilize labeled data, we add features based on our representation of unlabeled documents to a supervised classifier.

This combination of distributed word embeddings, non-parametric clustering, and staleness measure provides an efficient yet accurate representation of entities' contexts that can be updated in a streaming manner, thus addressing the document filtering requirements on large streams of text.

We present experimental results that demonstrate the benefits of our method and present our performance on the UW TREC KBA 2014 Vital Filtering task. As part of the Accelerate and Create task, we also describe an exploratory tool for efficient and intuitive visualization of large streams.

2. VITAL FILTERING TASK

In this section, we formalize the problem setup and introduce our notation. We assume a set of m target entities $E = \{e_1, ..., e_m\}$. We further assume a set of n documents $D = \{d_1, ..., d_n\}$ that arrive in chronological order.

Each document is a sequence of sentences composed by collections of words, annotated with NLP tools. Further, we assume w.l.o.g. that every document in D refers to a single entity $e \in E$. Since our focus here is to distinguish vital and non-vital references, we use a naive classifier based resolution to identify the documents relevant to each entity (details in Section 5.4), although in practice this is a challenging task and more sophisticated techniques are required [20, 22].

A mention to e in a document $d_i \in D$ is identified by a string matching algorithm that searches for exact matches of canonical and surface form names of the entity e. We represent each d_i as a compound of a timestamp t_i and a bag of words $W_i = \{w_{i1}, ..., w_{ip}\}$ located in the context of (and including) mentions to the entity e. Finally, we assume an online setting, i.e. the algorithm should provide predictions for documents arriving at time t before seeing any documents arriving at time $t+1$.

Given this setup, the vital filtering task requires classification of each document d relevant to an entity e as follows:

- *Vital* if the document contains information that, at the time it enters the stream, would cause an update to the entity e with timely, new information about the entity's current state, actions or situation, e.g. "Barack Obama has been elected President".

- *Non-Vital* if the document is relevant, but contains information that is not timely, i.e. it may contain information relevant when building an initial profile of the entity e, but does not contain information that an accurate, updated profile would not have, e.g. "Barack Obama was born on August 4th, 1961".

3. PROPOSED APPROACH

Given a stream of documents D that refer to entity e, the task at hand is to predict whether those documents are *vital* or *non-vital* to e. To detect whether a document contains novel information, one needs to provide an accurate and generalizable representation of historical contexts and capture the temporal dynamics of the references.

To this end, we propose a three-pronged solution: (1) represent documents with low-dimensional embeddings that address sparsity and generalization (Section 3.1), (2) represent the entity's context using non-parametric topic clustering (Section 3.2), and (3) estimate the novelty of the document information using a staleness measure (Section 3.3).

3.1 Distributed Document Representation

To identify whether an entity's context in a document contains novel information, or even if it is relevant for the entity, we need a structured representation of the context. A common solution to this problem is to use vector space models, often the Bag of Words (BOW) models, where a document is represented as the bag of its words, disregarding grammar and even word order. Unfortunately, vector space models are often too sparse to represent fine-grained information in contexts, for example, straightforward BOW representations will have minimal overlap between "Barack was elected president today" and "Obama has won the election", treating the other as novel information even after having seen one of them. Further, BOW representations ignore the syntactic structure as it will further increase sparsity, for example, BOW will have a high overlap for "Obama defeated Bush" and "Bush defeated Obama".

In order to address these concerns, we propose to represent contexts of entities in documents using word embeddings. A word embedding is a dense, low-dimensional, and real-valued vector associated with every word in a vocabulary such that they capture useful syntactic and semantic properties of the contexts that the word appears in. The low-dimensionality of the embeddings as compared to vector space models (hundreds as compared to millions) make them an elegant solution to address lexical sparsity in settings with very few labels [24], and further, they can be efficiently trained on massive corpora. Many of the syntactic patterns can be represented with simple algebraic operations. For example, the result of $v_{paris} - v_{france} + v_{germany}$ is closer to v_{berlin} than to any other word vector [17, 18].

We define a function $f : w \rightarrow v_w \in \mathbf{R^d}$ that computes the word embedding representation of the word string w. To define embedding for a set of words W, we use a function $g : W \rightarrow v_W \in \mathbf{R^d}$ that computes embeddings as:

$$g(W) = v_W = \frac{1}{|W|} \sum_{w \in W} f(w) \tag{1}$$

Given the document $d_i \in D$ that refers to entity e and contains the words $w_i \in W_i$, we compute its vector representation using function g as follows:

$$v_{d_i} = v_{W_i} = g(W_i) \tag{2}$$

With this, we intend to capture the context where the entity e is mentioned in a document, i.e. the topic, and represent it with a dense, low-dimensional vector.

Further, it may be useful to separately capture the context in terms of different parts of speech. Let W_{i_n} denote the set of all nouns in W_i, where $W_{i_n} \subseteq W_i$, and W_{i_v} to the set of all verbs in W_i, where $W_{i_v} \subseteq W_i$. We compute the embedding vector of all the nouns and verbs that appear in the context of entity e using function g, as:

$$v_{d_{i_n}} = v_{W_{i_n}} = g(W_{i_n}) \tag{3}$$

$$v_{d_{i_v}} = v_{W_{i_v}} = g(W_{i_v}) \tag{4}$$

Computing separate embeddings for different word types is a flexibility our method provides that may better encapsulate the underlying context (topic) of the document.

3.2 Non-parametric Clustering

Although word embeddings can capture context around a single topic quite accurately, they are unable to represent the variety of topics that an entity may be mentioned in. For example, Obama in the context of *elections* is quite different to Obama in the context of *presidential speech* or *international visit*. Using a single word embedding to represent multiple such topics may result in embeddings that conflate them, being inaccurate for all of them.

One typical approach to tackle this problem is using topic models [3]. Such models can be trained in an offline manner over a large corpus, followed by streaming inference for each document. However, the number of topics often needs to be decided apriori, which is quite difficult to specify for each entity of interest (non-parametric approaches to LDA are quite expensive). Further, drift over time can make the topic distributions obsolete. Finally, it is difficult to learn per entity topic distributions if some of the entities have very few relevant documents.

Instead of representing the context using only a single embedding, we propose to use a number of embeddings that capture the different *topic clusters* of the entity, thus retaining the advantages of using word embedding while still having a precise context representation. We assume that the context in a single document belongs to a single topic, though we dynamically estimate the number of clusters in a non-parametric manner. As we are concerned with a streaming setting, topic clusters evolve over time, i.e. identities, members and number of clusters change over time.

We represent each topic cluster by the mean embedding vector of the documents assigned to that cluster at a certain timestamp. More precisely, the vector representation of the

Algorithm 1 Non-parametric Clustering

Input: $doc = doc\_word\_embedding(d, e, c)$, and topic clusters list tcs for entity e
Output: new or updated topic cluster tc
Body:
Initialize $tc = nil$
if tcs **is empty then**
  $tc = create\_topic\_cluster(center{=}doc)$
  $tcs.append(tc)$
else
  $dist, i{=}\min_{\forall i \in tcs} cosine\_dist(tcs[i].center, doc)$
  if $dist >= \alpha$ **then**
    $tc = create\_topic\_cluster(center{=}doc)$
    $tcs.append(tc)$
  else
    $tc = update\_topic\_cluster(tcs[i], doc)$
  end if
end if
return tc

j-th topic cluster at timestamp t_i, c_i^j, can be computed using:

$$v_{c_i^j} = \frac{1}{|D_i^j|} \sum_{d \in D_i^j} v_d \quad (5)$$

where D_i^j is the subset of all the documents that belong to cluster j at timestamp t_i, and $\forall\ d_q \in D_i^j, t_q \leq t_i$.

The number of topic clusters for the context of entity e is unknown beforehand. Initially, we let the entity's context to have zero topic clusters. We create the first topic cluster for the entity's context when the first relevant document is observed. For any following relevant document d, the topic clusters are updated as follows. We first compute a distance of v_d with every existing topic embedding. If the minimum distance to any topic cluster is greater than or equal to α ($0 \leq \alpha \leq 1$), we create a new topic cluster just containing document d, otherwise we merge document d into the closest cluster to v_d, and update the cluster's vector representation. Our approach is closely related to the online non-parametric clustering procedure described in Neelakantan et al. [19].

More formally, $\forall\ c_{i-1}^j$, at time i, document d_i is added to the topic cluster that solves the following optimization problem:

$$\begin{aligned} &\arg\min_j \ dist(v_{d_i}, v_{c_{i-1}^j}) \\ &\text{subject to } dist(v_{d_i}, v_{c_{i-1}^j}) < \alpha \end{aligned} \quad (6)$$

where $dist(\cdot, \cdot)$ is the cosine distance defined as:

$$dist(x, y) = 1 - \cos(x, y) = 1 - \frac{x \cdot y}{||x||||y||} \quad (7)$$

The j-th topic cluster at time i is updated, and therefore composed by the subset of documents $D_i^j \subseteq D$, where $D_i^j = D_{i-1}^j \cup \{d_i\}$. Note that the cluster center is updated in constant time by incrementally maintaining the sum of the member embeddings.

Figure 1 illustrates an example of such clustering, using two-dimensions to represent the vectors. Let's assume document d_1 appears in the stream first, and mentions the days Barack Obama was a senator. As it is the first document referring to the entity Obama, we add a new topic cluster *senator* with vector v_{d_1}. Then, document d_2 appears in the stream, and refers to Obama as being elected President of the United States. The distance with the previous cluster *senator* is greater than α due to semantic difference in the words, therefore the algorithm proceeds to create a new topic cluster *president* centered at v_{d_2}. Finally, d_3 enters the stream. It talks about Obama as the current President of the U.S. The algorithm compares its distance to the previous clusters and finds that it is closest to the *president* cluster. The distance is less than α, hence it adds d_3 to the *president* cluster and updates the cluster center.

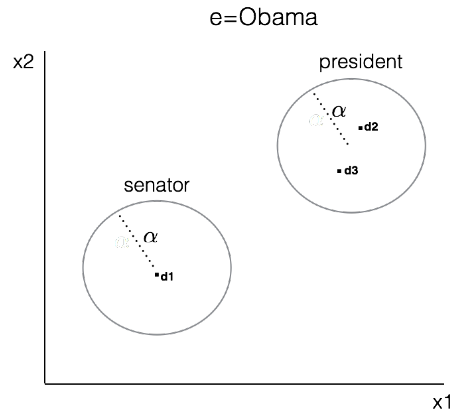


Figure 1: Example of Non-parametric Clustering

3.3 Staleness

We have been concerned with detecting whether a document d contains a novel context in terms of the documents seen so far. By representing the context of an entity as a set of topic clusters, each with an embedding vector, we are able to accurately summarize the entity's context information. We expect that documents that are not close to existing clusters contain novel information. Unfortunately, this representation ignores the timeliness of the information, and it is quite possible that a document that is similar to existing clusters contains novel information. For example, when a document describes Obama victory in an election, it may be assigned to an existing cluster describing a previous election he won, nonetheless it actually contains new information.

A potential solution is to keep track of when the last document was assigned to a cluster, however, KBA challenge requires *all* documents that contain novel information within a time frame to be marked vital as per the timeliness of the document. Such timeliness is a subjective interpretation that can vary per entity and event. As an example let's assume that several documents talk about an event that happened to entity e. During a "short" time frame (here is where the subjective interpretation comes in) that information can be considered new. After a while, that new information transitions to a background state, so as the documents transition from being *vital* to *non-vital*.

In order to address such temporal dynamics that capture novelty and transition documents from a *vital* to a *non-vital* state, we propose a dynamic staleness measure λ_i, $0 < \lambda_i \leq 1$. This staleness measure can be used both for entities and topic clusters. Low staleness of the assigned entity/cluster represents *vital* documents, while high staleness intends to

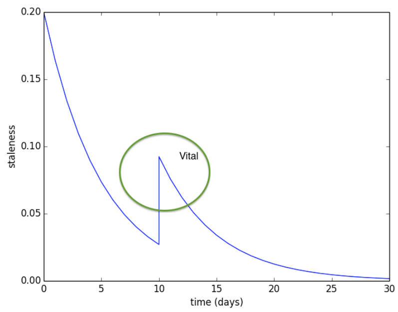


Figure 2: Staleness of Unpopular Entity

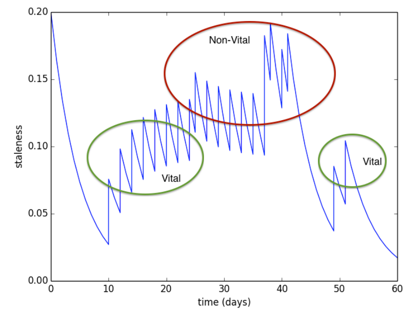


Figure 3: Staleness of Entity with Fluctuating Popularity

represent *non-vital* ones.

The staleness of an entity/cluster at any time t depends on the staleness and the time of the last document d_j assigned to the entity/cluster. The staleness decay rate is exponential, and is controlled by the hyperparameter γ_{dec}:

$$\lambda_t = \lambda_j \exp\left(-\gamma_{dec}\frac{t - t_j}{T}\right) \tag{8}$$

where $\gamma_{dec} \geq 0$, t_j and λ_j are the timestamp and staleness of the last document assigned to the entity/cluster, and T is a constant (used to transform the units of time).

When a new document d_i is assigned to an entity/cluster at time t_i, we can estimate the staleness of the entity/cluster at that time using the above equation, $\lambda_{t_i} = \lambda_{i-1} \exp\left(-\gamma_{dec}\frac{t_i - t_{i-1}}{T}\right)$. This staleness can be used to estimate the novelty of the information in d_i, i.e. a low λ_{t_i} suggests the document contains information that has not been observed for a while.

Thereafter, since we have just observed a relevant document for the entity/cluster, we need to increase its staleness. We use a simple interpolation to increase it, using γ_{inc} to control the change as follows:

$$\lambda_i = 1 - \gamma_{inc}(1 - \lambda_{t_i}) \tag{9}$$

where $0 \leq \gamma_{inc} \leq 1$. The staleness for the entity/cluster is now λ_i, which is used when the next document d_{i+1} is observed.

Figure 2 illustrates an example of an entity with a decreasing staleness. There are almost no documents referring to the entity. As soon as some activity is detected, i.e. a document mentioning the entity appears ($t = 10$), the staleness increases slightly. Given the fact that there's not much information about the entity, every new document would drive an update to the entity's profile, strongly suggesting vitalness.

Figure 3 aims to represent staleness of an entity with fluctuating activity levels in the stream of documents. At time t=10, a main event involving the entity starts and continues for a long period of time, showing a growing trend in popularity. At the beginning, those documents can be considered vital, but as time goes by and documents continue commenting on the same event, the information starts staling, clearly indicating non-vitalness. Near t=40, the event can be considered over, a steep decrease in popularity is shown. At a later time, $t = 50$, a new event occurs, which denotes vitalness.

3.4 Algorithm

4. RELATED WORK

Several knowledge based acceleration competitions have been done in the recent past, testifying the great progress achieved in these fields [11]. Liu and Fang [15] present one of the best performing systems in TREC KBA 2012. They created broader representations of entities' profiles based on a Wikipedia snapshot and considered the anchor text of all internal Wikipedia links as related entities. In TREC KBA 2013 competition, different families of methods were proposed, including query expansion, classification, and learning to rank.

Our strategy is somewhat similar to Wang et al. [25] in the sense that we first target a high recall system and then apply different classification methods to differentiate between *vital* and *non-vital* documents. One key difference is that we do not exploit any external resources to construct features, e.g. we do not use Wikipedia entity pages nor existing citations in the Wikipedia page of an entity.

Representing words as continuous vectors has been around for a while [12, 7]. The progress of machine learning techniques in recent years enabled training more complex models on much larger data sets [17]. One popular approach to increase accuracy in existing system is to use unsupervised methods to create word features, or to download word features that have already been produced [24]. In our method, we do the latter, we use already induced word embedding features in order to improve its accuracy.

To our best knowledge, no techniques propose a combination of distributed word embeddings, non-parametric clustering, and a notion of staleness to solve this problem. The closest example we found that addresses the problem of staleness detection is Gamon [9], where the author builds an association graph connecting sentences and sentence fragments, and uses graph-based features as good indicators of lack of novelty.

Streaming document filtering is also related to several other fields, including but not limited to, entity linking [13], text categorization [14], news surveillance [23], and cross-document coreference [20, 22].

5. KBA VITAL FILTERING EVALUATION

5.1 Data

To assess our method we use TREC KBA 2014 filtered stream corpus. It has around 20M documents annotated with

Basic Features, F_b	
Based on document d	
$\log(len(d))$	log of the length of d
$source(d)$	discretized source of d
Based on document d and target entity e	
$n(d, e)$	# of occurrences of target entity e in d
$n(d, e^p)$	# of occurrences of partial name of e in d
$\text{fpos}(d, e)$	position of first occurrence of entity e in d
$\text{fpos}_n(d, e)$	$\text{fpos}(d, e)$ normalized by document length
$\text{fpos}(d, e^p)$	position of first partial occurrence of e in d
$\text{fpos}_n(d, e^p)$	$\text{fpos}(d, e_p)$ normalized by document length
$\text{lpos}(d, e)$	position of last occurrence of entity e in d
$\text{lpos}_n(d, e)$	$\text{lpos}(d, e)$ normalized by document length
$\text{lpos}(d, e^p)$	position of last partial occurrence of entity e in d
$\text{lpos}_n(d, e^p)$	$\text{lpos}(d, e^p)$ normalized by document length
$\text{spread}(d, e)$	$\text{lpos}(d, e) - \text{fpos}(d, e)$
$\text{spread}_n(d, e)$	$\text{spread}(d, e)$ normalized by document length
$\text{spread}(d, e^p)$	$\text{lpos}(d, e^p) - \text{fpos}(d, e^p)$
$\text{spread}_n(d, e^p)$	$\text{spread}(d, e^p)$ normalized by document length
Embedding Features, F_e	
v_d	mean word embedding representation of d
$\text{zero}(v_d)$	$\mathbb{1}_{v_d=0}$, set to 1 if v_d is 0
Clustering Features, F_c	
$\min_c(v_d, v_c)$	minimum distance of v_d to topic clusters of e
$\text{avg}_c(v_d, v_c)$	average distance of v_d to topic clusters of e
Temporal Features, F_t	
$\lambda(e)$	current staleness of entity e
$\lambda(e, c)$	current staleness of topic c of target entity e

Table 1: Features for Vital Filtering classification

BBN's Serif NLP tools, including within-doc coreference and dependency parse trees. Further, we use the 71 target entities given by KBA organizers for the Vital Filtering task. Among the 20M documents, around 28K have truth labels. From these, only 8K are training instances while the rest are test examples. We preprocess the corpus to filter the documents that contain exact string matches to the target entities names, including canonical and surface form names.

5.2 Features

Our approach extends the classifier introduced by Wang et al. [25]. We construct a basic set of features based on the document and the entity of interest. Using our representation, we include additional features for the embedding, clustering, and staleness. A summary of the features we use is presented in Table 1.

5.3 Relevance Classification

TREC KBA 2014 corpus contain documents that do not refer to the target entities, even though they may contain mentions to them. We therefore need to use a *non-referent* category of documents. A *non-referent* document denotes that it does not refer to a target entity or the context is so ambiguous that it is impossible to decide whether the mention refers to an entity or not. An example of the former case is "Barack Ferrazzano provides a wide range of business-oriented legal". For the latter, an example is "Barack is a great father and a better husband". The mention "Barack" may refer to any married parent named Barack, therefore, we consider it *non-referent*. The *vital* and *non-vital* classes described in section 2 fall into a *referent* (or *relevant*) category, which contains documents that refer to the target entities.

Due to the fact that not all documents in the corpus refer to the target entities, we include an extra step in our classification process. We introduce an additional classifier, called *rnr*, which is trained offline and classifies documents as *referent* or *non-referent*. Consequently, in every experiment, each document goes first through the *rnr* classifier. Only the *relevant* documents outputted by *rnr* are used as inputs to the *uv* model, which discriminates between *vital* and *non-vital* documents, the overall focus of this work.

5.4 Methods

We use randomized tree ensembles classifiers [10] for both *rnr* and *uv*, each composed of 100 weak learners. Each tree in the ensembles has a maximum depth of 150. All the experiments use the same *rnr* model trained with the basic features listed in section 5.2. The different methods differ on the features used to train and test the *uv* classifier.

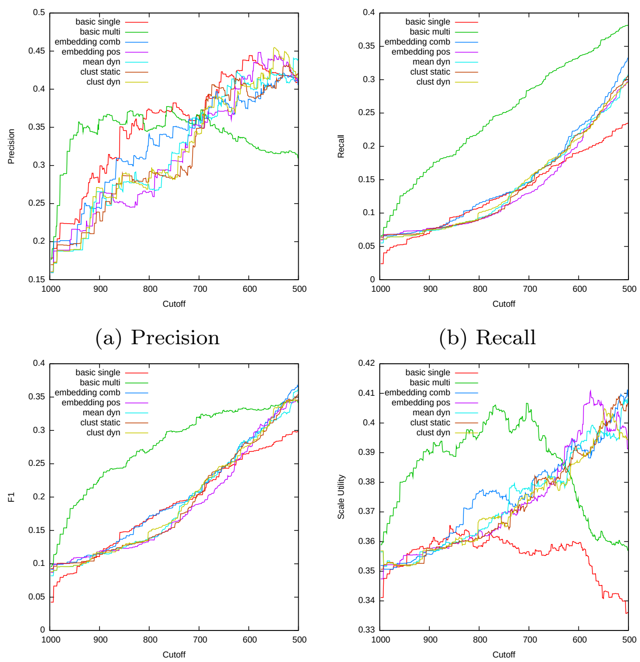
- *Baseline Single*: baseline method that uses F_b features. Wang et al. [25], Bellogín et al. [2] have a similar method, though they train their models with more features.
- *Baseline Multi-task*: method similar to *Basic Single* but includes multi-task learning [5].
- *Embedding Combined*: similar to *Basic Multi-task* but includes F_e features. The word embeddings are computed using the Google News dataset. Each document has a single combined embedding, which is calculated from the nouns, proper nouns and verbs found around the mentions, as described in section 3.1.
- *Embedding POS*: similar to *Embedding Combined* but instead of a single combined embedding per document, it has one embedding per word type in each document, i.e. it computes separate embeddings for nouns, proper nouns and verbs.
- *Mean Dynamic*: similar to the *Embedding POS* method but it includes the F_t features.
- *Clustering Static*: similar to the *Embedding POS* method but it includes the F_c features.
- *Clustering Dynamic*: similar to the *Embedding POS* method but it includes F_t and F_c features.

6. RESULTS AND DISCUSSION

Table 2 shows the precision, recall and F1 results of the methods explained in 5.4, computed using KBA official scorer tool with cutoff=50. The *Clustering* models use $\alpha = 0.8$. Also, the *Dynamic* methods use $\gamma_{dec} = 1$ and $\gamma_{inc} = 0.1$. *Baseline Single* performs as expected, i.e. has lower F1 than the other models. On the other hand, *Baseline Multi-task* perform far better that *Baseline Single*, which evidences that multi-task learning does work. Most of the more advanced models perform better than *Baseline Multi-task*. Using a combined embedding (*Embedding Combined*) outperforms using individual embeddings representations for the different type of words (*Embedding POS*). The staleness and non-parametric clustering runs (*Mean Dynamic*, *Clustering Static*, *Clustering Dynamic*) perform slightly worse than the simple *Embedding Combined* method. Nevertheless, they illustrate

Model	Features	Micro-averaged			Macro-averaged		
		P	**R**	**F1**	**P**	**R**	**F1**
Baseline Single	F_b	0.468	0.246	0.323	0.409	0.235	0.298
Baseline Multi-task	F_b	0.465	0.403	0.432	0.319	**0.377**	0.346
Embedding Combined	$F_b + F_e$	0.467	**0.465**	**0.466**	0.413	0.331	**0.367**
Embedding POS	$F_b + F_e$	0.480	0.419	0.447	0.412	0.299	0.346
Mean Dynamic	$F_b + F_e + F_t$	**0.483**	0.436	0.459	**0.440**	0.308	0.362
Clustering Static	$F_b + F_e + F_c$	0.483	0.440	0.460	0.420	0.306	0.354
Clustering Dynamic	$F_b + F_e + F_c + F_t$	0.481	0.435	0.457	0.423	0.293	0.346

Table 2: UW Vital Filtering results for TREC KBA 2014



(a) Precision (b) Recall

(c) F1 (d) Scale Utility

Figure 4: P-R-F1-SU over confidence cutoffs

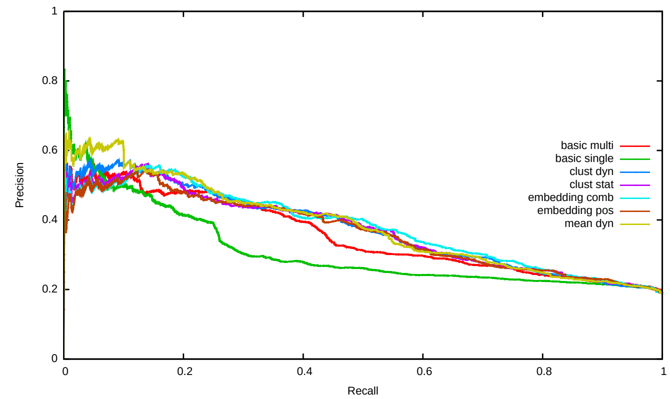


Figure 5: Precision-Recall

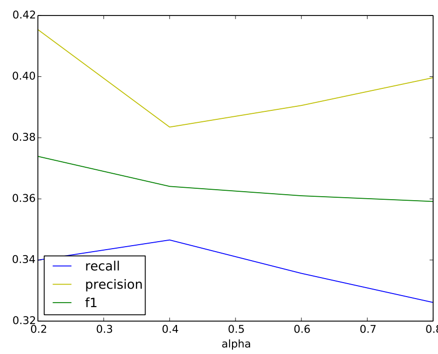


Figure 6: P-R-F1 vs. α

the importance of these new features as they use separate embeddings for the different word types, and they all improve the performance of the simple *Embedding POS* model.

Figure 4 complements the results in Table 2 for different confidence cutoffs. Figure 5 further illustrates the precision-recall for the different methods. On low recall, the precision of the different models meets our expectations, the more complex methods, which include non-parametric clustering and staleness, in general outperform the simpler ones. Nevertheless, on high recall, *Embedding Combined* takes the lead.

Figure 6 shows the results for a variant of *Clustering Dynamic*. As opposed to the *Clustering Dynamic* in Table 2, this new version uses a single embedding representation instead of embeddings per word type. We show the change in macro P-R-F1 for different values of α, using $\gamma_{inc} = 0.1$ and $\gamma_{dec} = 1$. We see that the performance degrades as α increases, which further reflects the importance of hyper-parameter tuning. Even more, this variant with $\alpha = 0.2$ performed the best, achieving a macro F1 of 0.374.

Further, we present some qualitative results on the topic clusters of this variant of *Clustering Dynamic* using α=0.8. Table 3 illustrates the 5 closest words to the word that is most similar (in cosine similarity) to the cluster centroids of the entity *Shawn Atleo* on 11/15/2012. By doing some manual search on *Shawn Atleo*, we can easily confirm that C2 elements make sense. *Mr. Atleo* is an activist for the rights of First Nations in Canada, and received several Honorary Doctorate of Laws degrees from different universities, which explains a cluster with the words in Table 3. C1 is somewhat harder, it is a cluster close to dates and finances. It may be explained by all the articles that talk about *Shawn*

C1	C2
Monday	conduct
Tuesday	complication
month	judicial
billion	government
million	guidance

Table 3: Topic clusters closest words for *Shawn Atleo*

announcing huge investments in different areas.

We believe further experimental investigations are needed to account for the correct tuning of the hyperparameters of the model. Exploiting external resources such as Wikipedia entity pages to construct more features [15] should probably increase the overall accuracy of our method.

7. CONCLUSION AND FUTURE WORK

Filtering streaming documents to accelerate users filling knowledge gaps plays a crucial role in the maintenance and update of knowledge bases. With the exponential increase of information on the web, it becomes critical to detect relevant documents and incorporate their information to entities in a timely manner.

In this paper we introduced a semi-supervised learning model for document filtering tasks. We proposed a distributed, non-parametric representation of documents suitable for streaming settings, that groups entities' references into topic clusters. Further, we present a notion of staleness computed per entity as well as per topic cluster, which dynamically estimates entities' and clusters' relevances. Combining these three core ideas, distributed word embeddings, non-parametric clustering, and staleness, results in a more accurate representation of entities' contexts, and simultaneously addresses the filtering requirements of large corpora of streaming text documents.

Further work needs to be done. A possible line of future research would be exploring hierarchical clustering algorithms to better represent topic clusters. It would also be interesting to assess the effect of learning the hyperparameters of the model instead of just manual tuning them for the specific datasets. It would be worthwhile to assess the effects of using different pre-trained word embeddings.

References

[1] J. Alsakran, Y. Chen, D. Luo, Y. Zhao, Y. Jing, W. Dou, and S. Liu. Real-Time Visualization of Streaming Text with a Force-Based Dynamic System. *IEEE Computer Graphics and Applications*, pages 34–45, 2012.

[2] A. Bellogín, G. G. Gebremeskel, J. He, J. Lin, A. Said, T. Samar, A. P. de Vries, and J. B. P. Vuurens. CWI and TU Delft at TREC 2013: Contextual Suggestion, Federated Web Search, KBA, and Web Tracks. In *Proceedings of the Twenty-Second Text REtrieval Conference (TREC 2013)*, 2013.

[3] D. Blei. Probabilistic topic models. *Communications of the ACM*, page 77âĂŞ84, 2012.

[4] V. Bouvier and P. Bellot. Filtering Entity Centric Documents using Numerics and Temporals features within RF Classifier. In *Proceedings of the Twenty-Second Text REtrieval Conference (TREC 2013)*, 2013.

[5] R. Caruana. Multitask Learning: A Knowledge-Based Source of Inductive Bias. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 41–48, 1993.

[6] M. Efron, C. Willis, P. Organisciak, B. Balsamo, and A. Lucic. The University of IllinoisâĂŹ Graduate School of Library and Information Science at TREC 2013. In *Proceedings of the Twenty-Second Text REtrieval Conference (TREC 2013)*, 2013.

[7] J. L. Elman. Finding structure in time. *COGNITIVE SCIENCE*, pages 179–211, 1990.

[8] J. R. Frank, M. Kleiman-Weiner, D. A. Roberts, F. Niu, C. Zhang, and C. Ré. Building an Entity-Centric Stream Filtering Test Collection for TREC 2012. In *Proceedings of the Twenty-First Text REtrieval Conference (TREC 2012)*, 2012.

[9] M. Gamon. Graph-Based text Representation for Novelty Detection. In *Proceedings of TextGraphs: the First Workshop on Graph Based Methods for Natural Language Processing*, 2006.

[10] P. Geurts, D. Ernst, and L. Wehenkel. Extremely Randomized Trees. *Machine Learning*, pages 3–42, 2006.

[11] O. Gross, A. Doucet, and H. Toivonen. Term Association Analysis for Named Entity Filtering. In *Proceedings of the Twenty-First Text REtrieval Conference (TREC 2012)*, 2012.

[12] G. E. Hinton, J. L. McClelland, and D. E. Rumelhart. Distributed Representations. In *Parallel Distributed Processing: Volume 1: Foundations*, pages 77–109. 1987.

[13] H. Ji and R. Grishman. Knowledge Base Population: Successful Approaches and Challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pages 1148–1158, 2011.

[14] B. Kjersten and P. McNamee. THE HLTCOE APPROACH TO THE TREC 2012 KBA TRACK. In *Proceedings of the Twenty-First Text REtrieval Conference (TREC 2012)*, 2012.

[15] X. Liu and H. Fang. Entity Profile based Approach in Automatic Knowledge Finding. In *Proceedings of the Twenty-First Text REtrieval Conference (TREC 2012)*, 2012.

[16] X. Liu, J. Darko, and H. Fang. A Related Entity based Approach for Knowledge Base Acceleration. In *Proceedings of the Twenty-Second Text REtrieval Conference (TREC 2013)*, 2013.

[17] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *CoRR*, 2013.

[18] T. Mikolov, W. tau Yih, and G. Zweig. Linguistic Regularities in Continuous Space Word Representations. In *HLT-NAACL*, pages 746–751. The Association for Computational Linguistics, 2013.

[19] A. Neelakantan, J. Shankar, A. Passos, and A. McCallum. Efficient Non-parametric Estimation of Multiple Embeddings per Word in Vector Space. *EMNLP*, 2014.

[20] D. Rao, P. McNamee, and M. Dredze. Streaming Cross Document Entity Coreference Resolution. In *COLING (Posters)*, pages 1050–1058, 2010.

[21] A. Silic and B. D. Basic. Visualization of Text Streams: A Survey. *Knowledge-Based and Intelligent Information and Engineering Systems*, pages 31–43, 2010.

[22] S. Singh, A. Subramanya, F. Pereira, and A. McCallum. Large-scale cross-document coreference using distributed inference and hierarchical models. In *Association for Computational Linguistics (ACL)*, 2011.

[23] R. Steinberger. A survey of methods to ease the development of highly multilingual text mining applications. *CoRR*, 2014.

[24] J. Turian, L. Ratinov, and Y. Bengio. Word representations: A simple and general method for semisupervised learning. In *ACL*, pages 384–394, 2010.

[25] J. Wang, D. Song, L. Liao, and C.-Y. Lin. BIT and MSRA at TREC KBA CCR track 2013. In *Proceedings of the Twenty-Second Text REtrieval Conference (TREC 2013)*, 2013.

[26] C. Zhang, W. Xu, R. Liu, W. Zhang, D. Zhang, J. Ji, and J. Yang. PRIS at TREC2013 Knowledge Base Acceleration Track. In *Proceedings of the Twenty-Second Text REtrieval Conference (TREC 2013)*, 2013.