Word Embedding based Clustering to Detect Topics in Social Media

Carmela Comito
Nat. Research Council of Italy (CNR)
Institute for High Performance Computing and Networking (ICAR)
Rende, Italy
carmela.comito@icar.cnr.it

Agostino Forestiero
Nat. Research Council of Italy (CNR)
Institute for High Performance Computing and Networking (ICAR)
Rende, Italy
agostino.forestiero@icar.cnr.it

Clara Pizzuti
Nat. Research Council of Italy (CNR)
Institute for High Performance Computing and Networking (ICAR)
Rende, Italy
clara.pizzuti@icar.cnr.it

ABSTRACT

Social media are playing an increasingly important role in reporting major events happening in the world. However, detecting events and topics of interest from social media is a challenging task due to the huge magnitude of the data and the complex semantics of the language being processed. The paper proposes an online algorithm to discover topics that incrementally groups short text by incorporating the textual content with latent feature vector representations of words appearing in the text, trained on very large corpora to improve the check-in topic mapping learnt on a smaller corpus. Experimental results show that by using information from the external corpora, the approach obtains significant improvements with respect to classical topic detection methods.

CCS CONCEPTS

• **Information systems → Clustering**; **Data stream mining**; **Data extraction and integration**; • **Computing methodologies → Neural networks**.

KEYWORDS

Social Media, Topic Detection, Word Embedding, Clustering

ACM Reference Format:
Carmela Comito, Agostino Forestiero, and Clara Pizzuti. 2019. Word Embedding based Clustering to Detect Topics in Social Media . In *IEEE/WIC/ACM International Conference on Web Intelligence (WI '19), October 14–17, 2019, Thessaloniki, Greece.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3350546.3352518

1 INTRODUCTION

Social media have become the main means through which information is launched and propagated on a global scale. This led to the collection of huge volumes of data about people interests, topics of discussion, real-word events, emergency situations, important events like elections, disasters, concerts, and football games. As a result, social media provide a great medium for situation awareness through the messages written by millions of real world users. The detection of topics (also called events in the paper) can be obtained by grouping together messages which show similar content. Extracting knowledge from short texts, like messages and posts, however, poses new problems to the classical *bag-of-words* representation of documents [22] because of data sparsity, mainly due to very poor contextual information available contained in them. In fact, in this representation a document is a vector of words weighted using the *term frequency (tf)* and the *inverse document frequency (idf)*, appropriately normalized, which is based on simple term occurrence information.

The expansion of the representation for short documents is a key problem in social media based applications. Developments in this research line have exploited mainly syntactical variations of the same word (stems) and the lexical relatedness of terms. Traditional clustering based topic detection methods rely on the *bag-of-words* and they lack context information since they ignore the order and semantic relations between words. A strategy to overcome this problem is to use *n-grams*, i.e. a sequence of n co-occurring words (or, simply, a context window of n words) instead of single words. The same *tf-idf* score is used, but applied to the *n-grams*. The use of *n-grams* has been shown to produce significant improvements in learning, although it is still limited in capturing contextual information observed in non-sequential patterns.

On the other hand, conventional topic modeling approaches such as *pLSA* and *LDA* [20, 23], have been designed to implicitly capture word co-occurrence patterns at the document level to reveal topic structures, by expanding short texts using latent semantics, learned by latent Dirichlet allocation (LDA) and its extensions. However, these methods still consider a text as bag-of-words and, because of the length of each document, suffer a lot from the data sparsity problem in short texts, leading to inferior topic inferences. Therefore, they are not effective in capturing fine grained semantics for short texts modeling.

Earlier research efforts done about semantic similarity relied on some type of distance among the words, for instance hops in a manually built semantic dictionary. The problem with most of these dictionary-based approaches is that they are manually built, usually for a particular application, not scaling in real word fast growing and dynamic settings, like huge streams of social media data, and also unable to adapt to new contexts.

More recently, deep learning based methods have drawn much attentions with word embeddings. Word embedding represents words with dense, low-dimensional and real-valued vectors. Each dimension of the vectors encodes a different aspect of words.

WI '19, October 14–17, 2019, Thessaloniki, Greece

ACM ISBN 978-1-4503-6934-3/19/10…$15.00
https://doi.org/10.1145/3350546.3352518

Accordingly, in [16, 17] Mikolov et al. implement the semantic similarity task by computing the distances between words exploiting their position within an embedding space. In [8] word embedding has been integrated within conventional topic modeling approaches. A common drawback of most of these embedding-based approaches, mainly in the context of topic modeling, is that they only consider general implicit relationships between words in the context of the document, not taking advantage of the local information from semantic relationships between a word and its neighbors.

In this context, the paper proposes a novel approach to topic detection in social media by exploring potential synergies between word embedding and clustering methods. The method, named *Word embedding Clustering* (*WEC*), is an online semantic-based clustering method which groups streams of social media posts discussing the same topic by exploiting both semantic and lexical characteristics of the post contents. Word embedding, in fact, allows to exploit the ordering of the words and semantics information from the text corpus. The approach uses a pre-trained (over large-scale external corpus) word embedding model to obtain the semantic representations of short texts and overcome the weakness of conventional methods. Specifically, our strategy, exploits the nearest words of a given pre-trained word embedding to generate semantic enriched word representation, in terms of syntactic and semantic information. The explicit exploitation of similarity between word embeddings provides fine-grained information about relationships between words.

The key feature of the proposed clustering method is the similarity measure formulated as a combined weighted sum of semantic and syntactic metrics. The similarity takes also into account the importance of terms according to their frequency and uses a fading function to effectively consider more important the most recent posts. The experimentation on a dataset of tweets addressing health issues, and a comparison with state-of-the-art approaches show that the method is effectively capable to detect topics of interest.

The paper is organized as follows. In the next section an overview of the most recent proposals for discovering topics in social data is given. Section 3 introduces the data representation of our approach and defines the similarity measure between a post and a cluster. Section 4 presents the online algorithm. Section 5 describes the data set of tweets used to evaluate the method. Section 6 reports the experimental results obtained by executing *WEC* and compares them with state-of-the-art approaches. Finally Section 7 concludes the paper.

2 RELATED WORK

The detection of topics and events in social media is an important research activity which is receiving an increasing interest because of the applicability in different application domains. In the following we review the most popular methods. For detailed surveys see [5, 11, 14, 15, 24].

LDA [7] is a topic model that relates words and documents through latent topics. It associates with each document a probability distribution over topics, which are distributions over words. A document is represented with a set of terms, which constitute the observed variables of the model. The topic and term distributions are estimated by using Bayesian Inference. One of the main drawback of *LDA* is that it requires the expected number of topics as input parameter. *Doc-p* [20] is an on-line clustering method that represents each document as a list of words, and computes the cosine similarity of the *tf-idf* [22] representation of a new text with respect to all those already examined. If the best similarity is above a threshold θ, then the document is assigned to the cluster with the best match, otherwise a new cluster is generated. Becker et al. [6] proposed an online clustering approach coupled with a post-processing classification step that distinguishes between events and non-events. To identify real- world events, each message is represented as a *tf-idf* weight vector, where terms are all those appearing in all the documents. Since the number of terms may be very high, the authors consider the n most frequent terms, where n is experimentally fixed, to compute features that help in revealing the events. Aiello et al. [3] introduced two methods to detect trending topics: *Soft Frequent Pattern Mining (SFPM)*, a soft version of the well known frequent pattern mining approach, and *BNGram*, which finds emerging topics by considering the co-occurrences of n-grams instead of unigrams, and comparing the frequencies of terms in the current time slot and the preceding ones. *TweetHealth* [10] is an online clustering algorithm for detecting health-related topics by leveraging on the frequency of lexically similar terms. The similarity is implemented only in terms of syntactic regularities among tweets.

As regards the use of word embedding for clustering problems, very few methods have been proposed. Quimin et al. [21] proposed a feature cluster-based vector space model to improve the accuracy of document clustering. The authors introduce the concept of non-contiguous phrases, i.e. sequences of words which have a clear meaning but are not close in the text, and include them in the feature vector, along with words and phrases. A distributed representation of terms appearing in the features is obtained by using the Skip-gram model [17], then similar terms are clustered by using the K-means method to obtain feature clusters. A document is then represented as the co-occurrence matrix of feature classes, where the weight of each feature class is computed by using the *tf-idf* score, obtained by considering the number of occurrences of a feature class in the documents. After that, the documents are clustered by using again the K-means method. This double clustering approach obtained higher F-measure values with respect to other classical approaches, such as *LDA*. Dai et al. [12] presented a word embedding based clustering to classify tweets as related or not to a topic. To this end, each tweet is represented as a semantic vector, obtained by using the Word2Vec model [16, 17], and then these vectors are clustered by computing the cosine similarity between them. Thus the authors do not group together similar tweets, rather they cluster the terms appearing in each tweet and then compare these clusters with a topic to decide if that tweet is related to the selected topic. Fraj et al. [13] employed the Word2Vec model to obtain word vector representation of tweets and then used the K-means clustering method to group topically similar tweets. The number of clusters is an input parameter which must be fixed in advance. A comparison with *LDA* on a dataset of tweets showed the better performance of their approach.

It is worth noting that all the described methods use only word embedding to perform clustering of text. Our approach, instead, combines the textual contents of short text with the distributed representations in a vector space of the words appearing in the text.

3 PRELIMINARIES

In this section we introduce the concepts used to model the word embedding based clustering approach, mainly the social media post representation, the cluster structures and the similarity measure introduced to group similar posts within the same cluster.

A social media post smp is defined as a data structure storing relevant features of the post textual content:

Definition 1. *A social media post smp is defined as a tuple* $smp = (id, t, fv, sfv)$ *where id is the post identifier,* t *is the time at which the post has been published,* $fv = (w_u, w_b, h_u, h_b, m_u, m_b)$ *is a vector of textual features extracted from the post, representing words, unigram* w_u *and bigram* w_b*, hashtags, unigram* h_u *and bigram* h_b*, mentions, unigram* m_u *and bigram* m_b*,* $sfv = (ew_u, ew_b, eh_u, eh_b, em_u, em_b)$ *is the semantic feature vector corresponding to* fv*.*

The semantic feature vector sfv is a vector where each element in turn is an n-dimensional vector in a semantic space obtained through word embedding. Each feature item is represented as a real-valued vector, and each dimension contains a certain amount of semantic information. To store summary information of all the posts assigned to a cluster C, we introduce the *cluster centroid* as a compact data structure. The centroid keeps the textual items as in the feature vector fv of each social media post smp assigned to the cluster, its corresponding semantic vector sfv, together with their frequencies and their temporal evolution.

Definition 2. *The centroid of a cluster* C *is a tuple* $CC = (c, t_0, t_c, fv^c, ff, sfv^c)$*, where* c *is the cluster label,* t_0 *is the creation time of the cluster,* t_c *is the time stamp of the last time a social object was added to* C*,* fv^c *and* sfv^c *are the textual and semantic feature vectors, respectively, analogous to the ones defined for the social medial post smp, and* $ff = (f_{w_u}, f_{w_b}, f_{h_u}, f_{h_b}, f_{m_u}, f_{m_b})$ *is the list of frequencies corresponding to* fv^c*.*

A post is assigned to a cluster if it is similar to the cluster centroid. The similarity is based both on the lexicon used in the posts and on their semantics obtained by exploiting word embedding. Therefore, the similarity will be the combination of two measures accounting both syntactic as well as semantics of the posts. A further consideration in designing the similarity metrics is that textual features appearing more frequently should have a higher weight when computing the similarity. Moreover, we assume that posts discussing the same topic are usually temporally close, thus also the temporal proximity should be taken into consideration.

Let smp $= (id, t, fv, sfv)$ be a social media post and CC=$(c, t_0, t_c, fv^c, ff, sfv^c)$ a centroid. For each word, hashtag, mention, both unigram and bigram of smp, the centroid with the most similar textual and semantic feature vector has to be identified. Specifically, for each feature item in smp we find the cluster whose CC has the maximum similarity in both semantic and syntactic spaces.

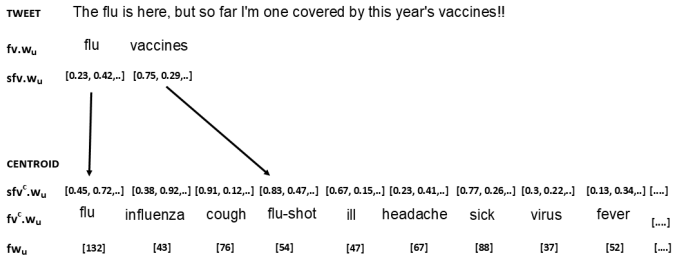

Figure 1: Example of tweet and cluster centroid representation.

$$sim(\text{smp}, CC) = (\alpha sim_{sem}(\text{smp}, CC) + \beta sim_{lex}(\text{smp}, CC)) \times f(t_c) \quad (1)$$

where α and β are the weights of the semantic and textual similarity, $f(t_c)$ is a fading function which biases the similarity function towards clusters temporally closer to the post. The fading function we used is that proposed in [9] defined as $f(t_c) = 2^{-\lambda(t_{smp} - t_c)}$ where t_{smp} is the publication time of the post that could be added to C, t_c is the time of the last post assigned to C and λ is the *decay rate* of a cluster. $\lambda = 1/lf$, where $lf = \delta + 2^h \times (t_c - t_0)$, where t_0 is the creation time of the cluster. δ and h are user defined parameters. δ is the minimum time period allowed to a cluster to survive after its creation, even if no new message is added to it within a δ time, while h determines the temporal horizon a cluster is considered active. λ establishes the importance of the historical data in the social streams. A cluster is considered *active* if it continues to receive new messages.

The semantic similarity metrics is formulated by exploiting the cosine similarity between the semantic feature vectors of the incoming social post and that of the cluster centroid. The similarity is computed by taking for each item in the semantic feature vector of the post the maximum cosine similarity value with the semantic feature of the centroid. In other words, for each item in the sfv of smp the semantically most similar term in the sfv^c of the centroid is determined. Such a value is then multiplied by the frequency of the corresponding term in the centroid.

Let $sfv_i, sfv_i^c, i = 1, \ldots, 6$ denote a generic component of the semantic feature vectors of a post smp and the centroid CC of a cluster C, respectively, and $n = \mid sfv_i \mid$, $m = \mid sfv_i^c \mid$ be the corresponding size. Then, the semantic similarity between smp and CC is computed as:

$$sim_{sem}(\text{smp}, CC) = \frac{\sum_{i=1}^{6} \underset{\substack{x \in \{1,\ldots,n\} \\ y \in \{1,\ldots,m\}}}{\arg\max} cos(sfv_i(x), sfv_i^c(y)) \times ff_i^{max}}{\sum_{i=1}^{6} ff_i^{max}} \quad (2)$$

where the *arg max* function gives the highest cosine similarity value computed among the $n \times m$ couples (x, y) of features of sfv_i and sfv_i^c, and ff_i^{max} is the frequency of the textual feature of fv_i^c corresponding to the determined most similar semantic feature of sfv_i^c.

For what concerns the syntactic similarity we followed the same approach but, instead of using the cosine similarity, we exploit the Levenshtein distance to measure the similarity among the feature

vector of the social media smp and that one of the centroid. Note that in such a case, since we have words and not vectors of numbers, it is not necessary to compare all the $n \times m$ couples of fv and fv^c, but only those contained in their intersection. Figure 1 shows an example of a tweet and the cluster centroid it has been assigned, along with the list of word fv_{w_u}, $fv^c_{w_u}$ and semantic word sfv_{w_u} $sfv^c_{w_u}$ unigrams, of the tweet and the cluster, respectively. For the cluster centroid also the frequencies f_{w_u} are reported. Note that the word *vaccines* of the tweet and the word *flue-shot* of the centroid are the most semantically similar words.

4 WORD EMBEDDING BASED CLUSTERING

In this section we describe the online semantic-based clustering method which groups social media posts discussing the same topic by employing the similarity measure defined in the previous section, that includes both semantic and lexical characteristics of the post contents, along with the temporal information on when the message has been issued. The method follows the classical online clustering approach for event clustering introduced by Yang et al. [25] and adopted by other authors [4, 6, 9, 26] as basic scheme for clustering streams of messages. However, differently from these approaches, the proposed method uses a richer representation of a social media post by including both textual and semantic information, which is then exploited to assign a post to the most similar cluster.

The pseudo-code of the algorithm is reported in Figure 2. The algorithm receives the social media posts as input, the similarity threshold ϵ, the minimum survival time period δ and the active time period h of a cluster. The first time the algorithm receives a social media post smp_1 from the stream, it generates the first cluster C_1 by storing in the centroid the feature vector fv_1 of smp_1 and setting the corresponding frequencies to 1. Moreover C_1 receives the time stamp of smp_1.

While a new post smp_i arrives at time t_i, the algorithm builds the representation fv_i of smp_i and its word embedding sfv_i, and computes both the semantic similarity and the syntactic one between the post and the clusters active at the time stamp t_i, while the inactive clusters are removed from the set of clusters . Let C_c be the cluster whose centroid has maximum similarity with smp_i. If this similarity value $sim(\text{smp}_i, C_c)$ is lower than ϵ, a new cluster is generated from fv_i and sfv_i and added to the set of clusters, otherwise the social media post in the form of fv_i and sfv_i is added to C_c by updating the centroid. These steps are repeated until the algorithm receives new posts.

To update a centroid $CC = (c, t_0, t_c, sgn)$ when a new post $\text{smp} = (id, u, t, l, fv, sfv)$ is added to any cluster C, the time stamp of C is updated with the time stamp t of smp. Then all the items of each feature must be checked if already present in the centroid. Thus, the intersection between the feature vectors of smp and that of CC are computed. Then, for each feature $\text{smp}.fv(i)$ of the post, if an element of this feature already appears in the feature vector of the centroid $fv^c(i)$, the corresponding frequency must be incremented by 1, otherwise, it will be added to the centroid feature and its frequency is set to 1.

Algorithm *WEC*
Input: A continuous stream of social media posts $\text{smp}_1, \ldots, \text{smp}_n, \ldots$ a similarity threshold ϵ the minimum survival time period δ, the active time period h
Output: The set of currently active clusters C

begin
1. $C \leftarrow \emptyset$;
2. i=1;
3. $\text{fv}_1 \leftarrow$ GetFeatureVector(smp_1);
3. $\text{sfv}_1 \leftarrow$ GetSemanticFeatureVector(fv_1);
7. $C_1 \leftarrow$ CreateCluster(fv_1, sfv_1);
8. $C \leftarrow C \cup C_1$;
9. **while (not end of stream)**
10. i ← i + 1;
11. Receive the next post smp_i;
12. $\text{fv}_i \leftarrow$ GetFeatureVector(smp_i);
13. $\text{sfv}_i \leftarrow$ GetSemanticFeatureVector(fv_i);
14. **for each** cluster $C_j \in C = \{C_1, \ldots, C_k\}$
15. **if** isActive(C_j) **then**
16. $\text{sim}_{\text{sem}}(\text{sfv}_i, C_j) \leftarrow$ ComputeSemanticSimilarity(sfv_i,C_j);
17. $\text{sim}_{\text{lex}}(\text{fv}_i, C_j) \leftarrow$ ComputeSimilarity(fv_i,C_j);
18. $\text{sim}(\text{smp}_i, C_j) \leftarrow (\alpha \text{sim}_{\text{sem}}(\text{sfv}_i, C_j) + \beta \text{sim}_{\text{lex}}(\text{fv}_i, C_j)) * f(t_c)$;
17. **else**
18. $C \leftarrow C - C_j$;
19. **end**
20. **end**
21. $c = \underset{j \in \{1,\ldots,k\}}{\arg\max}\ \text{sim}(\text{smp}_i, C_j)$
22. **if** $\text{Sim}(\text{smp}_i, C_c) >= \epsilon$ **then**
29. updateCentroid(C_c, fv_i, sfv_i);
30. **else**
31. $C_i \leftarrow$ CreateCluster(fv_i, sfv_i);
32. $C \leftarrow C \cup C_i$;
33. **end if**
34. **end while**
end

Figure 2: The *WEC* algorithm.

5 DATASETS

5.1 Tweets Data

The dataset used for the experimental evaluation comes from the data collected in the context of a study about influenza surveillance in US in [19] and made available from the web page [2]. The data consists of about 50,000 tweets posted in the period September 2015 - April 2016 in US. This dataset of tweets has been obtained by first filtering tweets having health-related keywords (including flu-related words) using the Twitter streaming API, and then using Amazon Mechanical Turk, a crowdsourcing service, [1] to distinguish relevant health tweets from spurious ones. Specifically, the tweets are selected if their content matches any of 269 health keywords. The selection of these 269 keywords was made by identifying words strongly associated with the collection of health-related tweets used in a previous study of the same authors presented in [18] and manually removing non-informative terms.

Table 1: A sample of ground-truth topics of the tweet dataset.

Topics	Keywords
Flu shot	"season" "caught" "flu" "shot" "vaccination" "fever" "influenza"
Bird Flu	"infection" "avian" "fever" "virus" "bird" "flu" "death" "stomach" "china"
Vaccine cause cancer	"cancer" "survivors" "vaccine" "fever" "influenza" "protects" "highrisk"
Dengue fever	"chinese" "economy" "flu" "dengue" "fever" "malaria" "touch"

To evaluate the capability of the approach to discover the actual topics occurring in the datasets, i.e. the so called ground-truth, we relied on the results reported in [10, 18]. The ground-truth consists of 25 topics. Each topic is characterized by a set of keywords, namely the most frequent hashtags and terms used to describe it. Topics

range from influenza-like illness and vaccines to allergies, from insomnia and sleep issues to diet and exercise from cancer and serious illness to injuries and pain. In Table 1 a sample of ground-truth topics are listed. For each topic, we report the list of words representing it.

5.2 The Word2Vec Model

Word2Vec [16, 17] is a group of related models that are used to produce word embeddings. These models are shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words. Word2Vec takes as its input a large corpus of text and produces a vector space, typically of several hundred dimensions, with each unique word in the corpus being assigned a corresponding vector in the space. Word vectors are positioned in the vector space such that words that share common contexts in the corpus are located in close proximity to one another in the space.

Word2Vec can utilize either of two model architectures to produce a distributed representation of words: continuous bag-of-words (CBOW) or continuous skip-gram. We used the skip-gram model that uses the current word to predict the surrounding window of context words. The skip-gram architecture weighs nearby context words more heavily than more distant context words.

We trained the word embedding on two corpora by exploiting in-domain and out-domain knowledge bases.

The main idea is to employ different semantic spaces in order to find which method is more effective for learning vector representation of words to capture their semantic meaning.

The in-domain corpora is represented by the set of tweets itself. The out-domain corpus is built by exploiting health data collected always in the work proposed in [19] and made available from the website [2]. The data consists of 20,000 key phrases related to the names of diseases, symptoms, treatments and medications. The words were crawled from consumer oriented websites like wrongdiagnosis.com because the language is more likely to match the informal language used in social media as compared to language used in literature intended for medical professionals. The authors in [19] also collected articles concerning 20 health issues from WebMD about the most popular health topics featured on the homepage of WebMD. Within each health condition, they collected all articles that contained information describing the condition and its symptoms and treatments.

6 EXPERIMENTAL EVALUATION

In this section, we show the effectiveness of *WEC* by running the method on the set of health tweets described earlier. Specifically, we first evaluate the ability of the algorithm in detecting diseases and health issues by identifying the best parameter setting, also evaluating the approach with different α and β weights for the similarity measure. After that, we compare *WEC* with three state-of-the-art methods, *LDA* [7], *Doc-p* [20], and *SFPM* [3] and with *TweetHealth* [10].

6.1 Evaluation metrics

To assess the effectiveness and efficacy of the *WEC* approach the following metrics are used:

- *Event Recall*: $R = \frac{TP}{TP+TN}$ is the percentage of ground truth events successfully detected by a method, where TP (true positive) is the number of successfully detected ground-truth events, and $TP+TN$ (TN :True Negative) is the total number of ground-truth events. A ground-truth event is considered successfully detected if there exists a predicted event that matches at least one of its terms.
- *Event Precision*: $P = \frac{TP}{TP+FP}$. It is the fraction of correctly detected ground truth events out of the total number of events detected $TP + FP$ (FP: false positive).
- *F-Score*: *F-score*= $\frac{2RP}{R+P}$ is the harmonic mean of precision and recall metrics, reaches its best value at 1 and worst at 0.

6.2 Results

In a first set of experiments we aimed to identify the best parameter settings for the *WEC* algorithm.

The parameters of *WEC* are δ, the minimum survival time period, the temporal horizon a cluster is active h, and the similarity threshold ϵ. Thus, we considered different values of δ, h and ϵ, and computed the event precision and recall. In particular, Figure 3 shows the precision obtained for $\delta = 24$, $h = 12$, $\delta = 12$, $h = 6$, $\delta = 6$, $h = 3$ and $\epsilon = \{0.5, 0.6, 0.7, 0.8\}$, where the time is expressed in hours. The graph shows that the precision grows slightly with ϵ, reaching its best value in correspondence of $\epsilon = 0.7$. In fact, for higher values of ϵ the algorithm splits the same topic into multiple clusters. For what concerns the parameters δ and h we obtained the best precision value for longer survival time and temporal horizon. This is due to the fact that in target dataset several topics of discussion have a prolonged temporal extension spanning through several days. Therefore, clusters remain active for longer time periods.

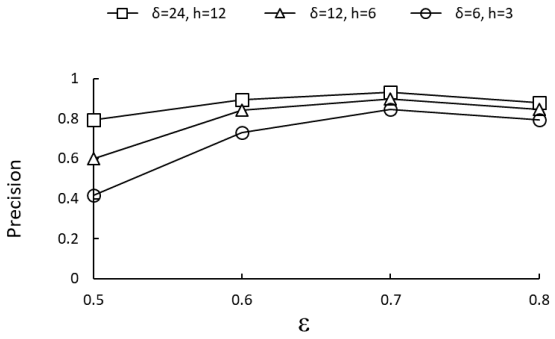

Figure 3: Precision of *WEC* w.r.t ϵ for different δ and h values.

Figure 4 points out that the event recall grows with the similarity threshold: for smaller values the algorithm groups different topics within the same cluster, failing this way to identify many ground truth topics. The best recall values is achieved when $\delta = 24$, $h = 12$, $\epsilon = 0.7$. In fact, from the graph it is evident that, for smaller values of δ and similarity threshold ϵ, the recall gets worse. What is important to note from the graph is that the temporal parameters δ and h impact significantly on the recall. As for the precision, a larger temporal horizon allows to group into the same cluster the posts

Table 2: Performance of *WEC* when using different values of α and β in the similarity metrics.

	Precision	Recall	F-score
$\alpha = 0.5\ \beta = 0.5$	0.932	0.938	0.935
$\alpha = 1\ \beta = 0$	0.920	0.659	0.768
$\alpha = 0\ \beta = 1$	0.756	0.545	0.633
$\alpha = 0.7\ \beta = 0.3$	0.915	0.743	0.820
$\alpha = 0.3\ \beta = 0.7$	0.849	0.774	0.810

relative to the same topics spanning along a longer time interval even through several days.

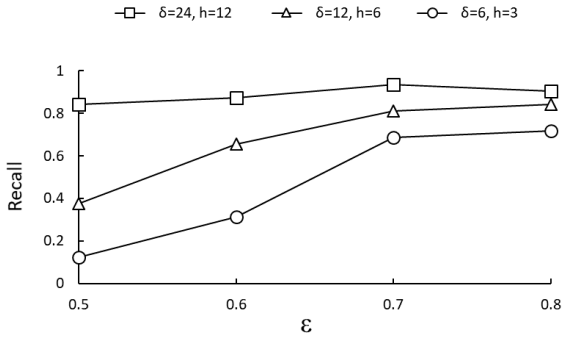

Figure 4: Recall of *WEC* w.r.t ϵ for different δ and h values.

According to the above results, the best parameter configuration is obtained with $\delta = 24,\ h = 12,\ \epsilon = 0.7$. Configuration that will be used throughout the paper.

As detailed earlier, another key aspect of the algorithm *WEC* is the similarity function (see Equation 1), exploiting both semantics and syntactic features of text. It is formulated as a combined weighted metrics where the two similarity terms contribute to the overall similarity on the basis of the value chosen for the respective weights. To this purpose we determined the best value of such weights by executing a set of experiments whose results are reported in Table 2. The results, obtained by fixing $\epsilon = 0.7$, indicate that the combination with $\alpha = 0.5$ and $\beta = 0.5$ is the one achieving the best performance in terms of both precision and recall. In fact, even if a higher value of the semantic weight produces a slightly higher precision, this improvement is payed in terms of recall.

To further highlight the impact of the similarity measure on the algorithm performance, we executed another set of experiments to compare the proposed similarity measure against some baseline models implementing either syntactic or semantic similarity. To this purpose we run the algorithm *WEC* by using alone the Levenshtein and the Jaccard similarity, as syntactic measures, only semantic similarity, as defined in equation (2), and the combination of both, as defined in equation (1). Then we computed the performance achieved in terms of precision and recall for all the measures when increasing the similarity threshold ϵ. The results reported in Table 3 show that the proposed hybrid approach, combining both semantic and syntactic relations of terms, achieved the best results. Both the syntactic models got the worst performance especially in terms of precision since they do not group correctly the posts about the same topic. This is more evident for higher values of ϵ. The semantic similarity improves the precision for higher values of ϵ but, at the same time, its recall decreases, making, thus, the hybrid approach the better choice.

6.3 Comparison with related approaches

In this section we compared *WEC* with four methods: *LDA* [23], *Doc-p* [20], *SFPM* [3], and *TweetHealth* [10].

We evaluated *WEC* against related approaches in terms of precision and recall. Figure 5 shows the precision of the different methods with the number of top detected topics. The graph outlines that, overall, for all the methods the precision slightly increases with the number of top events detected. *WEC* substantially outperformed traditional approaches to topic detection like LDA and Doc-p. Furthermore, it also improves *TweetHealth* showing that the joint use of semantic and syntactic relations of terms, together with temporal proximity of posts, highly enhance the detection performance. SFPM achieved the lowest precision that is quite stable with the number of detected topics.

Similar comments hold for the recall. Figure 6 shows that for all the methods the event recall remains quite stable when the number of produced topics is greater than 15. *WEC* achieves the higher event recall that grows with the detected topics till reaching the value of about 0.94 when $K = 15$, then the recall curve gets flat. Although *LDA* for small number of topics detected exhibits the worst performance, it then improves with the number of detected topics till reaching a good recall of about 0.8. *TweetHealth* achieved a high event recall for smaller values of detected topics then it gets flat for $K = 10$. One can note the worst performance obtained by *Doc-p*.

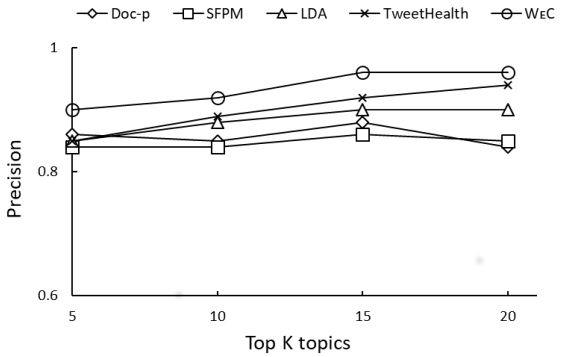

Figure 5: Precision w.r.t the top K topics for different algorithms

We used two performance indices to evaluate the quality of the clustering obtained by *WEC*, the entropy and the silhouette metrics, as detailed in the following.

The entropy of a cluster allows to understand cluster homogeneity by means of the term distribution across the social media posts. The entropy value decreases when the posts share a similar vocabulary, whereas it increases when the vocabulary varies among posts

Table 3: Performance of *WEC* when using different similarity measures.

	$\epsilon = 0.5$			$\epsilon = 0.6$			$\epsilon = 0.7$			$\epsilon = 0.8$		
	Precision	Recall	F-score	Precision	Recall	F-score	Precision	Recall	F-score	Precision	Recall	F-score
Semantic+Syntactic	0.794	0.844	0.818	0.895	0.875	0.885	0.932	0.938	0.935	0.879	0.906	0.892
Semantic	0.669	0.550	0.604	0.824	0.612	0.702	0.920	0.659	0.768	0.901	0.632	0.743
Syntactic (Levenshtein)	0.562	0.489	0.523	0.589	0.532	0.559	0.756	0.545	0.633	0.712	0.522	0.602
Syntactic (Jaccard)	0.477	0.467	0.472	0.510	0.450	0.478	0.652	0.498	0.565	0.632	0.466	0.536

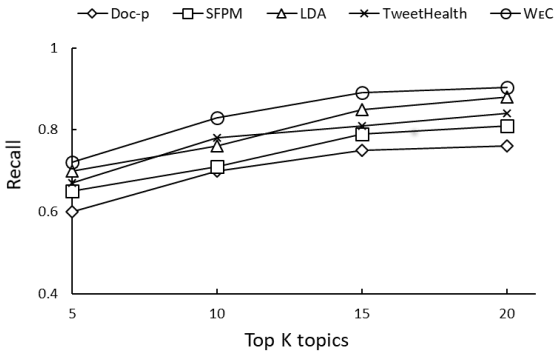

Figure 6: Recall w.r.t the top K topics for different algorithms

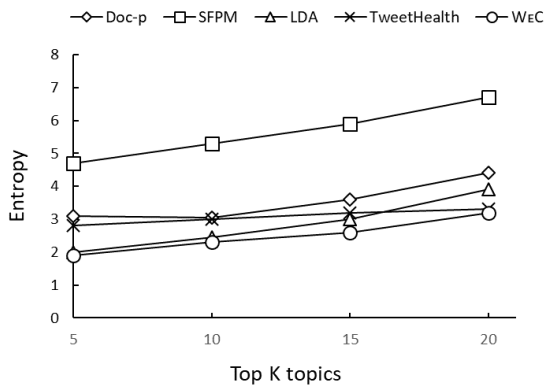

Figure 7: Entropy w.r.t. the top K topics for different algorithms.

grouped in the cluster. Thus, it gives information on how people discuss about a topic, by checking whether people talk about a topic by using similar terms.

We computed the entropy for each feature f in the feature vector fv of the centroid CC of each cluster C. Specifically, we computed the variation of the feature throughout the posts in the cluster. Entropy is computed as:

$$H(f)_C = -\sum_{i=1}^{|f|} p_i log\, p_i, \quad p_i = \frac{f_{f_i}}{N} \tag{3}$$

where $H(f)_C$ is the Shannon's entropy of feature f for the cluster C, f_{f_i} is the size of the value i of feature f (in other words its frequency), $|f|$ is the number of distinct values, N is the total size of feature f (the sum of the frequencies of the $|f|$ different values of the feature), and p_i is the observed probability of the value i.

Figure 7 shows that the entropy of the clusters obtained with *WEC* algorithm is rather low and smoothly increases with the number K of top detected topic, proving the good homogeneity of the clusters produced by the proposed algorithm. The only contestant approach that exhibits a comparable entropy is LDA but only for small values of K. *TweetHealth* entropy remains about constant with the number of detected topics and is comparable to the one of *WEC* only for $K = 20$.

Another important characteristic of a good clustering algorithm is the separability. To this aim we used the silhouette index, which is a measure of how similar an object is to its own cluster (cohesion) compared to other clusters (separation). The silhouette value ranges from -1 to 1. A high value indicates that the object is very similar to the other objects of its own cluster and it is poorly alike to objects of neighboring clusters.

Figure 8 points out that *WEC* performance is good also in terms of separability of the clusters: the silhouette value is quite high and remains rather stable with the top k detected topics. The trend of *TweetHealth* is very close to the one of *WEC*. In this case LDA is outperfomed by Doc-p. As for the entropy, one can not the bad behavior of SFPM.

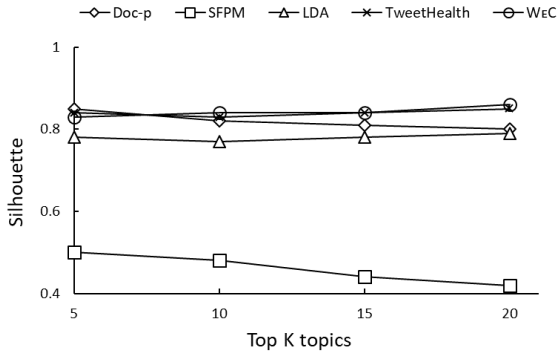

Figure 8: Silhouette w.r.t. the top K topics for different algorithms

7 CONCLUSION

The paper proposed a method for clustering topics from messages posted by users on social media. The algorithm exploits a new

similarity measure that combines syntactic and semantic information contained in the text. Latent feature vector representations of words appearing in the post are obtained by training the skip-gram model of Word2Vec on a very large corpora to improve the check-in-topic mapping learnt on a smaller corpus. Word vectors learned on neural embedding exhibit linguistic regularities and patterns which allow to obtain significant accuracy improvements when compared to purely syntactic approaches, but also with respect to state-of-the-art methods. Future work will evaluate the method on larger datasets of different contexts.

REFERENCES

[1] 2017. Amazon Mechanical Turk. https://www.mturk.com

[2] 2017. Using Machine Learning to Analyze Twitter for Real Time Influenza Surveillance. https://medium.com/@justinzcai/

[3] Luca Maria Aiello, Georgios Petkos, Carlos Martin, David Corney, Symeon Papadopoulos, Ryan Skraba, Ayse Goker, Ioannis Kompatsiaris, and Alejandro Jaimes. 2013. Sensing Trending Topics in Twitter. *IEEE Transactions on Multimedia* 15, 6 (2013), 1268–1282.

[4] Nasser Alsaedi, Pete Burnap, and Omer Rana. 2017. Can We Predict a Riot? Disruptive Event Detection Using Twitter. *ACM Trans. Internet Technol.* 17, 2 (March 2017), 18:1–18:26.

[5] Farzindar Atefeh and Wael Khreich. 2015. A Survey of Techniques for Event Detection in Twitter. *Comput. Intell.* 31, 1 (Feb. 2015), 132–164.

[6] Hila Becker, Mor Naaman, and Luis Gravano. 2011. Beyond Trending Topics: Real-World Event Identification on Twitter. In *Proceedings of the Fifth International Conference on Weblogs and Social Media*.

[7] David M. Blei, Andrew Y. Ng, Michael I. Jordan, and John Lafferty. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research* 3 (2003), 2003.

[8] Moody Christopher. 2016. Mixing Dirichlet Topic Models and Word Embeddings to Make lda2vec. *https://arxiv.org/pdf/1605.02019.pdf* (2016).

[9] Carmela Comito, Clara Pizzuti, and Nicola Procopio. 2016. Online Clustering for Topic Detection in Social Data Streams. In *28th IEEE International Conference on Tools with Artificial Intelligence, ICTAI*. 362–369.

[10] Carmela Comito, Clara Pizzuti, and Nicola Procopio. 2017. How people talk about health?: Detecting Health Topics from Twitter Streams. In *Proceedings of the International Conference on Big Data and Internet of Things,*. 85–90.

[11] Mário Cordeiro and João Gama. 2016. *Online Social Networks Event Detection: A Survey*. Springer International Publishing, Cham, 1–41.

[12] Xiangfeng Dai, Marwan Bikdash, and Bradley Meyer. 2017. From social media to public health surveillance: Word embedding based clustering method for twitter classification. In *Proc. of the IEEE SoutheastCon, Charlotte NC March 30 ? April 2, 2017*. 1–7.

[13] Maha Fraj, Mohamed Aymen Ben HajKacem, and Nadia Essoussi. 2018. A Novel Tweets Clustering Method using Word Embeddings. In *15th IEEE/ACS International Conference on Computer Systems and Applications, AICCSA 2018, Aqaba, Jordan, October 28 - Nov. 1, 2018*. 1–7.

[14] Anuradha Goswami and Ajey Kumar. 2016. A survey of event detection techniques in online social networks. *Social Network Analysis and Mining* 6, 1 (17 Nov 2016), 107.

[15] Mahmud Hasan, Mehmet A Orgun, and Rolf Schwitter. 2017. A survey on real-time event detection from Twitter data stream. *Journal of Information Science* online March 29017, https://doi.org/10.1177/0165551517698564 (2017), 1–21.

[16] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *1st International Conference on Learning Representations, ICLR 2013*.

[17] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'13)*. 3111–3119.

[18] Michael J. Paul and Mark Dredze. 2011. You Are What You Tweet: Analyzing Twitter for Public Health. In *Proceedings of the Fifth International Conference on Weblogs and Social Media, Barcelona, Catalonia, Spain, July 17-21, 2011*.

[19] Michael J. Paul and Mark Dredze. 2014. Discovering Health Topics in Social Media Using Topic Models. *PLoS ONE* 9, 8 (2014).

[20] Saša Petrović, Miles Osborne, and Victor Lavrenko. 2010. Streaming First Story Detection with Application to Twitter. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 181–189.

[21] Cao Qimin, Guo Qiao, Wang Yongliang, and Wu Xianghua. 2015. Text Clustering Using VSM with Feature Clusters. *Neural Comput. Appl.* 26, 4 (May 2015), 995–1003.

[22] Gerard Salton and Michael J. McGill. 1986. *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc., New York, NY, USA.

[23] Yee Whye Teh, David Newman, and Max Welling. 2007. A collapsed variational Bayesian inference algorithm for latent Dirichlet allocation. *Adv. Neural Inf. Process. Syst* 19 (2007).

[24] Andreas Weiler, Michael Grossniklaus, and Marc H. Scholl. 2017. Survey and Experimental Analysis of Event Detection Techniques for Twitter. *Computer Journal* 60, 3 (2017), 329–346.

[25] Yiming Yang, Tom Pierce, and Jaime Carbonell. 1998. A Study of Retrospective and On-line Event Detection. In *Proc. of the 21st ACM SIGIR (SIGIR '98)*. ACM, 28–36.

[26] Jie Yin, Andrew Lampert, Mark A. Cameron, Bella Robinson, and Robert Power. 2012. Using Social Media to Enhance Emergency Situation Awareness. *IEEE Intelligent Systems* 27, 6 (2012), 52–59.