Expert Systems With Applications 113 (2018) 397–414

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

AuthCom: Authorship verification and compromised account detection in online social networks using AHP-TOPSIS embedded profiling based technique

Ravneet Kaur*, Sarbjeet Singh, Harish Kumar

University Institute of Engineering and Technology, Panjab University, Chandigarh, India

ARTICLE INFO

Article history:
Received 26 April 2018
Revised 16 June 2018
Accepted 3 July 2018
Available online 5 July 2018

Keywords:
Authorship verification
Compromised accounts
Online social networks
Natural language processing
AHP
TOPSIS
n-grams
Stylometry

ABSTRACT

In view of the rise in security and privacy concern in social networks, there has been an inadvertent increase in research related to framing of appropriate measures to detect the security breaches in social networks. Cyber criminals are misusing social networking platforms for inappropriate and illegitimate purposes such as posting or sending of illegitimate content which a genuine user will rarely do. Hence, whenever a sensitive and unusual text is posted by a user, there is a need to authenticate whether it is posted by the legitimate owner of the account or some imposter who might have compromised the legitimate profile. The process of authentication called authorship verification helps to handle the same. In this paper, authorship verification has been performed using different textual features such as n-grams, Bag of words (BOW), stylometric and folksonomy features to examine the authorship of tweets posted by the users on the microblogging platform Twitter. Appropriate classification and statistical analysis techniques have been applied to compute different performance parameters. From the experimental analysis, an important observation found is that though char n-grams have an upper hand to other features, still other applicable measures such as word n-grams, BOW, stylometric and folksonomy features cannot be overlooked as each user maintained consistency in different set of features. Accordingly, different feature selection techniques have been used to rank and select best feature for each user. From the comparative analysis of various similarity and statistical based feature selection techniques it is observed that AHP weighted TOPSIS method surpassed others in terms of different performance parameters. Further computation as per ranked features helped to improve the result by achieving an overall average *F*-score value of 93.82%.

1. Introduction

With the increase in use of Internet, there has been an inadvertent rise in the use of social networking sites such as Twitter, Facebook, Instagram, and many others as new communication platforms for sharing information. Social network users stay in touch with their friends and communities through chat and personal messages, sharing wall posts, tweets, status updates, etc. As per a recent statistics, on an average 6000 tweets are tweeted on Twitter in a second which amounts to 50 million tweets in a day (Stats, 2018). With so much information flowing in a fraction of seconds, there is a huge concern for the credibility of information being spread. Social networking sites are found to be heavily prone to misuse by cyber criminals who always keep an evil eye on these platforms. With the intention of spreading spam and perform other malicious activities, cyber criminals are creating fake accounts or compromising legitimate accounts (Adewole, Anuar, Kamsin, Varathan, & Razak, 2017). Fake and compromised accounts are usually misused by criminals to spread illegitimate and wicked information such as vulgarity, offensive messages, promotional posts, phishing and malware related posts (Adewole et al., 2017).

In comparison to creation of fake accounts (which nowadays are quickly detected and blocked because of the prevalent verification and reporting mechanisms), compromising the legitimate accounts is relatively a safer option for criminals so as to impersonate themselves as the original user thereby hiding their true identity. As the compromised account originally belong to a genuine user, so the activities on the profile involve the mixture of usual (before compromisation) and unusual activities (after compromisation). Because of such mixed behavior, detection of breach in these accounts become difficult. Account holder and his/her connections

* Corresponding author.
E-mail addresses: ravneets48@gmail.com (R. Kaur), sarbjeet@pu.ac.in (S. Singh), harishk@pu.ac.in (H. Kumar).

https://doi.org/10.1016/j.eswa.2018.07.011
0957-4174/

may be completely unaware of the fact that this account is compromised. As a result, the account being operated, the posts being published, friend requests being sent, applications being used etc. may not be by the legitimate user. Thus, it is highly important to know the authenticity of activities being performed by a user.

Moreover, as per Pariser's Filter theory (Pariser, 2011), every user unknowingly build his own bubble space based on his interest and search patterns. Hence, being ingenuous he usually keep living in his own social space with the same likeminded people. As a result, his postings, topic of interests and social network usage build up a unique pattern which he inadvertently obeys. With an imposter exploiting the profile for his own interest base, this behavior pattern gets an automatic strict violation. This deviation in behavior is marked as a point of compromise. To detect this deviation, it is a good practice to profile and maintain a user's historical behavioral patterns and compare the incoming patterns with the stored profile. This paper focuses on mining the textual content to look for the authorship and thereby the compromisation of accounts. Textual features such as n-grams, bag of words (BOW), stylometric and folksonomic have been analyzed to check for the authorship of tweets. To facilitate this task, an authorship verification process is performed by building a historical profile of a user and thereby comparing the incoming tweets against this created profile. It is believed that users may have fluctuating moods and their way of writing may evolve over time, hence, the profile is dynamically updated so as to replace the old tweets with the most recent ones. As extra security is always a good idea, hence verification of tweets could be deployed continuously at the back end as a complementary measure to the other compromised account detection approaches. However, in order to deploy the authorship verification process at the back end, there is a strict need to evaluate the performance of different textual features.

It is hypothesized that social network users do not remain consistent on a single and moreover same set of features. For example, a user may be biased towards a topic and may have limited vocabulary, hence, his choice of words and topics in the tweets will be limited. But, there are chances that other users may not behave in the same fashion. They may tweet randomly about different topics but may have a unique writing style. Thus, in such cases, identifying users on the basis of writing style can be more useful than the choice of words and topics. Hence, there is a need to explore multiple features and check for the consistency maintained by each user on the respective set of features.

This study aim to check the applicability of authorship verification features for compromised account detection keeping the following research questions in mind:

(i) Are textual features efficient enough for authorship verification of social network content?
(ii) From the comparative analysis of different textual features, can a set of features be generalized that could always be deployed for verification task?
(iii) Do users maintain consistency in the same set of features?
(iv) What is the benefit of analyzing different features at an initial stage?

While analyzing the efficiency of various textual features, a comparative analysis of various statistical based, similarity based and multi-attribute decision making feature selection techniques is provided to select the best set of features for each respective user. Techniques namely, AHP-TOPSIS, Chi-squared, correlation feature selection, fisher score, *t*-score and Gini Index have been examined. AHP-TOPSIS compares the relative closeness of each feature to the ideal solution hence is hypothesized to perform better than other methods which only consider feature-feature or feature-class correlation.

Remainder of this paper is structured as follows. Section 2 covers the brief summary of related works in the domain of authorship verification along with an outline of different features applicable for the same. Section 3 gives a detailed explanation of the proposed approach to handle the authorship verification of online messages. Section 4 discusses the obtained experimental results followed by a comparative analysis of different feature selection techniques. Finally, some concluding remarks and future directions are presented in Section 5.

2. Related work

Authorship Verification is a subset of authorship analysis process which involves the analysis of characteristics of texts written by a user. Among three authorship analysis tasks, namely identification, verification and characterization, the domain of detecting compromised accounts could be correlated to the authorship verification process which is the process of detecting **whether the unknown text is written by the alleged user or not**? Given an unknown text and a set of online messages or posts from a known user, the task is to determine whether this unknown message/post is written by the same user or not. If the message is suspected to be not belonging to the same user, the compromisation of account is alarmed.

For the task of authorship analysis, text mining has already attracted a lot of attention in the literature (Kocher & Savoy, 2017; Stamatatos, 2009). Researchers have used different textual features such as n-grams, stylometric, semantic, idiosyncratic features for various authorship analysis tasks. As the undertaken problem is primarily concerned with the authorship verification, hence, some of the prominent research works carried out solely for the authorship verification tasks are discussed in the next subsection.

2.1. Review of authorship verification tasks for online data

Though authorship verification process has been in talks from years, but most of the prior literature only focused on the authorship verification of literary works, such as books, essays and documentaries (Halteren, 2007; Koppel, Schler, & Argamon, 2009; Koppel, Schler, & Mughaz, 2004). Online data such as e-mail, blogs, social network status and tweets differ a lot from the literary documents in terms of size as well as layout and syntactic structure. Specially, the social network data has relatively shorter length as well as poor documentation and structure. Some of the authorship verification works focusing on online data have been discussed below.

Brocardo, Traore, Saad, and Woungang (2013) performed the authorship verification process on short e-mail messages using supervised learning techniques. Instead of following the usual practice of comparing the frequency distribution of n-grams in a text, researchers focused on the binary approach i.e. presence or absence of a particular n-gram.

Later, as an extension to their previous work, Brocardo, Traore, and Woungang (2015) further used Mutual Information as a feature selection measure and followed a hybrid approach, combining outputs from two classification algorithms namely, Support Vector Machine (SVM) and Logistic regression. Experiments were performed using Twitter feeds and email data from Enron hence, giving a touch to social network scenario. The proposed model followed two steps namely, enrollment and verification. Enrollment process involved the creation of behavior profiles for a user whereas verification phase was carried out as a two class classification problem taking both positive and negative samples from the users. Appropriate classification approaches were applied for verification. In order to handle oversampling, weight equivalent to the

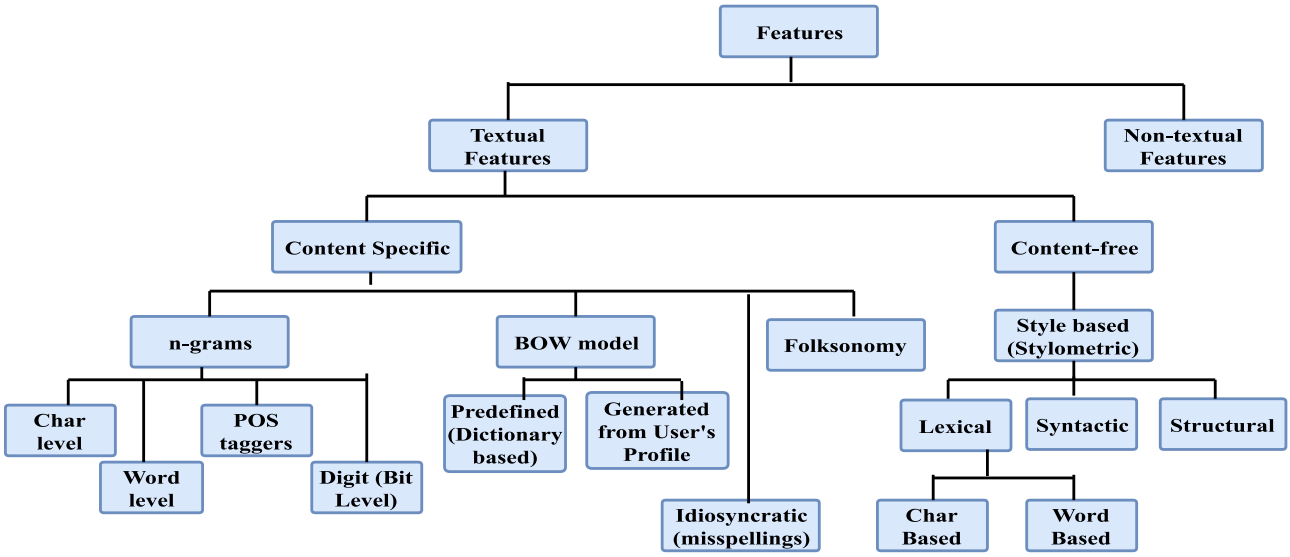


Fig. 1. Common textual features applicable for authorship verification.

ratio of positive to negative samples was assigned to the negative samples.

Furthermore, Mayor et al. (2014) used three approaches, namely, Nave Bayes Classifier (using n-grams), imposter and sparse representation for the authorship verification of online data. For profiling, a document was represented as a vector space model. In addition to the bag of words, frequency values of other features such as n-grams, prefixes, suffix, punctuation, stop words were taken into account. Sparse representation method otherwise found efficient for face recognition task, when applied to authorship verification of text outperformed the performance of n-grams and imposter methods. Similarly, Potha and Stamatatos (2014) investigated the text data in a profile based manner. Text samples by an author were profiled and an unknown text was compared with this profile to verify the authorship of the unknown text.

Instead of using n-grams and other content specific features such as Bag of words, Li, Chen, Monaco, Singh, and Tappert (2016) analyzed the efficiency of stylometric features for authorship verification (AV) of social network data stating it to be a work towards the detection of compromised accounts. Unlike existing works on AV that only focused on e-mail forums, documents and other platforms with long text messages, this was the first research towards authorship verification of short messages, especially on social networks. The work aimed to examine the efficiency of machine learning algorithms to correctly distinguish the posts of an authentic user from fraudulent users by analyzing the writing patterns and history of users. It worked as a complementary method to the existing authentication schemes which only looked for the verification on the basis of username-password at the time of login. Researchers analyzed 233 combined stylometric and social network features identifying the correct authorship of a text with 79.6% accuracy.

In a similar fashion, Iqbal, Khan, Fung, and Debbabi (2010) used stylometric features in both classification as well as regression techniques independently to perform authorship verification of e-mail messages. As a two-class classification problem, the performance of three classifiers namely, AdaBoost, Descriptive Multinomial Naive Bayes (DMNB) and Bayesian Network were analyzed using the concepts hired from National Institute of Standards and Technology (NIST) approved Speaker Recognition Evaluation (SRE) framework (Martin & Przybocki, 2009). It was observed that the performance of Bayesian network outperformed others. For performance analysis, detection error tradeoff (DET) curve was preferred to Receiver Operating Characteristic (ROC) curve as the former help to analyze both false positives as well as negatives. Similarly, performance of three regression techniques namely, linear regression, Support vector machine (SVM) with Sequential Minimum Optimization (SMO) and SVM with Radial Basis Function (RBF) were analyzed where SVM with RBF achieved better results. On the whole, performance of regression techniques superseded classification approaches.

Relating authorship verification and detection of compromised accounts, Barbon, Igawa, and Zarpelao (2017) performed a preliminary investigation to examine how effectively authorship verification tasks could help to identify compromised accounts in social networks. A pure text mining based N-gram authorship verification approach was proposed. Ease of use of n-grams motivated the authors to use the same over other alternatives such as Parts of Speech (POS) taggers and Bag of Words (BOW) models. A 3-step process including baseline creation, matching and updating was followed. Simplified Profile Intersection (SPI) was used as a similarity measure to decide the value of baseline threshold. To include dynamic profile updating, an instance based learning classifier kNN was used. Continuous baseline updating helped to improve the accuracy (by 60%) as well as cover the tendency of changes in writing style and other relevant attributes. The proposed approach proved to be highly significant by attaining an accuracy of 93%.

More recently, an efficient unsupervised authorship verification technique was proposed by Kocher and Savoy (2017) using a simple distance comparison measure called SPATIUM-L1 for the extracted *k*-most frequent words as features. The proposed technique used statistical analysis and did not involve a learning step to train and define the values of parameters. Threshold values were varied to compute the values of different performance parameters. Experiments were performed on six datasets and the proposed technique was able to achieve performance close to the baseline Meta-classifier method. Picking some random texts and users, a deeper analysis was performed which depicted the typical scenarios where SPATIUM-L1 failed to give good performance.

2.2. Authorship verification features

As shown in Fig. 1, textual features have been broadly categorized as content-specific and content free. Content-specific features deal with 'what is written' whereas content free features focus on

Table 1
Bit level 2-gram representation of "We".

Gram	00	01	10	11
Value	1	6	4	3

'how it is written'. Before proceeding further, a brief discussion of all these features is presented below.

2.2.1. Content-specific features

N-grams. N-gram approach involves the principle of using '*n*' consecutive units of text together for processing. Based upon the unit to be processed, n-gram techniques are mostly categorized as character based, word-based and bit/binary based. Other than this, various other n-gram features are also popular such as syntactic n-grams (Sidorov, Velasquez, Stamatatos, Gelbukh, & Chanona-Hernández, 2014), morphemes (Gómez-Adorno, Sidorov, Pinto, Vilariño, & Gelbukh, 2016), etc. Apart from authorship verification, n-grams have widely been used for other related domains such as plagiarism, intrusion, spam detection and many more.

N-gram approach follows a sliding window principle with window size (*n*) indicating the number of units to be taken at a time. Sliding window approach states that in every new window next '*n*' adjacent units are considered for computation. Usually in practice, an overlapping window is taken, but in general, windows may or may not overlap with each other. In this work, a set of overlapping n-grams is considered.

(i) Word n-grams: For word n-grams, a single word is taken as a unit. For example, word n-grams of the text "We are friends now" are as follows:

Unigrams (1-gram): 'We', 'are', 'friends', 'now'

Bigrams (2-grams): 'We are', 'are friends', 'friends now'

Word based n-grams give a good representation of the author's writing style but because of the variability in word lengths, they are sometimes considered a less preferred choice than other variants. As number of words in each n-gram are fixed but lengths of words are flexible hence, this approach raises some complexity issues such as use of larger space and time for processing.

(ii) Char n-grams: In char n-grams, a single character is considered as a unit i.e. words from word based n-grams get decomposed into smaller units. For example, char n-grams of the text: " We are friends now" are as follows (with overlapping window and space replaced by _

Unigrams(1-gram): 'W', 'e' , '_', 'a', 'r', 'e', '_', 'f', 'r', 'i', 'e', 'n', 'd', 's', '_', 'n', 'o', 'w'

Bigrams(2-grams): 'We', 'e_', '_a', 'ar', 're', 'e_', '_f', 'fr', 'ri' , 'ie' , 'en' , 'nd ' , 'ds' , 's_', '_n', 'no', 'ow'

Unlike word n-grams, char based n-grams have a fixed length in terms of length of units. Though they increase the dimensionality but still throughout the literature, they have shown notable performance.

(iii) Bit level n-grams: Bit level n-grams take into consideration the binary representation of the text. It involves the decomposition of characters into even smaller units i.e. '0' and '1'. In the ASCII code binary representation, text is represented in the form of '0' s and '1' s. Every character in the text is represented by its ASCII code. For example, the ASCII binary string of word "We" in the string "We are friends now" is 0101011101100101. Corresponding 2-gram and 3-gram bit level representations of the word 'We' are presented in Table 1 and Table 2 respectively.

As bit level n-grams also have a fixed length of 1 bit with only two possible outcomes, many researchers (Peng, Choo, & Ashman, 2016; Peng, Detchon, Choo, & Ashman, 2016) have relied on this bit level categorization for text analysis.

Table 2
Bit level 3-gram representation of "We".

Gram	000	001	010	011	100	101	110	111
Value	0	1	3	2	1	4	2	1

Parts-of-Speech (POS) tagging. In POS tagging, each word in a sentence is assigned a part of speech tag such as noun, pronoun, verb, adjective etc. Nowadays, more polished and advanced POS tagging tools also contain tags such as 'noun-plural', 'pronoun-possessive' etc. for better analysis. Different algorithms and in-built toolkits are available to assign POS tags to words. Examples include, Stanford Log-linear Part-Of-Speech Tagger (Part-Of-Speech, 2015) Penn Treebank (Santorini, 1990), Baum–Welch algorithm (Cutting, Kupiec, Pedersen, & Sibun, 1992) etc. Length and structure constraints in social network data usually reduce the efficiency of generalized POS taggers on such data, hence, social network specific POS taggers have also been devised in the literature (Gimpel et al., 2011).

BOW model. In Bag of words (BOW) model, either a dictionary or a data-driven approach is followed. Dictionary based approach has a predefined set of words for which the model is built i.e. it stores the frequency count of each word in the dictionary. On the other hand, data-driven approach stores all the words used in the document with its frequency count. Frequency count reflects that frequent usage of certain words by the user dominates other words. Usually, term frequency, *tf*, (count of number of occurrences of a term in the text) is used as a feature for comparing different texts. But many a times, *idf* (inverse document frequency) is also used in combination to *tf* to form a *tf-idf* measure, where *idf* represents the number of documents/texts in which the said term exists.

Idiosyncratic features. Study of idiosyncratic features involves the analysis of different grammatical mistakes, misspellings and other unusual writing errors by the user. These features also reflect the unique patterns followed by a user. As each user may have varying commands in a particular language or relevant grammatical rules, hence he/she may make common grammatical mistakes very often which could be profiled and looked for in the future messages. But such features are difficult to control because of the involved subjectivity and variation from user to user. Moreover, social networks being a free space to express ideas in any manner makes it difficult to analyze whether the grammatical error is actual or intentional.

Folksonomy features. Folksonomy is used to describe the content of a web related document (a blog, picture, video, tweet) by annotating it with the user defined tags. Hashtags being a popular entity in Twitter reflects a folksonomy concept. Hashtags state the use of a word or a phrase headed by the crosshatch symbol to define the content and context of the written tweet. In the absence of hashtag and to algorithmically define the folksonomy concept, topic modeling may be used to extract topics out of tweets and use them as tags for further processing (Xue, Qin, Liu, & Xiang, 2013). Moreover, use of folksonomy features also help to utilize the semantic behavior of a tweet, hence their presence seems viable.

2.2.2. Content-free features

Stylometric features. are one of the most traditional features used in authorship analysis which reflect the writing style of a user. These are categorized as: lexical, syntactic and structural.

Lexical features contain a set of lexical items extracted in the form of characters or words from a text. Examples include frequency count of number of words and characters, white spaces or punctuation. Other legomenas and vocabulary richness measures are also categorized under lexical features.

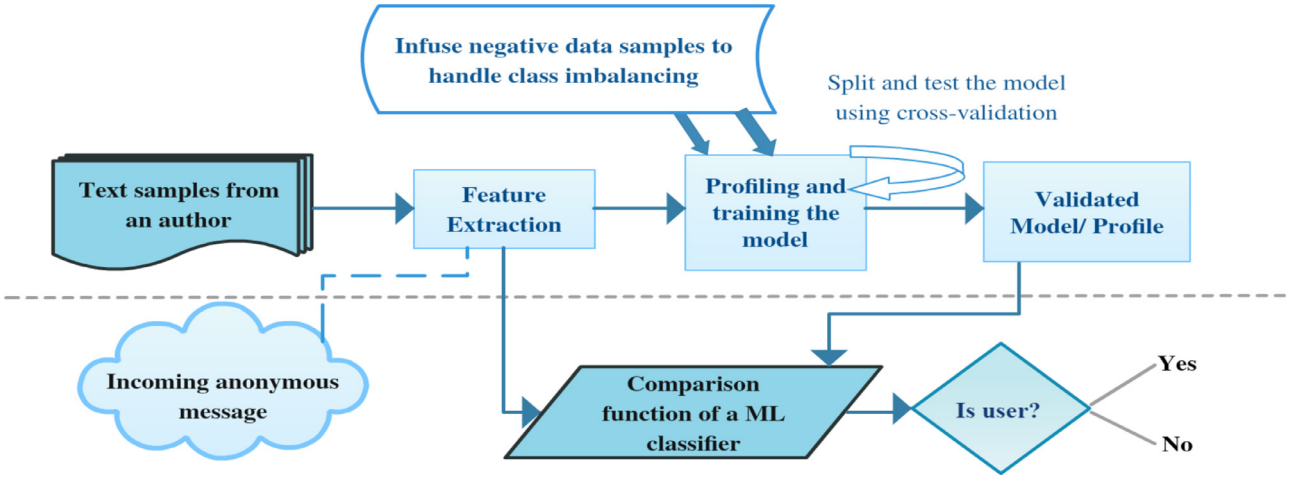
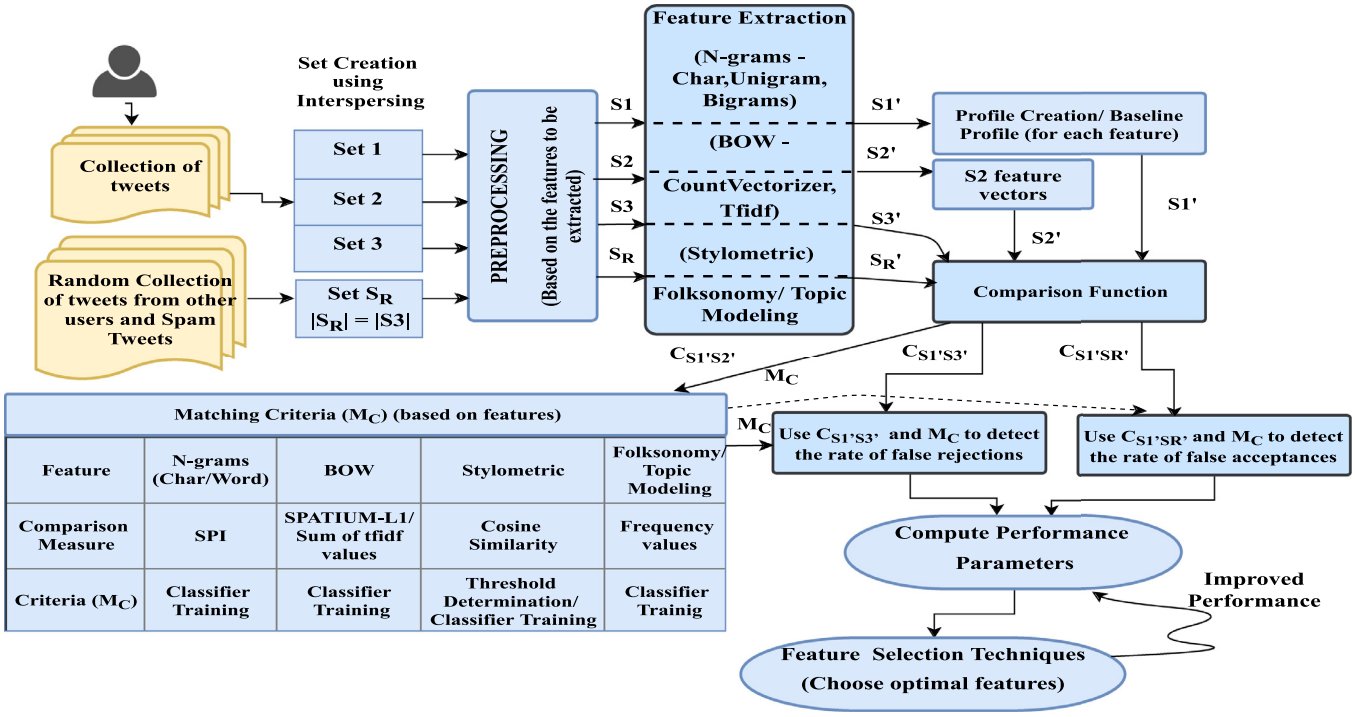


Fig. 2. Generic steps followed in authorship verification.



Feature	N-grams (Char/Word)	BOW	Stylometric	Folksonomy/ Topic Modeling
Comparison Measure	SPI	SPATIUM-L1/ Sum of tfidf values	Cosine Similarity	Frequency values
Criteria (M_C)	Classifier Training	Classifier Training	Threshold Determination/ Classifier Training	Classifier Trainig

Fig. 3. Profiling based approach for analyzing the effectiveness of different textual features for authorship verification.

Syntactic features represent a users writing style with respect to punctuation, function words, parts of speech and other such variants. Baayen, Van Halteren, and Tweedie (1996) introduced and Halteren (2007) extended the concept of punctuation and function words in syntactic features. Punctuations describe the meaning of the text by defining the boundaries around text with the use of symbols such as exclamation, question mark, quotations, etc. Similarly, function words represent a user's writing style irrespective of the topic.

Structural features are related to the organization and layout of the text. For social network messages, especially tweets which have a character count restriction, structural features such as number of lines, paragraphs, lines per paragraph, separators between paragraphs, indentation, etc. are rarely applicable.

These kind of textual features are useful to have a bird's eye view on the data. Hence, they could be easily and efficiently used for profiling the user's behavior. Accordingly, any inconsistencies found could be alarmed as the target of concern.

In this work, effectiveness of these textual features is analyzed for the authorship verification of Twitter content posted by a user. Approach followed to perform the task along with the results and observations found have been discussed in the subsequent sections.

3. Profiling based approach for authorship verification

This section presents the methodology followed to accomplish the task. A generalized approach for authorship verification as illustrated in Fig. 2 is followed in this work. A behavioral profile is built by selecting and extracting the relevant features from a user's data samples. Same set of features obtained from the unknown sample are compared against the learned patterns. After comparing the features with the already learned patterns of both positive and negative class, a class with which feature values of the unknown sample match the most is predicted as the label. As illustrated in Fig. 3, authorship verification process in this work is performed us-

ing a profiling based approach for each user. Process involves the creation of a baseline set for a user containing his/her historical tweets and thereby using this set as a standard set for comparison of incoming tweets.

Profile creation involves the process of collecting the tweets from a user and storing them in a file as a document. From this document, three independent documents or sets, namely, $S1$, $S2$ and $S3$ are generated. Rather than partitioning the file directly into three sets, tweets from the main document are placed in each set in an interspersed manner with a factor of 'r' used for interspersing. Interspersing ensures that first 'r' posts are put in set $S1$, next 'r' in $S2$, next 'r' in $S3$ and next 'r' again in $S1$ and so on. The reason why interspersing is considered a better and preferred option than direct partition is to cover neighboring tweets usually related to the same subject or trend in every set, thereby reducing the seasonality problem. Also, all further evaluation use the combination of 'r' tweets as a single tweet (t'). This is opted because of the 140 character limit constraint by the Twitter. Recently, the allowed character length has been increased to 280 characters but the tweets used in the analyzed dataset are limited to 140 characters in length. It is observed that average character length of a tweet in the given dataset is 102 characters. If only a single is considered at a time, the number of characters become very less for extracting features and building up profile characteristics. Hence, 'r' tweets are combined to form a larger text. For char n-gram features, experiments are performed with different values of 'r'. Value performing the best is then fixed as a reference for future tweets. It is seen that 48% of users performed best at $r = 4$. Hence, for other features, a static value of $r = 4$ is chosen (though it may affect the results).

One of the biggest challenge in the domain of compromised accounts is the non-availability of ground truth data consisting of the point of compromise and the compromised tweets (Kaur, Singh, & Kumar, 2018). This directed us to choose the next best logical course of action i.e. to manually inject spam and randomness into the accounts and thereby artificially embed compromise into the accounts. This practice of creating the ground truth and validating the model seems to be prolific and has been encouraged in literature (Seyler, Li, & Zhai, 2018; Trång, Johansson, & Rosell, 2015). Therefore, in addition to the creation of three sets, an additional set consisting of the unusual tweets i.e. tweets other than that of the user, is created to validate the model for the detection of compromised accounts. A set S_R generated for each user consists of equal number of tweets as that in every S set. Tweets in the set S_R consists of a mixture of random and spam tweets. Spam tweets help to cover the aspect of compromisation for malicious purpose whereas imparting the tweets from random users help to cover the non-malicious activities that could be performed after the compromisation. Because of the high likelihood of accounts getting compromised for malicious purposes, once compromised, accounts will have a lot of advertisements or spam tweets. Therefore, a set of spam tweets is collected by crawling the Twitter network using '#spam', '#BuyFollowers', '#BuyLikes', tags along with collecting the spam tweets openly available on Google. Random tweets on the other hand are the collection of randomly picked tweets from the profile of other users and are generally different from user's own tweets.

Next, respective preprocessing steps as stated in Section 4.2 are adopted for each feature evaluation. After data collection and preprocessing, any authorship analysis task, be it attribution, verification or characterization entails the identification of features and techniques applicable towards its accomplishment. In this paper, we have focused on the applicability of textual features for the verification of social network content. Both content specific and content free features have been independently analyzed. Among content specific features, char n-grams, word n-grams, data-driven BOW model and folksonomy features have been explored. In content free features, stylometric analysis of lexical and syntactic features has been performed. Structural features rarely reflect any importance in the short length messages, especially for the unstructured social network content. Hence, structural stylometric features are ignored.

$S1'$, $S2'$, $S3'$ and S_R' are the sets formed as a result of the feature extraction phase. Based on the features, these sets either correspond to the frequency distributions or simply a textual content such as n-grams. This is just a generic notation otherwise respective sets will be formed for each feature. Based on the feature in consideration, $S1'$, $S2'$, $S3'$ and S_R' correspond to the set of values of those features. $S1'$ is stored as the baseline profile with which every other available or incoming tweet is compared. This profile is stored as a baseline with dynamic updations at a later stage. With the passage of time and new tweets coming up, a FIFO (First In First Out policy) should be adopted to discard the oldest tweets and replace them with the recently recognized ones in order to keep the baseline profile updated with the latest set of tweets. Every time 'r' number of tweets are recognized as written by the given user, the oldest 'r' tweets in the baseline profile are discarded, and the new 'r' tweets just recognized are added to the profile, where 'r' represents the number of tweets validated at a time.

As a result of feature extraction, feature vectors in $S2'$ represent a training set to determine a matching criterion. Different features have different comparison functions defining respective matching criteria and approaches applied for the authorship verification. $S3'$ and S_R' act as test cases for the final performance evaluation. For each user, the construction of sets Sx' ($x = 1, 2, 3, R$), is followed by the comparison of $S1'$ and $S2'$ using a comparison/similarity detection measure to determine a threshold value. For the features involving statistical analysis, this threshold determination is sufficient to derive a formula for further evaluation of test cases. But for those features where a classification approach is used, the value of each feature derived from the comparison of each tweet t' and baseline profile is stored to train the positive class in a classifier. Similarly, the negative class is trained with values of comparison of features extracted from some random tweets and the baseline profile. As set $S3'$ belong to the actual authentic user, hence the comparison of tweets from this test with the baseline set using some predefined comparison function and matching criteria, helps to yield the true positives and false negatives (these terms being more precisely defined in Section 4). Unlike other domains, instead of treating the problem to be only one-class classification problem, a two-class classification problem is addressed where the presence of both false positives and false negatives are considered solemnly severe. To study true negatives and false positives, the same process as that for $S3'$ is repeated for S_R'. Finally, based upon the count of different parameters, various performance metrics are computed.

3.1. n-gram approach for authorship verification

This section discusses the calculation of interspersing value 'r' followed by the n-gram based verification approach.

Comparison function used in n-gram analysis is SPI (Simplified Profile Intersection) value. SPI is an intersection measure which yields as output the number of units common between the compared sets. Every Sx' set is divided into 'p' blocks where each block represents features extracted from one t' tweet. Each block 'p' in $S2'$ is compared against the baseline profile, $S1'$ to determine the common unique n-grams between the two sets. The count of common unique n-grams defines the SPI value. Though a percentage factor of the number of unique n-grams shared between $S1'$ and block 'p' of $S2'$ could also be used as a compari-

Table 3
Layout of the data for *k*-NN classifier.

Block	SPI value	Class label
$S3_{p1}'$	$V1$	SameUser
$S3_{p2}'$	$V2$	SameUser
$S3_{p3}'$	$V3$	SameUser
...	...	...
$S3_{px}'$	Vx	SameUser
S_{Rp1}'	$V1'$	NotUser
S_{Rp2}'	$V2'$	NotUser
S_{Rp3}'	$V3'$	NotUser
...	...	...
S_{Rpx}'	Vx'	NotUser

son measure but in this work, instead a simple intersecting count has been taken as a SPI value. Formally, SPI value is calculated as follows:

$$\mathrm{SPI}(S1', S2'_p) = |N(S1') \cap N(S2'_p)| \tag{1}$$

$N(S1')$ and $N(S2'_p)$ represent the set of unique n-grams in set $S1'$ and pth block of set $S2'$ respectively. For n-grams, experiments are performed with different interspersing ('r') values. Hence, for each 'r', the minimum SPI value (minSPI) obtained from each block of $S2'$ and $S1'$ is taken as a threshold (th). A subset of $S3'$ and S_R' are used to determine the false positives and negatives using this threshold. Minimum SPI value specifies that a user has at least minSPI number of common n-grams with its standard baseline profile. Hence, for each $S3'$ test case,

$$\begin{cases} tp = tp + 1, & \text{if } \mathrm{SPI}(S1', S3'_p) \geq th \\ fn = fn + 1, & \text{otherwise} \end{cases} \tag{2}$$

Similarly, for each S_R' test case,

$$\begin{cases} tn = tn + 1, & \text{if } \mathrm{SPI}(S1', S'_{R_p}) < th \\ fp = fp + 1, & \text{otherwise} \end{cases} \tag{3}$$

where tp, fn, tn and fp denote true positives, false negatives, true negatives and false positives respectively. For each user, the value of 'r' giving the highest accuracy is chosen as the best interspersing factor. Once 'r' is fixed for each user, a classification approach is followed to train the classifier with the SPI values as features and a class label. SPI values obtained with each block of $S3'$ and $S1'$ are stored followed by the label 'SameUser'. Likewise, SPI values obtained from the comparison of tweets from S_R' and $S1'$ are stored with a 'NotUser' class label. Table 3 specifies the layout of the data obtained as a result of the above.

Once, this data is obtained, a hold-out validation is applied in which SameUser and NotUser data is randomized to have a mixed and unordered data for both SameUser and NotUser categories. Thereafter, the randomized data is split into two parts. 2/3rd of the resultant data is fixed for training and the remaining 1/3rd for testing. To avoid bias, the cross validation is set to 30 to repeat the above process 30 times. Average value of each performance parameter for thirty repetitions is taken as the final output. *k*-NN (*k*-Nearest Neighbor) classifier is used to test whether the written tweets are by the actual user or not. *k*-NN is an instance-based classifier which makes it distinguishable from other supervised learning algorithms. As with other instance based learners, focus is given on the classification step rather than emphasizing much on building an accurate training data for matching. *k*-NN employees a *k*-matching criterion in which a sample is classified based on the class label of its *k*-neighbors. After finding the '*k*' neighbors, a weighing mechanism is applied to determine the most frequent class among the *k* neighbors. The elected class is then assigned as the label of the unknown instance. In our *k*-NN approach, we employed Euclidean distance as the distance measure for comparing neighbors and 1/distance as a value to be given as weights to neighbors. The above experiments are performed with char n-grams, unigrams and bigrams as features.

3.2. BOW model approach for authorship verification

Bag of Words model is used as an unordered collection of all the words in the complete data set. Frequency distribution of all the words in Set $S1'$ excluding stop words, punctuation and other special symbols (removed at the preprocessing stage) are stored during the baseline profile creation of a user. Sorting this distribution in descending order gives the sequence of most common to the least common words. BOW shows a fall in performance when we have different forms of the same word e.g. play, played, plays, etc. Even though the user is writing something related to the same topic i.e. playing but BOW will take them as different words, and hence it will have an influence on the count. Also, with the words like happy, haaaaappy, hhhaaaappppppyyy a user wants to convey the same message of being happy, but social networks being a free space to express the ideas, he/she has the right to choose any pattern. Hence classifying them in different categories will be no good. Keeping such constraints in mind, once the terms are extracted from a set, lemmatization is performed to represent different variants of a word in a standard form. It helps to cover the user's inclination for using the same word but in a different form as per the sentence structure or phrasing.

Under BOW approach, two type of frequency features have been used for classification task. Sum of term frequency-inverse document frequency (*tf-idf*) values of each word are stored and used as a feature where *tf-idf* helps to evaluate the importance of each word by representing text as a vector space model. With vector representation of baseline profile, for each term, the term frequency, *tf*, (frequency of occurrence of a term in the given tweet block '*p*') and inverse document frequency, *idf*, (logarithm of inverse of number of tweets the term occurs at least once) is used to compute *tf-idf* as follows:

$$tfidf(term) = tf(term, p) * idf(term, S1') \tag{4}$$

After sorting the computed *tf-idf* scores, the top ranked terms are picked and stored as frequently occurring terms in the users profile. These terms are then used against test sets ($S3'$ and S_R') to compute *tf-idf* values further used as features with target labels 'SameUser' and 'NotUser' respectively.

Apart from the use of *tf-idf* values as features, SPATIUM-L1, a distance measure, proposed by Kocher and Savoy (2016) is also used to find the distance in terms of probability of occurrence of each frequently occurring term in the baseline profile with the given set of tweets in the test set as follows:

$$SL(S, S') = \sum_{j=1}^{h} \left(P_S[w_j] - P_{S'}[w_j] \right) \tag{5}$$

where h defines the number of frequently occurring words taken into consideration and $P_S[w_j]$ and $P_{S'}[w_j]$ defines the probability of occurrence of word w_j in the baseline profile S and the test set S' respectively. Probabilities P_S and $P_{S'}$ are computed as ratio of term frequency of word w_j to the length of tweet. SL values for each 'p' block of $S3'$ and S_R' are stored with a label 'SameUser' and 'NotUser' respectively. Like char n-grams, a similar *k*-NN approach from partitioning the labeled set to the computation of performance values is followed for BOW scenario.

3.3. Stylometry approach for authorship verification

Under the category of stylometric features, various lexical and syntactic features are employed to check their efficiency for au-

Table 4
List of stylometric features used.

Count	Feature	Remarks	Category
		Lexical character based	
Individual character based			
26	Count of each alphabet (Aa–Zz)	Both upper and lower case letters are taken as same	C1
10	Count of each numeric character (0–9)	Digit Characters	C2
1	Number of white spaces	White space is taken as one character	C3
21	Count of each special character	~ @ # $ % ^ & * - _ + = < > [ ] { } / |	C4
Character group based			
1	Total number of alphabetic characters	All alphabetic characters taken as one. Presence of any one of them increments the counter	C6
1	Total number of uppercase letters	Counts only Uppercase letters (A–Z)	C6
1	Total number of lowercase letters	Counts only lowercase letters (a–z)	C6
1	Total number of digits	Counts all the digits	C6
1	Total number of special characters	All special characters taken as one	C6
1	Total number of special characters	All special characters taken as one	C6
1	Total number of punctuation characters	All punctuations taken as one	C6
1	Total number of characters in a tweet	Count the total number of characters	C6
1	Total count of consecutive characters	Count of 3 or more characters occurring consecutively	C9
Social networking specific characters			
1	Count of @	Count occurrences of @	C7
1	Count of #	Count occurrences of #	C7
1	Count of RT	Count occurrences of RT	C7
1	Count of http or https	Count of URLs in a users tweets	C7
12	Probability of the above characters being in the first/middle/last part of text of URLs in a users tweets	4 characters (@ # RT http) and 3 positions (first/middle/last)	C8
		Lexical word based	
Individual word based			
1	One character word	Count of words with one character	C10
1	Two character words	Count of words with two characters	C10
1	Three character words	Count of words with three characters	C10
1	Four character words	Count of words with four characters	C10
1	Five character words	Count of words with five characters	C10
1	Six character words	Count of words with six characters	C10
1	Seven character words	Count of words with seven characters	C10
1	Eight character words	Count of words with eight characters	C10
1	Nine character words	Count of words with nine characters	C10
1	Ten character words	Count of words with ten characters	C10
1	Eleven character words	Count of words with eleven characters	C10
1	Twelve character words	Count of words with twelve characters	C10
1	Words with more than twelve characters	Count of words with more than twelve characters	C10
Group word based			
1	Number of words	Count of total number of words in a tweet	–
1	Hapax Legomena	Frequency of once-occurring words	C11
1	Hapax dislegomena	Frequency of twice-occurring words	C12
1	Count of function words in each tweet	All function words taken together	C13
1	Count of short words in each tweet	All short words (with length < 4) taken together	C13
1	Count of dominant words in each tweet (including stopwords)	All dominant words including stopwords for a user taken together	C14
1	Count of dominant words in each tweet (excluding stopwords)	All dominant words excluding stopwords for a user taken together	C14
		Syntactic features	
8	Punctuation characters	, . ? ! : ; ' "	C5
303	Count of each function word	All function words taken individually.	–

Function words: a, about, above, after, all, although, am, among, an, and, another, any, anybody, anyone, anything, are, around, as, at, be, because, before, behind, below, beside, between, both, but, by, can, cos, do, down, each, either, enough, every, everybody, everyone, everything, few, following, for, from, have, he, her, him, I, if, in, including, inside, into, is, it, its, latter, less, like, little, lots, many, me, more, most, much, my, need, neither, no, nobody, none, nor, nothing, of, off, on, once, one, onto, opposite, or, our, outside, over, own, past, per, plenty, plus, regarding, same, several, she, should, since, so, some, somebody, someone, something, such, than, that, he, their, them, these, they, this, those, though, through, till, to, toward, towards, under, unless, unlike, until, up, upon, us, used, via, we, what, whatever, when, where, whether, which, while, who, whoever, whom, whose, will, with, within, without, worth, would, yes, you, your.

thorship verification. Table 4 contains the list of stylometric features used in this work. It is a subset of features used by Zheng, Li, Chen, and Huang (2006) for authorship attribution of lengthy online messages such as chatting, newsgroup, or e-mail messages. In comparison to e-mails or newsgroup messages, the length of tweets is very small, therefore, the applicability of these features on short length messages is examined in this work.

Cosine similarity is used as a comparison measure to determine the similarity score of the tweets. Writing patterns of the tweets by the same user are more similar and consistent as compared to those by other users. Hence, cosine similarity score amid the tweets by the same author will be high and with other users will be low. Also, similar and related features are grouped together to form a total of 14 categories. As a part of profile creation, normalized values of stylometric features extracted for each 'p' block of $S1'$ are grouped under the defined category and stored as a document. A similar procedure is repeated for other sets ($S2'$, $S3'$ and S_R'). It is observed that the differences between the values of a user's features and that of a random user are vital but trifling, hence, statistical analysis is preferred to classification approach. Even literature has also emphasized the use of simple statistical analysis to classifier training in many scenarios, specially for stylometric analysis (Hand et al., 2006; Kocher & Savoy, 2017). From the comparison of $S1'$ and $S2'$, a threshold value is determined which

is used as a decisive condition for future test cases. For each group of features, the minimum value of cosine similarity (*cs*) is taken as a threshold value for that respective group. Algorithm 1 is used to determine the threshold value.

Algorithm 1: Algorithm for threshold determination (*CS*).

```
Data: Array/List of Cosine similarity values (CS)
Result: Minimum Cosine Similarity (Min_th) value as
Threshold
INITIALIZE Threshold = MIN(CS);
Num_of_tweets = LENGTH(CS)
n=⌊0.1*Num_of_tweets⌋ ;        // Count of 10% of total
tweets
SET count=0;
while count≤ n do
    RESET count=0
    for each cs' value in CS do
        if cs'≤Threshold + 0.05 then
            count=count+1;
        end
        if count≥n then
            Min_th = Threshold;
            RETURN(Min_th);
            EXIT;
        end
    end
    if count≤n then
        CS.REMOVE(Threshold); // Remove threshold value
        if it is not satisfying at least 'n' tweets
    end
    if length(CS)≤n then
        EXIT ;            // No threshold value is found
        satisfying at least 'n' tweets
    end
    SET Threshold=MIN(CS) ;    // New minimum value from
    updated CS list
    SET Num_of_tweets = LENGTH(CS)
end
```

Given a set of similarity values, instead of simply selecting the minimum value which at times may be a case of an outlier, that minimum value of cosine similarity is taken which is in reach of 0.05 factor of at least 10% of the tweets i.e. around 10% of the tweets have a cosine similarity value as, $cs' \leq cs + 0.05$. This help us assure that the chosen value is not an outlier but a fair value as 10% of total tweets also have values nearby. If this condition is satisfied, then cs is chosen as the threshold, otherwise, the process is repeated for the next minimum value.

Algorithm 1 helps to select a fair threshold value by assuring that the value is being satisfied by a percentage of tweets. Apart from this, for each user, the 14 feature groups are ranked according to the average value of cosine similarity of each group. Higher the average cosine similarity value of the group better it is, hence given a lower (better) rank. Thus, from the comparison of $S1'$ and $S2'$, the threshold value, as well as ranks of groups of stylometric features, are generated. Using these measures, the test instances in $S3'$ and S_R' are evaluated. Each feature group is assigned a weight proportionate to 1/rank. Because of the presence of variations and inconsistencies in features with a higher rank (and thus lower weight), such features may be ignored. Hence, instead of matching all the features, the output is generated on the basis of a match with at least top '*d*' features. '*d*' is taken to be 3 which acts as the best combination because the value of first 3 features is approximately 1.83 (1+1/2+1/3) which is more than the sum of weights of all other group features 1.42 (1/4+1/5+···+1/14).

3.4. Folksonomy approach for authorship verification

Folksonomy (also called social tagging) concepts are used to extract both user defined as well as algorithm defined tags from the tweets. Hashtag being a popular user defined folksonomy entity in Twitter has been used to pull out the context of a tweet i.e. the topic defining the tweet. Use of hashtags has been complemented with another popular tagging entity named mentions (@). Occurrence of any hashtag or mention in the baseline profile is recorded and forthcoming tweets in the test sets are validated against the recorded hashtag/mention entities. Alternatively, a topic modeling approach using Non-Negative Matrix Factorization (NMF) has also been used to categorize tweets under various topics. NMF works on the clustering principle wherein topics are extracted based on clusters discovered from documents and the membership weights for each topic in a document. The input to the algorithm consists of a document term matrix (DTM) where terms are extracted from the baseline profile S_1' and the documents are the tweets in set S_2'. This helps to analyze both terms and the documents i.e. terms and the tweets in which the terms appear. Using DTM, two additional matrices namely, WTM (Word Topic Matrix) and TDM (Term Document Matrix) are created that help to find clusters of topics. NMF algorithm defined under matrix decomposition module of Python machine learning repository, scikit-learn, has been used to extract topics from the tweets. Extracted topics are referred as tags in folksonomic terms. Tags selected from both the techniques are tested against test sets (S_3' and S_R'). Sum of frequency of each tag is stored and used as a feature for classification task. It is hypothesized and further observed that sum of frequency count of tags remain higher for the same user as compared to other users.

4. Experimental results and discussion

This section gives a brief description about Twitter including its unique features, dataset used in this work, experiments performed and analysis of results to draw further implications.

4.1. Dataset

Twitter,[1] a microblogging service permits a user to post short texts of information called tweets (not more than 280 characters). Registered users on this platform have the permission to read and write tweets, retweet or reply to an existing tweet, send personal messages to someone and so forth. Once, a tweet is posted it automatically gets broadcasted to all the followers of a user. Social networking platforms have some unique characteristics which make them distinguishable from other online services. Use of unique keywords or symbols such as @ # RT : ≪ (...) (specific to twitter) along with the practice of using slangs, a mixture of different languages, informal writing style, and abbreviations makes the way of handling data on a social networking platform different from other domains.

To perform the experiments, a large online available data set[2] of Twitter users collected by Li, Wang, Deng, Wang, and Chang (2012) is utilized. It consists of the data from 1,47,909 users with each user having at most 500 tweets. For our experiments we required a time ordered recorded set of tweets, which were available in this data set. Alternatively, tweets could be fetched and profiled for each user using the Twitter API. But this would hardly introduce a difference as the proposed approach is a generalized one. The format of dataset is such that after a certain count there is a repetition of tweets in each file, hence, it is important

[1] https://www.twitter.com.

[2] https://wiki.cites.illinois.edu/wiki/display/forward/Dataset-UDI-TwitterCrawl-Aug2012# Dataset-UDI-TwitterCrawl-Aug2012-4.Creation.

to know the repetition point and extract only the tweets prior to that. Apart from the tweet's textual content, other related information is available in each user's file. Information include tweet id, retweet count, time and location of the tweet, favorite or not, mentioned entities, hashtags used, etc. Irrespective of this available metadata, this research work focus only on the textual content of the messages. Amongst approximately 0.15 million user profiles, 1000 users are randomly selected to perform experiments. Random selection of a user will not introduce bias, as each user in the data set has sufficient amount of tweets to distribute them for profile creation and testing phase. Experiments are performed with an assumption that the selected users are genuine and does not post spam tweets.

4.2. Data pre-processing

Based on the features being extracted, relevant preprocessing of the data is performed. For n-gram and BOW analysis, preprocessing involve the removal of stop words as the presence of articles, pronouns and prepositions do not carry any significant information and hence does not look beneficial for authorship verification. But in the case of stylometric features, stop word removal may reflect a decrease in accuracy because as per literature the use of certain function words splendidly distinguish different users (Li et al., 2016; Zheng et al., 2006).

4.3. Experimental evaluation

Independent experiments are performed with eight textual features namely, char n-grams, unigrams, bigrams, SPATIUM-L1, tfidf, stylometric, NMF and folksonomy features. For each user, a baseline profile is created with which both positive (same user) as well as negative (other users/spam) test sets are compared and analyzed. Following performance parameters are used to analyze the correct recognition rate.

Accuracy: Accuracy defines the percentage ratio of the total number of correct instances found to the total number of instances. In this work, ratio of the number of tweets correctly recognized (i.e. the ones belonging to the authentic user as well as those from the other users) to the total number of tweets under consideration defines the accuracy.

$$Acc(A_i) = \frac{\text{Correctly recognized tweets}}{\text{Total number of tweets}} \quad \forall i \in n \tag{6}$$

$$\text{Average Accuracy } (A_{avg}) = \sum_{i=1}^{n} \frac{A_i}{n} \tag{7}$$

where n in this work defines the number of repetitions performed for experiments and A_{avg} is the macro average of all repetition accuracies (A_i).

Before describing other performance parameters, the terms true positives, false positives, true negatives and false negatives need a definition as per the work undertaken.

True Positive (TP) is the correct recognition of a tweet written by the user i.e. a tweet actually written by the user is detected to be by the same user.

False Positive (FP) is the false recognition of the tweets written by an outsider to be by the user in question.

True Negative (TN) defines the case of correct recognition of a tweet written by some other user i.e. a tweet written by an outsider is recognized to be by an outsider.

False Negative (FN) is falsely recognizing a tweet written by the genuine user to be by an imposter.

As per the above metrics, other performance parameters are defined.

Precision: Precision is a performance metric computed as a result of both true and false positives leading to the correct recognition of tweets by an authentic user. Higher the precision, higher is the rate of recognition of tweets acclaimed by the same user. On the other hand, lower precision values reflect the presence of higher number of false positives i.e. tweets belonging to the outside users are incorrectly stated as being written by the user in question.

$$\text{Precision } (P) = \frac{TP}{TP + FP} \tag{8}$$

Recall: Recall is a result of both true and false negatives which determines the correct recognition of tweets by an outsider. High recall count depicts the high rate of correctly recognizing the tweets by outside users whereas low recall count marks the presence of higher number of false negatives i.e. tweets belonging to the genuine user being incorrectly recognized.

$$\text{Recall } (P) = \frac{TP}{TP + FN} \tag{9}$$

***F* -score** When presence of both false positives and false negatives is equally severe, a combination of precision and recall measures called *F*-score is used and calculated as:

$$F_{score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{10}$$

Two more parameters that reflect the rate of false positives and false negatives are added to the result section to have a percentage view of the average number of false positives and negatives produced by different features. These parameters are false rejection rate and false acceptance rate.

False Rejection Rate (*FRR*) measures the likelihood of rejecting a genuine user i.e. it defines the rate of an authentic user being incorrectly recognized as someone else. Hence, it is computed as the ratio between the number of false negatives and the total number of instances.

$$FRR = \frac{FN}{FN + TP} \tag{11}$$

False Acceptance Rate (*FAR*) is the likelihood of accepting a non-authentic user i.e. the rate by which an outsider or imposter is recognized as an authentic user. It is computed as the ratio of the number of false positives to the total number of instances.

$$FAR = \frac{FP}{FP + TN} \tag{12}$$

Co-efficient of Variance (*CV*) is a measure to define relative variability i.e. a factor to define the variability in values. It is computed using the standard deviation and mean values of respective performance parameter of all users as follows:

$$CV = \frac{\sigma}{\mu} \tag{13}$$

where, σ and μ define the standard deviation and mean of the respective performance parameters of all the users.

Remarks related to performance parameters

- Presence of a large number of false positives as compared to negatives may reflect a huge concern for security breaches and hence must be a target for minimization. Contrariwise, presence of false negatives may not show issues of security concern but still are crucial from the user point of view. A large population of people do not adopt and enroll for the e-mail and phone warning notifications because of a large rate of false alarms.
- Features should be selected so that performance parameters accuracy, precision, recall and *F*-score are as high as possible with very low values of *FRR* and *FAR*.

Table 5
Average accuracy performance of textual features for different values of k in k-NN classifier.

Features	$k = 3$	$k = 5$	$k = 7$	$k = 9$	$k = 11$
Char n-grams	86.68	86.85	87.35	87.80	87.99
Unigram	83.78	84.37	84.52	84.32	84.11
Bigram	76.57	75.21	74.01	73.90	72.83
SPATIUM L1	80.25	81.50	82.15	82.25	82.18
tf-idf	81.23	82.02	82.50	82.35	82.23
Stylometric	71.67	71.14	70.77	70.40	70.10
NMF	73.99	74.21	74.44	74.35	74.59
Folksonomy	73.18	72.79	70.16	70.16	70.05

4.4. Results and discussion

- While partitioning the sets using interspersing, the best value of '*r*' is examined only for the n-gram scenario. 48% users gave best accuracy with $r = 4$, therefore, to reduce extra processing, we avoid calculating '*r*' value for every feature type and fixed $r = 4$ as a static value. Experiments could be conducted with different values of r and checked for in case deviations in results are found.
- As stated in Barbon et al. (2017), for char based n-gram analysis, the value of $n = 6$ is used to extract the n-grams whereas for word-based n-gram analysis, experiments are performed using both unigrams ($n = 1$) and bigrams ($n = 2$).
- In k-NN classification, experiments are performed by varying the values of k (neighbors) to check different performance parameters. Table 5 outlines the result of average accuracy obtained for various '*k*' values. Experiments are performed with odd values of k to avoid conflict while performing majority voting during decision making.

Table 4 reflects that changing the value of k has a negligible difference in the results. Probable reason may have been the presence of distinguishable values for 'SameUser' and 'NotUser' in the training data. Hence, in most of the test cases, either the set of k-neighbors (3, 5, 7, 9, 11) retrieved must have been belonging to the actual class or most of the neighbors must have been given heavier weights for the actual class. Though choosing any k value doesn't add much difference to the results, still while comparing different features, arbitrarily $k = 7$ is chosen.

- Experiments are performed with 30 repetitions for each user. A 30-fold stratified shuffle split cross validation is used. In each repetition, randomization process is followed to create training and testing sets. Final output is achieved by taking the average of these 30 repetitions.
- Similarly, for word-based n-grams, experiments with both unigrams and bigrams are performed. Again SPI is deployed as a comparison measure and k-NN as a classifier. Similar procedure as that for char based n-grams is adopted to compute different performance parameters. With different values of k, maximum accuracy difference of 1.33% and 3.74% is achieved with unigram and bigram respectively. Again as a final output and for further experiments, value of '*k*' is taken as 7.
- BOW model uses the frequency distribution of commonly occurring words excluding the stop words, punctuation and other special symbols in the baseline profile of a user. Commonly occurring words extracted from the baseline profile ($S1'$) are stored as a dictionary. Sum of *tf-idf* counts of all dictionary terms in each test sample of the same user as well as the random user is stored. This data is used by the classifier for further processing. Similar to this, a distance measure named SPATIUM-L1 is also used to calculate probability of occurrence of each frequently occurring term in the baseline profile with the tweets in test set.
- In the stylometric analysis, normalized frequency count of different stylometric features are extracted and used to find threshold values as well as weights of different features. Accordingly, the test samples are classified and analyzed for correctness.
- Folksonomy concepts are also applied to extract tag information from hashtags and mentions. Also, topic modeling concepts are used to extract relevant topics from tweets using NMF. Sum of frequency count of all tags/topics are stored and used as feature for further classification task.

Table 6 gives a comparative analysis of different textual features in terms of the percentage of tweets correctly or incorrectly recognized. *Different features reflecting F-score range from 72% to 88% gives a remarkable impression towards the applicability of textual features for authorship verification of social network content, thereby giving an affirmative nod to our first research question.*

4.5. Deeper analysis

For a better analysis and understanding, text based experiments do require a deeper analysis of the obtained results. Overall performance measures only give a general behavioral trend failing to reflect the deep insights of the work. From Table 5, it is observed that for all the performance parameters, char n-gram features surpass other individual features. 86.2% average *F*-score with approximately 9% *FAR* and 17% *FRR* reflect a good performance by char n-grams. But, the parameters obtained are an average score for all users. Individual analysis of different features for each user reveals that char n-grams are not able to correctly recognize some of the users whereas other features are i.e. there are users who do not maintain consistency in char n-gram features but in some other set of features. Table 7 gives a comparison of different features on an individual user basis to reflect the percentage count of users giving better performance with the type of features.

Table 6
Comparative analysis of different textual features.

Features	Comparison metric	Accuracy (%)		Precision (%)		Recall (%)		*F*-score (%)		*FRR* (%)		*FAR* (%)	
		Average	*CV*	Average	*CV*	Average	*CV*	Average	*CV*	Average	*CV*	Average	*CV*
Char n-grams	SPI	87.35	0.13	90.63	0.14	83.25	0.17	86.20	0.15	16.75	0.15	8.79	0.14
Unigram	SPI	84.52	0.14	85.07	0.16	84.49	0.15	84.13	0.14	15.51	0.16	15.42	0.19
Bigram	SPI	74.01	0.3	75.12	0.31	93.58	0.1	80.56	0.18	6.41	0.10	44.28	0.73
SPATIUM L1	Probabilistic frequency	82.15	0.18	81.01	0.2	86.38	0.16	82.87	0.17	13.61	0.16	21.74	0.28
Tfldf Vectorizer	*tf-idf*	82.50	0.15	81.81	0.17	84.03	0.16	82.34	0.15	13.67	0.17	16.17	0.20
Stylometric	Cosine similarity	70.77	0.83	68.46	0.7	79.08	1.23	72.55	0.94	20.91	0.20	36.91	0.34
NMF	Topic frequency	74.44	0.23	71.46	0.25	86.22	0.16	77.32	0.19	13.77	0.16	36.63	0.41
Folksonomy	Tag frequency	70.16	0.33	69.8	0.34	96.45	0.07	78.13	0.19	3.54	0.07	56.38	0.79
All features	–	89.24	0.11	91.71	0.12	86.4	0.14	88.48	0.12	13.59	0.15	8.09	0.14

Table 7
Pairwise comparison of features with percentage count of users giving better performance.

Features	Char n-grams	Unigram	Bigram	SPATIUM-L1	*tf-idf*	Stylometry	Topic (NMF)	Folksonomy
Char n-grams>	–	65.97	67.31	66.28	74.94	83.50	78.24	64.43
Unigram>	34.02	–	60.82	56.59	66.80	79.79	73.09	60.31
Bigram>	32.68	39.17	–	48.55	51.64	68.55	64.63	50.82
SPATIUM-L1>	33.74	43.40	51.44	–	60.41	79.79	80.82	57.21
tf-idf>	25.05	33.19	48.35	39.58	–	76.49	68.65	57.52
Stylometry>	16.49	20.20	31.44	20.20	23.50	–	37.32	37.21
Topic (NMF)>	21.75	26.90	35.36	19.17	31.34	62.68	–	47.73
Folksonomy>	35.56	39.69	49.17	42.78	42.47	62.78	52.26	–

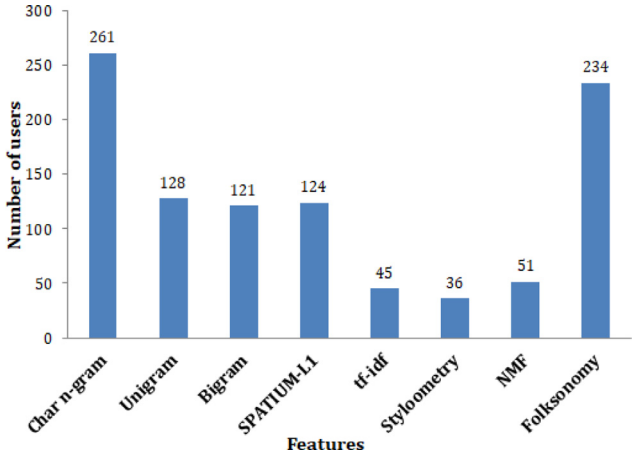


Fig. 4. Performance analysis of different features in terms of count of users.

Though as per experiments, char based n-grams have resulted in a better performance than other features, but *individual analysis reflect that the order of consistency maintained by each user is different*. Fig. 4 illustrate the comparison of individual features presenting the number of users giving better (*F*-score) performance with the type of feature.

Around 50% of users perform better with char n-grams (261) and folksonomy (234) features while remaining users gave better performance with other set of features.

More critical analysis of the results in Table 6 reveal that around 35% of users are performing better with word n-grams (unigrams or bigrams) than char n-grams. Similarly, 35.5% users performed better with folksonomy features than char n-grams. Similar is the comparison of remaining features amongst each other. There are users who even have an accuracy difference of more than 80% between different features i.e. a user is giving 80% more accuracy with one kind of feature than the other.

From the results it is observed that users maintain consistency in different features, therefore, *it is quite questionable to make a conclusion regarding char n-grams as the best features for authorship verification*. Moreover, consistency among features varies from user to user. Hence, each individual requires an independent analysis to find the best set of features for them. In view of that, a critical analysis of different features is required to check which user maintains consistency in what set of features. Therefore, a decision-making approach is adopted to extract the best set of features for each respective user.

4.6. AHP-TOPSIS based ranking of features for each user

Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) is a widely used multi-attribute decision making (MADM) method for scoring, ranking or choosing the best alternative amongst many (Saaty, 2008). Proficiently designed to handle both subjective as well as objective attributes, TOPSIS chooses the alternatives based on their closeness to the ideal solution. Rather than arbitrarily choosing the weights for each deciding attribute, a well known Analytic Hierarchy Process (AHP) is used. AHP method involves mathematical as well as psychological aspects by taking both objectivity as well as subjectivity of the decision makers into consideration. Mathematically, geometric mean, also called radical roots is used to deduce the weights. These weights are utilized in TOPSIS method to generate the ideal solutions. Finally, relative closeness of each alternative is computed to this ideal solution to decide for the ranking of different alternatives.

Experiments are performed for each of the 1000 users and it is analyzed that users did show the difference in order of preference of features. The basic methodological steps adopted to compute weights and rank the features as per AHP-TOPSIS technique are illustrated in Fig. 5. An automatic python script is written to compute the values and accordingly apply TOPSIS method for ranking the features. Practically, it is not feasible to show the results obtained for each user, hence, the stepwise implementation of TOPSIS for a randomly chosen user is demonstrated.

(i) Selecting the alternatives and attributes

While using any MADM method, two important quantities (i) alternatives and (ii) the attributes are chosen and fixed at the beginning. Alternatives state the 'objects to be selected or ranked' while attributes act as the deciding criteria for the alternate selection.

This work concentrates on the selection of best textual features for each respective user. Hence, in this work the set of various textual features act as the alternatives and to decide for the selection of these features different performance parameters are used as attributes (Table 8). Amongst all performance parameters, *F*-score, recall, precision and accuracy are desired to be maximum whereas *FAR* and *FRR* are desired to be as low as possible.

(ii) Creation of the decision table

With P alternatives and Q chosen attributes, a $P \times Q$ matrix is formed with values in each row representing the value of attribute Q_j for respective alternative P_i. Representation of the matrix $P \times Q$ is given in Eq. (14) with rows representing 'p' alternatives that needs to be ranked and 'q' columns representing the different attributes used for decision making.

Table 8
List of alternatives and attributes.

List of alternatives	List of attributes
Char n-grams (*C*)	*F*-score (*F*)
Unigrams (*U*)	Precision (*P*)
Bigrams (*B*)	Recall (*R*)
SPATIUM-L1 (*L1*)	*FAR*
tfidf (*T*)	*FRR*
Stylometric features (*S*)	Accuracy (*A*)
NMF (*N*)	
Folksonomy (*Y*)	

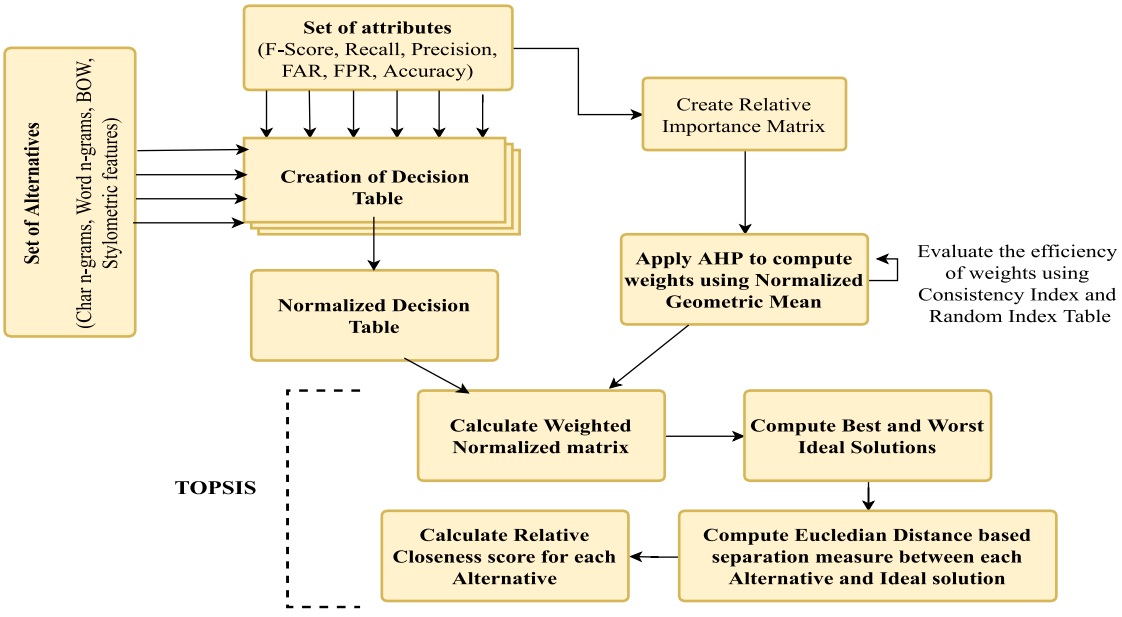


Fig. 5. Steps followed to rank features using AHP-TOPSIS.

We have 8 alternatives and 6 attributes, therefore, a 8 × 6 matrix is created. Values for a random user are inserted in this example to exhibit the procedure adopted.

$$A_{8\times 6} = L1 \begin{array}{c|cccccc} & F & R & P & FAR & FRR & A \\ C & 87.50 & 84.00 & 91.30 & 7.41 & 16.00 & 88.46 \\ U & 90.57 & 96.00 & 85.71 & 14.81 & 4.00 & 90.38 \\ B & 86.27 & 84.61 & 88.00 & 10.71 & 15.38 & 87.04 \\ L1 & 79.36 & 100.00 & 65.78 & 48.14 & 0.00 & 75.00 \\ T & 83.02 & 88.00 & 78.57 & 22.22 & 12.00 & 82.69 \\ S & 77.55 & 76.02 & 79.16 & 18.52 & 24.02 & 78.85 \\ N & 84.74 & 100.0 & 73.53 & 33.33 & 0.00 & 82.69 \\ Y & 93.88 & 95.83 & 92.03 & 3.7 & 8.04 & 94.23 \end{array} \quad (14)$$

For uniform scaling, the decision matrix (A) is normalized as:

$$B_{pq} = \frac{A_{ij}}{\sqrt{\sum_{j=1}^{q} A_{ij}^2}} \quad (15)$$

For the considered user, B, the resultant matrix formed after normalization is as shown in Eq. (16).

$$B = L1 \begin{array}{c|cccccc} & F & R & P & FAR & FRR & A \\ C & 0.3618 & 0.3901 & 0.3284 & 0.1085 & 0.4448 & 0.3674 \\ U & 0.3745 & 0.3662 & 0.3753 & 0.2168 & 0.1112 & 0.3754 \\ B & 0.3567 & 0.3760 & 0.3308 & 0.1568 & 0.4275 & 0.3615 \\ L1 & 0.3281 & 0.2810 & 0.3910 & 0.7046 & 0.0000 & 0.3115 \\ T & 0.3432 & 0.3357 & 0.3440 & 0.3252 & 0.3336 & 0.3435 \\ S & 0.3206 & 0.3382 & 0.2972 & 0.2711 & 0.6677 & 0.3275 \\ N & 0.3504 & 0.3142 & 0.3910 & 0.4878 & 0.0000 & 0.3435 \\ Y & 0.3881 & 0.4094 & 0.3598 & 0.0542 & 0.2235 & 0.3914 \end{array} \quad (16)$$

(iii) Assigning weights to attributes using AHP

To study the importance of attributes, a pair-wise comparison matrix stating the relative importance of attributes with each other is constructed using Saaty scale (Saaty, 2008) (Table 9). Assignment of importance to each attribute pair is subjective and is the decision maker's pick. Values are defined on a 1–9 scale where value a_{ij} defines the relative importance of attribute i with respect to attribute j. A value of 1 indicates both the attributes to be equally important, therefore diagonal values are always 1 because each attribute is equally important to itself. The scale afterward denotes the increasing order of importance. Matrix entries $a_{ij} = 1$ for $i = j$ and $a_{ji} = 1/a_{ij}$.

F-score covers all the crucial decisive parameters namely, true positives, false positives and false negatives. Hence, *F*-score has been assigned the highest importance. Secondly, in this work, the presence of false positives is more crucial than false negatives as an increase in the number of false positives raises the huge concern for security breaches. False negatives may not pose security threats but do infuriate users of false alarms. Though, both the factors are important, but false positives (used in Precision and *FAR*) have been given slightly more importance than false negatives (Recall and *FRR*). Thirdly, Precision and *FAR* deal with the same parameters, hence both have been given equal importance. So is the case with Recall and *FRR*.

$$M_{6\times 6} = \begin{array}{c|cccccc} & F & P & R & FAR & FRR & A \\ F & 1 & 2 & 3 & 2 & 3 & 4 \\ P & 1/2 & 1 & 2 & 1 & 2 & 3 \\ R & 1/3 & 1/2 & 1 & 1/2 & 1 & 2 \\ FAR & 1/2 & 1 & 2 & 1 & 2 & 3 \\ FRR & 1/3 & 1/2 & 1 & 1/2 & 1 & 2 \\ A & 1/4 & 1/3 & 1/2 & 1/3 & 1/2 & 1 \end{array} \quad (17)$$

Table 9
Saaty scale defining importance values (Saaty, 2008).

Importance value	1	2	3	4	5	6	7	8	9
Meaning	Equally important	Slightly important	Moderately important	Above moderate	Strongly important	Above strong	Very strong	Highly important	Extremely important

After constructing the relative importance matrix, relative normalized weights are computed for each attribute using the geometric mean method. Normalized values of geometric means for each row forms a $qx1$ matrix (NG) which represents the weights of attributes. Geometric mean (Gm) and normalized geometric mean (NG) for each row are computed using formulae stated in Eqs. (18) and (19) respectively.

$$Gm = \left(\prod_{j=1}^{q} a_{ij}\right)^{1/q} \tag{18}$$

$$NG_i = \frac{Gm_i}{\sum_{j=1}^{q} Gm_j} \tag{19}$$

The obtained Geometric and Normalized Geometric mean values are as represented in (18) and (19).

$$Gm = \begin{array}{c|c} F & 2.289 \\ P & 1.348 \\ R & 0.742 \\ FAR & 1.348 \\ FRR & 0.742 \\ A & 0.437 \end{array} \tag{20}$$

$$NG = \begin{array}{c|c} F & 0.331 \\ P & 0.195 \\ R & 0.107 \\ FAR & 0.195 \\ FRR & 0.107 \\ A & 0.063 \end{array} \tag{21}$$

For checking the consistency and correctness in weights of attributes, eigenvalue principle is adopted to calculate the consistency index (CI). To do the same, two additional matrices ($N1$ and $N2$) are formed where $N1 = MXNG$ and $N2 = N1/NG$.

$$N1 = \begin{array}{c|c} F & 2.009 \\ P & 1.175 \\ R & 0.646 \\ FAR & 1.175 \\ FRR & 0.646 \\ A & 0.382 \end{array} \tag{22}$$

$$N2 = \begin{array}{c|c} F & 6.069 \\ P & 6.025 \\ R & 6.037 \\ FAR & 6.025 \\ FRR & 6.037 \\ A & 6.063 \\ \hline (Avg\ \ \lambda) & 6.042 \end{array} \tag{23}$$

The average value (λ) from the eigenvalue matrix, $N2$ is used to calculate Consistency Index (CI), $CI=(\lambda - q)/(q-1)$ where q is the number of attributes. For efficient consistency, CI should be very small.

CI value obtained for this example is 0.0084. Random index (RI) value for 6 attributes as stated in Saaty's Random Index table (Table 10) is 1.25 and accordingly consistency ratio $CR = CI/RI$ comes out to be 0.00672 which is less than 0.1. In the literature, 0.1 has been fixed as a threshold and CR value less than 0.1 is considered within the acceptable limit.

(iv) Calculating the weighted normalized matrix

Next, a weighted normalized matrix is obtained by multiplying the elements of normalized matrix (B) with their corresponding AHP attribute weights (NG). Mathematically stating the same to be as $V_{ij} = NG_j B_{ij}$.

Table 10
Random index values Saaty (2008).

Attributes (q)	3	4	5	6	7	8	9	10
RI	0.52	0.89	1.11	1.25	1.35	1.4	1.45	1.49

The resultant matrix, C, obtained is as (24).

$$C = L1 \begin{array}{c|cccccc} & F & P & R & FAR & FRR & A \\ C & 0.1197 & 0.0761 & 0.0351 & 0.0211 & 0.0476 & 0.0231 \\ U & 0.1239 & 0.0714 & 0.0402 & 0.0423 & 0.0119 & 0.0236 \\ B & 0.1181 & 0.0733 & 0.0354 & 0.0306 & 0.0457 & 0.0228 \\ L1 & 0.1086 & 0.0548 & 0.0418 & 0.1374 & 0.0000 & 0.0196 \\ T & 0.1136 & 0.0655 & 0.0368 & 0.0634 & 0.0357 & 0.0216 \\ S & 0.1061 & 0.0660 & 0.0318 & 0.0529 & 0.0714 & 0.0206 \\ N & 0.1160 & 0.0613 & 0.0418 & 0.0951 & 0.0000 & 0.0216 \\ Y & 0.1285 & 0.0798 & 0.0385 & 0.0106 & 0.0239 & 0.0247 \end{array} \tag{24}$$

(v) Obtaining the ideal solutions

In order to judge for the beneficial and non-beneficial solutions, the best and worst ideal solutions are obtained using Eqs. (25) and (26).

$$D^+ = \begin{array}{|c|} D_1^+ \\ D_2^+ \\ D_3^+ \\ \vdots \\ D_Q^+ \end{array}, \quad D_q^+ = \begin{cases} \max(C_{pq}) & \forall q \epsilon Q \\ \min(C_{pq}) & \forall q \epsilon Q' \end{cases} \qquad p{=}1 \quad to \quad P \tag{25}$$

$$D^- = \begin{array}{|c|} D_1^- \\ D_2^- \\ D_3^- \\ \vdots \\ D_Q^+ \end{array}, \quad D_q^+ = \begin{cases} \min(C_{pq}) & \forall q \epsilon Q \\ \max(C_{pq}) & \forall q \epsilon Q' \end{cases} \qquad p{=}1 \quad to \quad P \tag{26}$$

where $Q{=}(q{=}1, 2, \ldots .Q)$ represent maximizing attributes, $Q'{=}(q{=}1, 2, \ldots .Q)$ represent minimizing attributes.

Each entry in the positive ideal solution (D^+) represents the best value of the corresponding attribute among the values of attribute for all the given alternatives. Similarly, entries in the negative ideal solution (D^-) represent the ideal worst values of attributes for the values among given alternatives. Eqs. (27) and (28) represent the ideal solutions obtained for the given example.

$$D^+ = \begin{array}{c|c} F & 0.1285 \\ P & 0.0798 \\ R & 0.0418 \\ FAR & 0.0106 \\ FRR & 0 \\ A & 0.0247 \end{array} \tag{27}$$

$$D^- = \begin{array}{c|c} F & 0.1061 \\ P & 0.0548 \\ R & 0.0318 \\ FAR & 0.1374 \\ FRR & 0.0714 \\ A & 0.0196 \end{array} \tag{28}$$

(vi) Calculating the Euclidean based separation measures

Euclidean distance between each alternative solution and ideal solution defining the separation measure is calculated as follows:

$$E_p^+ = \sqrt{\sum_{q=1}^{Q} (C_{pq} - D_q^+)^2} \qquad p = 1, 2, \ldots, P \tag{29}$$

$$E_p^- = \sqrt{\sum_{q=1}^{Q} (C_{pq} - D_q^-)^2} \qquad p = 1, 2, \ldots, P \tag{30}$$

Using the above stated formulae, following E^+ and E^- matrices are found for the given example.

$$E^+ = \begin{array}{c|c} C & 0.0501 \\ U & 0.0352 \\ B & 0.0518 \\ L1 & 0.1309 \\ T & 0.0673 \\ S & 0.0877 \\ N & 0.0875 \\ Y & 0.0241 \end{array} \tag{31}$$

$$E^- = \begin{array}{c|c} C & 0.1214 \\ U & 0.1152 \\ B & 0.1122 \\ L1 & 0.0721 \\ T & 0.0834 \\ S & 0.0853 \\ N & 0.0844 \\ Y & 0.1398 \end{array} \tag{32}$$

(vii) Calculating the relative closeness score for each alternative

The final step involves the computation of appropriate relative closeness of each alternative with the ideal solution.

$$S_p = \frac{E_p^-}{(E_p^+ + E_p^-)} \tag{33}$$

$$S_p^+ = \begin{array}{c|c} C & 0.7078 \\ U & 0.7659 \\ B & 0.6839 \\ L1 & 0.3554 \\ T & 0.5534 \\ S & 0.4929 \\ N & 0.4911 \\ Y & 0.8527 \end{array} \tag{34}$$

Arranging the generated closeness scores in descending order gives the order of preference of different alternatives. Alternative with highest score is considered as the best and most preferred alternative. For the experimented user, order of preference comes out to be:

Folksonomy features >Unigram >Char n-gram >tfidf >Stylometric >NMF >SPATIUM-L1

For the experimented user, folksonomy features turn out to be most reliable followed by n-gram features. Similar to this, AHP-TOPSIS method has been used to define the ranking of features for all other users.

4.7. Comparative analysis of AHP-TOPSIS and other feature selection techniques

AHP-TOPSIS has been compared with some of the popular filter based feature selection techniques to look for its performance efficiency. Comparison has been made with various similarity and statistical based feature selection methods namely, Chi-squared, correlational feature selection, Fisher score, *t*-score and Gini Index method. These methods handle each feature independently ignoring the feature redundancy aspect. This section continues with a brief discussion and further comparison of these feature selection techniques.

4.7.1. Fisher score

Fisher score is a similarity based feature selection technique that select features based on their value with respect to the class label. Those features are selected which have similar values for samples belonging to the same class whereas dissimilar values for features from the other class. Fisher score for each feature is calculated as:

$$Fisher_{score}(f_i) = \frac{\sum_{c=1}^{t} n_c(\mu_{ic} - \mu_i)^2}{\sum_{c=1}^{t} n_c \sigma_{ic}^2} \tag{35}$$

with $t = 2$, as the problem under consideration is a binary classification problem. n_c , μ_i, $\mu_i c$ and σ_{ic}^2 respectively represent the number of data points in class t, average of all the values for feature f_i, average of all the values of feature f_i for class t, variance of feature f_i for all the values in class t. Features are ranked according to the computed $\text{Fisher}_{\text{score}}$ and the feature with highest $\text{Fisher}_{\text{score}}$ is considered best i.e. higher the $\text{Fisher}_{\text{score}}$, better is the feature.

4.7.2. T-score

T-score is also a widely used statistical feature selection method designed to efficiently handle binary classification problems. With n_1 and n_2 denoting the number of samples in each class, *T*-score is computed using parameters such as mean and variance as follows:

$$T - score(f_i) = \frac{|\mu_1 - \mu_2|}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}} \tag{36}$$

where μ_1, μ_2, σ_1^2, σ_2^2 denote the mean and variance of values of respective binary classes. Using *t*-score, those features are selected which draws the mean of two classes statistically different. Ranking the features in descending order as per *t*-score value help to select the best feature i.e. the one with a high *t*-score value.

4.7.3. Chi-squared

Likewise statistics, where chi-squared test is applied to test the independence of events, in feature selection also it helps to test the independence of a feature and a class label. For a problem with '*c*' classes and a feature f_i with '*k*' different values, chi-square score is calculated as follows:

$$Chisquare(f_i) = \sum_{u=1}^{k} \sum_{v=1}^{q} \frac{(n_{uv} - \mu_{uv})^2}{\mu_{uv}} \tag{37}$$

where n_{uv} is the number of samples with '*u*' as the feature value. $\mu_{uv} = \frac{n_{*v} n_{u*}}{n}$ where n_{*v} indicate the number of samples in class v and n_{u^*} denotes the number of samples with '*u*' as the feature value. Again, a high chi-square value indicates a better feature.

4.7.4. Correlational feature selection (CFS)

Correlational feature selection method help to find a feature subset which is strongly correlated with the class label but is not correlated with other features i.e. a feature with good feature-class correlation ($\overline{r_{cf}}$) and weak feature-feature correlation ($\overline{r_{ff}}$). A CFS_{score} is computed for each feature as follows:

$$CFS_{score}(f_i) = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r}_{rff}}} \tag{38}$$

where, k defines the number of features to be selected. Higher is the CFS_{score}, higher is the importance of a feature.

Table 11
Overall performance obtained using different feature selection techniques.

Feature selection technique	Accuracy (%)		Precision (%)		Recall (%)		*F*-score (%)		*FRR* (%)		*FAR* (%)	
	Average	*CV*	Average	*CV*	Average	*CV*	Average	*CV*	Average	*CV*	Average	*CV*
AHP-TOPSIS	93.48	0.09	94.15	0.09	94.11	0.06	93.82	0.07	5.88	0.06	7.15	0.10
Chisquare	84.81	0.18	84.07	0.2	92.5	0.09	86.82	0.13	7.49	0.10	22.38	0.41
CFS	80.62	0.23	81.01	0.24	89.81	0.13	83.48	0.16	10.18	0.13	27.96	0.49
Fisher score	84.64	0.14	83.25	0.17	89.28	0.11	85.45	0.13	10.71	0.12	19.63	0.25
tscore	87.51	1.59	90.73	0.14	83.85	0.17	86.52	0.14	16.14	0.17	9.04	0.16
Gini Index	93.07	0.11	93.7	0.12	92.81	0.09	93.08	0.11	7.18	0.07	6.67	0.16

4.7.5. Gini Index

Gini Index is another statistical correlation based feature selection method used to select the features with an ability to distinguish the classes. It works on the principle of impurity i.e. the feature with lowest impurity is chosen. Impurity for a binary classification problem is calculated as:

$$Gini(S) = 1 - \sum_{k=1}^{2} P_k^2 \tag{39}$$

Using this impurity factor, Gini index for a feature is calculated by considering all the available values of that feature. Gini score for each feature value is calculated and subtracted from overall impurity. Minimum value among all the obtained values is taken as the Gini Index for that feature. Suppose a feature has '*r*' different values, then for each '*r*', Gini score is calculated as:

$$Gini_j = Gini(S) - \left(\frac{|n_j|}{|n|} Gini(S_{nj}) + \frac{|\overline{n_j}|}{|n|} Gini(S_{\overline{nj}}) \right) \tag{40}$$

where $j \in$ r and $Gini(S_{nj})$ and $Gini(S_{\overline{nj}})$ computes the occurrence of j and values other than j in each class respectively. The minimum Gini score value obtained for different '*j*' values is stated to be the Gini index of the feature f_i. Unlike other feature selection measures, feature with the minimum Gini index value is considered the best.

4.7.6. Performance analysis of different feature selection techniques

As evident from the experimental results in Sections 4.4 and 4.5, users do maintain consistency in different set of features and for each user an independent analysis is required to find the best set of features. Taking only the top ranked feature as per the respective feature selection technique, experiments are repeated for test sets *S*3' and S_R'. Decisions are taken according to the respective highly ranked feature of each user. Overall average score of various performance parameters for different feature selection techniques are tabulated in Table 11. It is evident from Table 11 that AHP-TOPSIS surpassed other feature selection techniques in terms of all the performance parameters followed by Gini Index which also gave noteworthy performance.

Table 11 contains the average value of performance parameters for all users. Individual analysis of each user reveals that around 61% of users performed better using AHP-TOPSIS method than other methods. Fig. 6 presents the performance analysis of different feature selection techniques in terms of count of users.

In order to analyze the efficiency of each feature with various feature selection techniques, Fig. 7 illustrate the variation in behavior by plotting the top ranked feature and the count of users adhering to that feature.

From Fig. 7, it is observed that each feature plays an important role in the overall performance computation. Selection of feature depends on the consistency of a user on that particular feature. Hence, choice of best feature for each user is different. Ranking of features reflect the order of consistency maintained by each user for different features. With an exception of few, most of the users

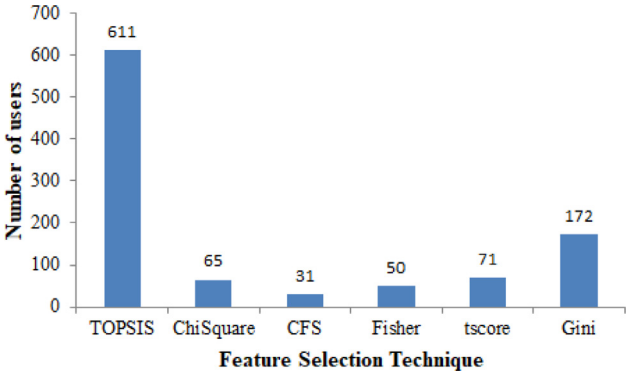


Fig. 6. Performance analysis of different feature selection techniques in terms of count of users.

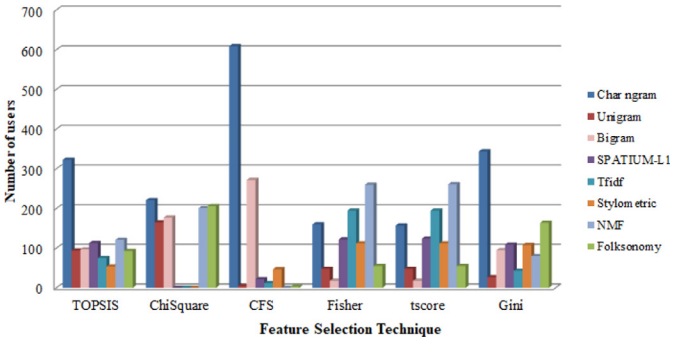


Fig. 7. Top ranked feature for different users with respective feature selection technique.

are considered as being consistent with their top ranked features and are very unlikely to show huge deviation in behavior for those features. Hence, a final call of decision is made as per the decision by the top ranked features. *Selecting the best set of features for each user and storing them for future reference helps to evade the practice of judging the user equally on the basis of each feature. Rather deviation in behavior could be analyzed and weighted as per the rank of features. For each user, a small deviation in better ranked features need a thorough investigation whereas small deviation in lower features could be overlooked. Alternatively, a reliable decision could be made for each user based upon the decision by their respective highly ranked feature.* A very few users who had performance parameters par below the acceptable level are seen not to show consistency among any of the considered features. Precisely speaking they are not found to be consistent in their textual behavior. For them, probably considering some non-textual features could help generate good results. But this could be a subject of further research.

Finally an overall performance comparison of individual features and that by the features selected for each respective user using different feature selection techniques have been shown in Fig. 8.

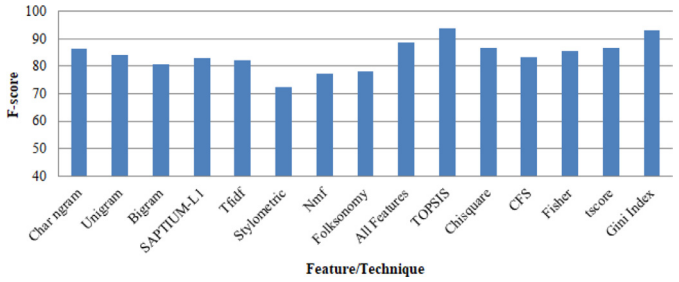


Fig. 8. Performance analysis of individual features and feature selection techniques.

Overall it is observed that working with the features selected through AHP-TOPSIS technique helped to attain better performance both in terms of overall performance parameters as well as the number of users.

4.8. Advantage and practical applicability of the proposed technique

Undoubtedly, for the detection of compromised accounts in social networks, apart from extracting and analyzing the textual features, numerous other ways could be employed, such as, checking the authenticity of a user using mouse dynamics, keyboard strokes, physiological biometrics, face/iris/fingerprint recognition, behavioral biometrics, user writing style, analyzing the habitual patterns of a user, multi-modal features and many others. But these physical approaches are too costly to be deployed as they require specialized hardware support. Moreover, once forged or compromised it becomes extremely difficult to replace them. On the other hand, authorship verification task can be applied unobtrusively at the back end without the active participation of the concerned user. The required information could be gathered smoothly at the service provider's end so as to not have the unnecessary active involvement of a user every time a check needs to be done. Also, no specially designed hardware is required for the profiling based authorship analysis task.

5. Conclusion and future work

Though in literature, there has been a lot of speculation about the best set of features for authorship verification task, but experimental results reveal that a single and that too same set of features can never be generalized for every user. Taking subjectivity, freedom of expression and way of expressing in social networks into consideration, behavior in social networks is never stated to be consistent among all users. However, because of the human nature of conforming to his/her own behavior, a user is found to remain consistent in some of his/her own set of activities and way of performing them. Experiments reveal that overall, char n-grams show better performance than other features with around 87% of tweets being correctly recognized for each user. But instead of relying on the average value, when the individual analysis is performed, some users are found not to conform to the consistency in their n-gram profiles. AHP-TOPSIS method is used to rank and give appropriate weights to different features for each user based on the different performance parameter values. Computation as per ranked features helped to improve the result by achieving an overall average *F*-score value of 93.8%.

In the near future, work will be further expanded by deeply analyzing the consistency maintained by various users on different features. Secondly, other remaining features such as idiosyncratic features, syntactic and semantic features, BOW model with predefined dictionary set etc. will be applied to check for their efficiency as well. Besides this, for the authorship verification and one of its application towards the detection of compromised accounts, apart from the textual features, other non-textual features may also reveal some interesting patterns, hence, such features also need a thorough examination. Also viability of various machine learning algorithms with varying parameter settings will be examined.

Acknowledgment

The work was financially supported through Visvesvaraya PhD Scheme by Ministry of Electronics and Information Technology (MeitY) under Ministry of communications and IT, Government of India with Grant Number: PhD/MLA/4(61)/2015-16.

References

Adewole, K. S., Anuar, N. B., Kamsin, A., Varathan, K. D., & Razak, S. A. (2017). Malicious accounts: Dark of the social networks. *Journal of Network and Computer Applications, 79*, 41–67.

Baayen, H., Van Halteren, H., & Tweedie, F. (1996). Outside the cave of shadows: Using syntactic annotation to enhance authorship attribution. *Literary and Linguistic Computing, 11*(3), 121–132.

Barbon, S., Igawa, R. A., & Zarpelao, B. B. (2017). Authorship verification applied to detection of compromised accounts on online social networks. *Multimedia Tools and Applications, 76*(3), 3213–3233.

Brocardo, M. L., Traore, I., Saad, S., & Woungang, I. (2013). Authorship verification for short messages using stylometry. In *Computer, information and telecommunication systems (cits), 2013 international conference on* (pp. 1–6). IEEE.

Brocardo, M. L., Traore, I., & Woungang, I. (2015). Authorship verification of e-mail and tweet messages applied for continuous authentication. *Journal of Computer and System Sciences, 81*(8), 1429–1440.

Cutting, D., Kupiec, J., Pedersen, J., & Sibun, P. (1992). A practical part-of-speech tagger. In *Proceedings of the third conference on applied natural language processing* (pp. 133–140). Association for Computational Linguistics.

Gimpel, K., Schneider, N., O'Connor, B., Das, D., Mills, D., Eisenstein, J., et al. (2011). Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies: Short papers-volume 2* (pp. 42–47). Association for Computational Linguistics.

Gómez-Adorno, H., Sidorov, G., Pinto, D., Vilariño, D., & Gelbukh, A. (2016). Automatic authorship detection using textual patterns extracted from integrated syntactic graphs. *Sensors, 16*(9), 1374–1393.

Halteren, H. V. (2007). Author verification by linguistic profiling: An exploration of the parameter space. *ACM Transactions on Speech and Language Processing (TSLP), 4*(1), 1–17.

Hand, D. J. (2006). Classifier technology and the illusion of progress. *Statistical Science, 21*(1), 1–14.

Iqbal, F., Khan, L. A., Fung, B., & Debbabi, M. (2010). E-mail authorship verification for forensic investigation. In *Proceedings of the 2010 ACM symposium on applied computing* (pp. 1591–1598). ACM.

Kaur, R., Singh, S., & Kumar, H. (2018). Rise of spam and compromised accounts in online social networks: A state-of-the-art review of different combating approaches. *Journal of Network and Computer Applications, 112*, 53–88.

Kocher, M., & Savoy, J. (2016). Unine at clef 2016: Author profiling. In *CLEF (working notes)* (pp. 903–911).

Kocher, M., & Savoy, J. (2017). A simple and efficient algorithm for authorship verification. *Journal of the Association for Information Science and Technology, 68*(1), 259–269.

Koppel, M., Schler, J., & Argamon, S. (2009). Computational methods in authorship attribution. *Journal of the Association for Information Science and Technology, 60*(1), 9–26.

Koppel, M., Schler, J., & Mughaz, D. (2004). Text categorization for authorship verification. In *Eighth international symposium on artificial intelligence and mathematics. Fort Lauderdale, Florida* (pp. 1–11). http://rutcor.rutgers.edu/~amai/aimath04/specialsessions/koppel-aimath04.pdf.

Li, J. S., Chen, L.-C., Monaco, J. V., Singh, P., & Tappert, C. C. (2016). A comparison of classifiers and features for authorship authentication of social networking messages. *Concurrency and Computation: Practice and Experience, 29*(14), 1–15.

Li, R., Wang, S., Deng, H., Wang, R., & Chang, K. C.-C. (2012). Towards social user profiling: Unified and discriminative influence model for inferring home locations. In *Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1023–1031). ACM.

Martin, A., & Przybocki, M. (2009). *The NIST speaker recognition evaluation series*. National Institute of Standards and Technology. [Online]. Available: http://www.nist.gov/speech/tests/spk.

Mayor, C., Hernández, J. G. G., Castro, A. I. T., Martinez, R., Ledesma, P., Fuentes, G., & Ruíz, I. V. M. (2014). A single author style representation for the author verification task.. In *CLEF (working notes)* (pp. 1079–1083).

Pariser, E. (2011). *The filter bubble: What the Internet is hiding from you*. Penguin UK.

Part-Of-Speech, S. L.-l. (2015). *Tagger, 2011*. The Stanford Natural Language Processing Group. Retrieved May.

Peng, J., Choo, K.-K. R., & Ashman, H. (2016). Bit-level n-gram based forensic authorship analysis on social media: Identifying individuals from linguistic profiles. *Journal of Network and Computer Applications, 70*, 171–182.

Peng, J., Detchon, S., Choo, K.-K. R., & Ashman, H. (2016). Astroturfing detection in social media: A binary n-gram–based approach. *Concurrency and Computation: Practice and Experience, 29*(17), 1–14.

Potha, N., & Stamatatos, E. (2014). A profile-based method for authorship verification. In *Hellenic conference on artificial intelligence* (pp. 313–326). Springer.

Saaty, T. L. (2008). Decision making with the analytic hierarchy process. *International journal of services sciences, 1*(1), 83–98.

Santorini, B. (1990). *Part-of-speech tagging guidelines for the penn treebank project (3rd revision)*.

Seyler, D., Li, L., & Zhai, C. (2018). Identifying compromised accounts on social media using statistical text analysis. arXiv:1804.07247.

Sidorov, G., Velasquez, F., Stamatatos, E., Gelbukh, A., & Chanona-Hernández, L. (2014). Syntactic n-grams as machine learning features for natural language processing. *Expert Systems with Applications, 41*(3), 853–860.

Stamatatos, E. (2009). A survey of modern authorship attribution methods. *Journal of the Association for Information Science and Technology, 60*(3), 538–556.

Stats, I. L. (2018). Twitter usage statistics. http://www.internetlivestats.com/twitter-statistics/, [Online; accessed 29-March-2018].

Trång, D., Johansson, F., & Rosell, M. (2015). Evaluating algorithms for detection of compromised social media user accounts. In *Network intelligence conference (ENIC), 2015 second European* (pp. 75–82). IEEE.

Xue, H., Qin, B., Liu, T., & Xiang, C. (2013). Topical key concept extraction from folksonomy. In *Proceedings of the sixth international joint conference on natural language processing* (pp. 480–488).

Zheng, R., Li, J., Chen, H., & Huang, Z. (2006). A framework for authorship identification of online messages: Writing-style features and classification techniques. *Journal of the American Society for Information Science and Technology, 57*(3), 378–393.