Journal of Network and Computer Applications 70 (2016) 171–182

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

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

Bit-level *n*-gram based forensic authorship analysis on social media: Identifying individuals from linguistic profiles

Jian Peng [a], Kim-Kwang Raymond Choo [a,b,*], Helen Ashman [a]

[a] School of Information Technology and Mathematical Sciences, University of South Australia, Australia
[b] School of Computer Science, China University of Geosciences, Wuhan, China

ARTICLE INFO

Article history:
Received 20 November 2015
Received in revised form
24 March 2016
Accepted 4 April 2016
Available online 6 April 2016

Keywords:
n-Gram
Profiling
Authorship
Social media

ABSTRACT

Users interact with social media in a number of ways, providing a variety of data, from ratings and approvals to quantities of text. Public discussion for hotspots in particular generates significant volume and velocity of user-contributed text, frequently attributable to a user identifier or nom de plume. It may be feasible to determine authorship of various tracts of text on social media using *n*-gram analysis on the bit-level rendition of the text. This paper explores the facility of bit-level *n*-gram analysis with other statistical classification approaches for determining authorship on two months of captured user postings from an online news and opinion website with moderated discussion. The results show that this approach can achieve a good recognition rate with a low false negative rate.

1. Introduction

With the rapid growth of social media, they have become one of the most popular ways for people to communicate. Many people prefer posting their viewpoints on discussion groups or rating movies, books and other items online rather than writing letters to newspapers or talking on radios or TVs in traditional media. These discussions and ratings can produce significant quantities of data. However, social media also permits users to assume multiple or false identities, and for example, some people can either pretend to be others to submit their postings or ratings (Krambia-Kpardis, 2004; Ratkiewicz et al., 2011a; Chen et al., 2013; Koppel et al., 2009), or even steal others' private information (Wu et al., 2013). It then becomes necessary to determine the authorship of postings according to the user's behaviour (such as input text) instead of according to their claimed credentials.

Postings on social media have several characteristics which compound the challenges of conducting authorship analysis, compared to authorship analysis for traditional text documents such as literary works, articles and emails. Firstly, social media postings are generally variable in length and may involve multiple topics. An author's writing style can be affected by different topics and different replies/comments (e.g. supportive, negative and aggressive). Secondly, they may contain less information compared to other electronic communications such as email (e.g. email metadata). Thirdly, people often do not pay attention to spelling and grammar rules in the informal environment (e.g. using abbreviations, such as “LOL” for “laughing out loud”, and emoticons), which complicates the use of high-level language attributes such as syntax and semantics. The method proposed in this paper aims to enable authorship attribution and verification, and it does this using simple analysis methods which do not require any interpretation of the text, while existing solutions are mostly designed with complicated techniques (Pfleeger and Caputo, 2012).

What can be our human's unique characteristics and how can we extract them? There are generally two approaches for these questions: one is a physical approach and the other is a behavioural approach. The physical approach commonly represents a person's profile by face, fingerprint, hand geometry, iris, keystrokes, swipe screen use, and so on. As these features are distinct between different people, they can be used to distinguish individuals from each other. Due to their high accuracy, they are widely used in our daily life (Mazhelis and Puuronen, 2007; Crawford et al., 2013; Colombini et al., 2012; Clara Maria Colombini1, 2012). However, there are some drawbacks with this method. One is that these features are easily forged or removed from their true context, and that once compromised, it is not possible to replace them. Another drawback is that it may require additional hardware to obtain them, which increases the cost (Bailey et al., 2014; Ngugi et al., 2011).

The behavioural approach is based on users' behaviours,

[*] Corresponding author at: School of Computer Science, China University of Geosciences, Wuhan, China; and School of Information Technology and Mathematical Sciences, University of South Australia, Australia.

E-mail address: raymond.choo@fulbrightmail.org (K.-K. Choo).

http://dx.doi.org/10.1016/j.jnca.2016.04.001
1084-8045/© 2016 Elsevier Ltd. All rights reserved.

preferences, intelligences, strategies, and so on (Hirst and Feiguina, 2007; Xinyi Huang et al., 2014). For example, when analysing text for the purpose of authorship attribution or for plagiarism, we extract an author's high-level features, like writing style, for analysis. Specifically, we may use some basic statistics such as word frequencies and/or other high-level data such as syntax structure and structural information. These features can accurately reflect the author's preferences and strategies when writing the text (Keselj et al., 2003).

Analysing text for the purpose of authorship attribution has antecedents in areas such as determining the true authorship of some of Shakespeare's works (Frantzeskou et al., 2006). A common method is to classify an unknown author's text into one of a set of predefined authorship candidates based on sample text known to belong to each candidate (Stamatatos, 2009; Holmes, 1994; Iqbal et al., 2008). Therefore, the main problem for this task is how to define individual profiles so as to distinguish different persons by their texts. Some linguists call these stylometries, and others name them writing styles, writprints, or simply writings (Iqbal et al., 2008; Afroz et al., 2012; Iqbal et al., 2010; Yang, 2010). These text profiles are also applied in plagiarism detection (Stein et al., 2010; Ali et al., 2011; Maurer et al., 2006). Generally, this problem is associated with the following two procedures: profile selection and text classification.

Profiling text can be achieved with different user characteristics. Some researchers use simple characteristics of text like lexical features such as word length, word frequencies, word *n*-grams, vocabulary richness, and similar (Hirst and Feiguina, 2007; Keselj et al., 2003; Frantzeskou et al., 2006; Stamatatos, 2007; Houvardas and Stamatatos, 2006; Cesare et al., 2014; Wen et al., 2014; Wen et al., 2015). Others make use of more complicated ones such as syntactic features, semantic features, and application-specific features (Stamatatos, 2009; Ratkiewicz et al., 2011b; Steve Martin et al., 2005). While higher-level text profiles can better reflect authors' writing styles, they require complicated algorithms and may be more costly. Thus, lower-level text profiles are mostly employed for text analysis (Hirst and Feiguina, 2007; Keselj et al., 2003; Frantzeskou et al., 2006; Stamatatos, 2007; Houvardas and Stamatatos, 2006). As complementary methods, higher-level text profiles are also employed to validate and improve the more basic analyses (Ratkiewicz et al., 2011a; Chen et al., 2013; Hirst and Feiguina, 2007; Houvardas and Stamatatos, 2006; Yang and Fang, 2013).

Selecting text profiles is not trivial. Sometime a certain writer's characteristic is highly relevant, but the characteristic may not be just the author's writing style. The characteristic might be content-related in addition to the writing style, e.g. although some medical-related characteristics may extremely be distinct in a medical paper, they may be not good candidate for the author's writing styles as they are topic-dependent and can be changed in different topics (Stamatatos, 2007).

Many classification methods have been successfully used in text attribution. These methods include *n*-gram analysis, support vector machine, Naive Bayes classification, decision tree, neural network, and other statistical methods. A number of researchers employ simple classification approaches because these methods do not involve complicated analysis, such as semantic and structural analyses. In particular, as the *n*-gram method can retain both lower-level and higher-level characteristics of text, it is widely used in text analysis (Hirst and Feiguina, 2007; Keselj et al., 2003; Frantzeskou et al., 2006; Stamatatos, 2009; Stamatatos, 2007; Houvardas and Stamatatos, 2006; Patrick Juola et al., 2013). The most-used *n*-gram methods are word, character, and byte based. Although character- and byte-based *n*-grams overcome the shortcoming of invariant length, there are known limitations with these two analyses, for example, the character-based method has been shown not to be suitable for processing languages with multi-byte characters (Stamatatos, 2009). The multi-bit-based *n*-gram method proposed in this paper can significantly simplify text processing, as there are only two possible statuses for each bit.

In this paper, we employ the bit-level based *n*-gram approach to create author profiles and use two statistical classifiers, the Euclidean distance and Interquartile Range methods, to measure the similarity between writing styles in collections of forum postings. The hypothesis is that different authors manifest distinguishable, individual writing styles when they compose their posts on social media. The results show the methods are effective for determining text authorships in social media with multiple topics.

This method has the potential to be extended to other applications. It has the potential to apply not only to verify the authorship for a suspected text, but for other profiling purposes such as classifying writers as bots or humans, or native versus second-language speakers, e.g. if a text expresses too many regularities or duplications it may be more likely to have been produced by a bot, or if a text includes lots of slang or idiomatic phrases, it is more likely to have been written by a native speaker of the language (Koppel et al., 2009; Manoj Harpalani et al., 2011; Shane Bergsma and Yarowsky, 2012). This is the subject of ongoing work.

The rest of the paper is organized as follows. Section 2 reviews related work. Sections 3 and 4 describe the data collection methodology and analysis, respectively. Then, we explain how the text is classified in Section 5. We discuss the proposed approaches in Section 6, and conclude the paper in Section 7.

2. Related work

In this section, we review the existing literature on the use of *n*-gram analysis and forensic linguistics. *n*-Gram techniques generally include three categories, based on what the unit to be regarded in the processed signal series. These categories are character/byte-based *n*-grams, word-based *n*-grams, and bit/binary-based *n*-grams. *n*-Grams can preserve simple text features, such as lexical information, and complex text writing styles, such as syntactic and structural information. In contrast, traditional forensic linguistics usually targets more complicated attributes for sophisticated analysis.

2.1. *n*-Gram analysis

n-Gram analysis has been widely used for many years in different domains, such as authorship attribution (Keselj et al., 2003; Houvardas and Stamatatos, 2006; US Military, Replace; Cavnar, 1994), intrusion detection (Abou-Assaleh et al., 2004a, 2004b, 2004c; Huang and Stamp, 2011; Wressnegger et al., 2013), plagiarism identification (Stamatatos, 2009), and file type identification (Wei-Jen et al., 2005). It can also be considered a more general case of frequency analysis, used in cryptanalysis. Word-based *n*-gram analysis techniques (Barrón-Cedeño et al., 2010), byte- or character-based approaches (Stamatatos, 2009; Cavnar, 1994; Abou-Assaleh et al., 2004a), and bit-level or binary methods (Ullmann, 1977) have also been used in the literature.

Cavnar and Trenkle propose a character *n*-gram approach for text categorization. They conducted two experiments: the language classification for different languages, and subject categorization within one language. The data comprised emails collected from the Usenet Newsgroups, each between 20 kb and 120 kb in length. The frequency profiles are from the *n*-grams and each is in the order of 4 kb. They collected about 3700 language samples from the soc.culture newsgroup for their language classification test. Their first experiment shows that the overall Percent Correct

Classification (PCC) is above 92% with 7% of the overall Ratio Incorrect Classification (RIC) when the length of profile is less than 300; otherwise, the PCC goes over 98% with less than 1.5% of RIC. In their second experiment, the subjects of the test data include "Security FAQ", "AI FAQ", "Compilers FAQ", "Compression FAQ", "JPEG-Compression FAQ", and "Go FAQ". The sizes of articles are between 21k and 132k. They selected 128 articles in "security", 125 articles in "AI", 66 articles in "Compilers", 187 articles in "Compression", and 252 articles in "Graphics". The result shows it correctly classifies the "Security" and the "Compilers" in its first choice, the "AI" and the "Compression" in its second choice. For the type of "Graphics", the classification rates are more evenly distributed (the highest one is less than 50%) than those in other types (Cavnar, 1994).

Similarly, Stamatatos introduces this technique to check intrinsic plagiarism. Specifically, he uses a sliding window moving along the whole text. Each time he compares the text in the window with the entire text. He also applies heuristic rules for this plagiarism detection. In his experiment, 3-grams are used, the window width was set to 1000 characters, and the step for a sliding window was set to 200 characters. He selected more than 3000 texts whose lengths varied from 3000 to 2.5 million characters. The result yielded a 78% accuracy at confirming no plagiarism on known plagiarism-free documents. The method, however, can be affected by the amount of plagiarism, so if there is too much plagiarism, it will change the writing style of the text (Stamatatos, 2009).

Abou-Assaleh et al. conducted a byte-based n-gram approach for detecting new malicious code. They generated n-gram signatures from collections of malicious code and benign code and then classified an unseen code against these signatures. They adapted the Common N-Gram (CNG) method and selected the L most frequent n-grams. Their classification criteria were based on the method of K-nearest neighbours (KNN). The data included 25 worms and 40 healthy programme files. By carefully configuring the relevant parameters during training, the training accuracy can be 95% and its 5-fold cross-validation average accuracy can reach 94% (Abou-Assaleh et al., 2004a).

Another byte-based n-gram approach is proposed by Frantzeskou et al. to determine the authorship of an unknown piece of source code. They first selected L most frequency n-grams to generate a profile set. Then they defined a similarity measure between two different profile sets by the size of the intersection of these two sets. They tested their method with 267 programmes in C++ by 6 different programmers. One half of the data was used as training data and the other half as testing data. The results show that the accuracy rate can reach 100% with collective n-grams. For the programmes written in Java, its accuracy is up to 97% (Frantzeskou et al., 2006).

Word-based n-gram methods are not as widely used as character/byte based techniques. Hovold employed a word-position-based n-gram technique with removal of frequent and infrequent words to filter spam emails via a Naive Bayes classifier. The n-grams used are 2-grams and 3-grams. The experiments were conducted on the PU corpora and SpamAssassin corpus; the PU corpus includes data from different users and is encrypted, while SpamAssassin data is unencrypted. The results show that after removing the most frequent words, the precision increases from 77% to 94% (Hugo Jair Escalante, 2011).

Stein et al. present a plagiarism detection method without a reference text. They gathered 3000 texts from the Gutenberg website, and divided each text into blocks of 5000 characters. The testing criterion was that if there were more than 50% of insertions of words between any two blocks, it is considered plagiarism occurred. The experimental results show that the overall accuracy can reach 90% (Stein et al., 2010).

Another study that used word-based n-gram analysis techniques is that of Shrestha and Solorio (2013). The researchers used three different word-based n-grams, namely stopword n-grams, n-grams with at least one named entity, and all words n-grams, to identify both verbatim plagiarism and obfuscated plagiarism. The stopwords used were the same as those used by Stamatatos (Ali et al., 2011). They showed that stopword n-grams and n-grams with named entities are very strict and, therefore, are more suitable for non-obfuscation detection. However, once the all words n-gram attributes were added in order to handle obfuscation cases, the precision decreased more than 20%. The precision returned to the previous level once proper constraints (e.g. at least $n+2$ consecutive detections) were introduced (Linguistic, author).

In order to break through the constraint of n-gram in fixed length, Houvardas and Stamatatos propose an n-gram method with variable-length to unify themes of text features. They propose a dominant n-gram to represent text features. It has the maximum 'glue' with the similar n-grams. The glue is measured by a Symmetrical Conditional Probability (SCP) of a feature set. First, they generated the L most frequent n-grams ($L=500$) for $n=3$, 4, 5 to form an initial feature set. Then this feature set was reduced by their proposed approach above. Finally they used the reduced feature sets to train a Support Vector Machine (SVM). They also proposed an improvement method by a pre-processing of removing the digits in the text. In their experiments, they use this SVM to classify 50 authors. They concluded that their method is at least as effective as the ones by selecting the most significant n-grams in terms of information gain. The recognition rate of their method was approximately 72% (Stein et al., 2010).

In order to obtain better detection performance, many researchers adopt hybrid n-gram techniques for their analysis. Masud et al. designed a hybrid model for detecting malicious programmes by combined profiles. These profiles include binary n-grams, assembly n-gram, and library function calls which are associated with binary executables, disassembled executables, and API calls. The binary n-grams profile code instructions and data and are formed from "hexdump" files. As some operations, such as collection, transfer, and search, need operating in these enormous data, they adopted more advanced techniques (Disk I/O) and algorithms (Adelson-Velsky-Landis tree) to improve the performance. They used two different data sets for their experiments, one including about 600 benign and 850 malicious executables, and the other consisting of 1500 benign and 1000 malicious executables. Their experiment showed that the classification accuracies are above 96% with false positives below 5.4 and false negatives below 3.6 for both datasets (Ullmann, 1977).

Barron-Cedeno et al. (Behavior-based modeling and its application to Email analysis.pdf) propose another word-based n-gram analysis method to improve detection speed and automatically detect shared contents in written documents for text reuse and plagiarisms. They designed a pre-processing procedure, which first gets rid of all non-alphabetic terms and replaces them with words of the same length (the maximum of length is 9). That is, their word-based n-grams are replaced by the character of digit-based ones. They calculate similarity estimation by these simple n-grams. They adopt two sets of data for their experiments: one from the PAN-PC-09 Corpus and the other from the Co-derivatives Corpus. The results show that their approach does not much affect the detection accuracy of the retrieval process. Their methods can significantly reduce the computational time and storage. Obviously, useful information is lost in this approach, which is not suitable to be used in the analysis of small datasets.

Bit-level or binary-based n-gram has been used for text processing for a long time, as it is simple and efficient, particularly when computer-related resources were much more limited than those today. A binary n-gram profile often keeps the presence

Table 1
Summary of related analysis methods on *n*-grams and forensic linguistics.

Methods	Study	Characteristics	Limitations
n-Gram analysis	Cavnar and Trenkle (Manoj Harpalani et al., 2011)	Character-based *n*-gram with most frequency	Formal language; only select most *n*-grams with higher frequency; Profiles are not normalized
	Abou-Assaleh et al. (2004a)	*L* most frequent byte *n*-gram signatures with most frequency	Only select *L* most freq *n*-grams; Formal language; Similar topics
	Hovold (2005)	Word-position-based attribute vectors	Remove most infrequent and most frequent words; Formal language; Similar topics
	Frantzeskou et al. (2006)	Byte-level *n*-gram and author profiles	Formal language; Similar topics
	Houvardas and Stamatatos (2006)	variable-length word *n*-gram with most frequency	Similar topics; Formal language
	Masud and Bhavani Thuraisingham (2007)	Binary/byte based *n*-gram (binary features; assembly features and function call features)	Formal language; Similar topics (same programming language)
	Stamatatos (2009)	Character *n*-gram normalized frequencies	Formal language; The plagiarized documents are automatically generated (not real environments).
	Stein et al. (2010)	Character and word based *n*-gram(lexical features, syntactic features, structural features)	Formal language (artificially plagiarized; documents); Similar topics
	Barrón-Cedeño et al., (2010)	Word-length encoded word based *n*-gram	Gain speed at cost of accuracy; Formal language (Wikipedia); Similar topics
	Shrestha and Solorio (2013)	Word based *n*-gram (stopword, named entity, all word)	Formal language; Similar topics
Forensic linguistics	Hirst and Feiguina (2007)	Syntactic features, lexical features	Formal language; Similar topics
	Iqbal et al. (2008)	Lexical, syntactical, structural and content-specific attributes	Formal language; Similar topics
	Afroz et al. (2012)	Lexical, syntactic, and content specific features	Most are formal language and only one of their 3 data set is from blogs; Similar topics

rather than the number of occurrences of each distinct *n*-gram in a string. Ullmann (1977) proposed a binary *n*-gram approach for correcting substitution, deletion, and insertion errors in words. The data are from about 2700-word dictionary of Riseman and Hanson (1974). The experiment results showed that the accuracy is 99.2% and its failure rate is 5.3%.

2.2. Forensic linguistics

Although the use of authorship attribution methods can be dated back to the 19th century, the phrase Forensic Linguistics was not used until 1968 when a linguist Jan Svartvik analysed statements for police officers at Notting Hill Police Station (https://en.wikipedia.org/wiki/Forensic_linguistics). Forensic linguistics applies the knowledge and methods of linguistics to the forensic context of law, the investigation of crimes, and so on. It often provides information for the questions of authorship, such as "Who wrote the text?", "Do these words mean some other thing?". Perhaps the most famous and earliest text analysis is related to Shakespeare's work which was authored several centuries ago (Frantzeskou et al., 2006).

Hirst and Feiguina (Clara Maria Colombini, 2012) propose an approach to extract stylometric features from text, such as lexical and syntactic features, and then calculate their bigram (2-gram) analysis for authorship authentication as a short text plagiarism detection method. Due to a lack of text in the study, they focused mainly on the syntactic structures of the short texts. They adopted a partial parsing method to process the short text and generate an intermediate language, which can keep most of the structural features without the complicated inversion operations that complete parsing requires. Afterwards, they applied the bigram analysis to the intermediate language to capture the syntactic features for the short text. Besides these syntactic features, they also made use of lexical features, such as vocabulary richness, and average word and sentence length. The classifier they used was support vector machine. They conducted several experiments according to different features (syntactic, lexical and all) and different block size of data. They collected texts from the Gutenberg site (about 250,000 words for each author), breaking each text into three lengths (200, 500, and 1000 words respectively). The results showed that while there are similar recognition rates for the larger block size (about 99% for all 3 types of features when the size is 1000); it does have a noticeable improvement when performing with the combination of features instead of individual ones (Hirst and Feiguina, 2007).

Emails are a common source of evidence in forensic analysis. Iqbal et al. (2008) used email 'write-prints' to determine authorship. The profiles they used for describing write-prints include morphology, syntax, and structure information in emails. Analysis approaches used are decision trees and support vector machines. They collected more than 200,000 email data from 158 employees of the Enron Company and filtered out high frequency words. Experimental results showed their method can achieve 80% accuracy when there are four suspects and 77% accuracy when there are 10 suspects. In follow-up research without training data (Iqbal et al., 2010), they extracted similar stylometric attributes for profiling authors. They used three clustering methods (Expectation Maximization, k-means clustering, and bisecting k-means clustering) to evaluate their method. Experimental results showed that the k-means method performed better when each cluster has fewer than 40 emails, while the bisecting k-means method was preferable when there is a large set of training data.

Afroz et al. (2012) presented a method to detect online hoaxes and frauds. They propose three types of characteristics, namely: (1) Writeprints set, which include lexical features (character-based features and word-based features), syntactic features, and content specific features which are the keywords for specific topics. (2) Lying feature set; and (3) Authorship-attribution features set, which includes nine features (number of unique words, Gunning-Fog readability index, average syllables per word, etc.). They used support vector machine with sequential minimal optimization to analyse the first feature set, and use a decision tree algorithm (J48) for the later 2 features. Their first approach achieved an overall detection rate of 96% and the second one was above 84%.

2.3. Summary

n-Gram analysis and forensic linguistic are widely used in text analysis. *n*-Gram analysis techniques are generally used to obtain simpler text features, such as lexical and syntactic. Forensic linguistic, on the other hand, tends to adopt more complex attributes such as syntactic, structural, and content. Most methods proposed so far are used to analyse natural language with similar topics, which are assumed to be less complex than text generated during real-time conversations (see Table 1).

While it is known that authorship attribution with higher-level complex features can generally obtain better detection performance (Hirst and Feiguina, 2007; Iqbal et al., 2008; Afroz et al., 2012), the approaches using simpler features, such as the *n*-gram method, are still widely used recently due to their higher efficiency and they can also be as accurate with appropriate tuning.

Although the *n*-gram methods outlined in Table 1 are putatively binary-based, to some extent these methods can be regarded as byte-based approaches, as their every step moved by 8 bits rather than by 1 bit. In contrast, the method proposed in this paper uses a binary *n*-gram method where the sliding window of n bits width moves a single bit with each step. Also, compared to most *n*-gram approaches (such as those described in this paper) which remove the most frequent and or most infrequent *n*-grams, the method here takes all *n*-grams into account for the analysis as their infrequency is also a part of the writing style characterization.

In summary, the approaches discussed above differ from the method proposed in this paper as follows: firstly, the test data in this paper include “informal” language that may not appear in dictionaries (e.g. the use of emoticons and abbreviations); secondly, the data is not topic-restricted; and thirdly, the analysis method is a true bit-level *n*-gram analysis.

3. Research methodology

3.1. Data collection

3.1.1. System architecture

The posting attribution system described here implements the binary *n*-gram profiling method. It includes four main components: (1) data collection, which is responsible for collecting relevant data from an external source; (2) data processing, which includes data filtering and profile selection; (3) profile classification, the core part of the system, performed by related classification algorithms; and (4) post processing, which is associated with exception handling and updating of history records (Fig. 1). The system components and/or objects are detailed as follows:

- The Posts are the messages posted by authors on social media.
- The Collector collects all data from the websites.
- The Pre-processor performs some primary processing, such as words filtering and posts concatenating, to facilitate later analysis and classification.
- The Profile Extractor extracts author's profile information from his/her posts.
- The Profile Vector Composer forms an author's profile vectors based on his/her multiple profiles.
- The Profile Depository stores authors’ profile vectors during the training stage.
- The Profile Classifier conducts analysis, classification and clustering to determine whether the current author is already in the depository.
- The Profile Updater updates the current author's profile vector in the Profile Depository according to relevant rules.
- The Exception Handler deals with the situation when the anomaly occurs, such as stopping the current author's connection, or just sending an alert to the website administrator.

Generally, there are two stages in the attribution system (as with most anomaly detection systems):

1. Training stage. In this stage, users’ writing styles are built and all styles are considered to be genuine.
2. Classification stage. In this stage, the current author's writing characteristics are compared with those created in the training stage and check to determine whether it satisfies the pre-defined threshold of metrics. If it does not, it means the current author is a new person, and the system triggers an alert for advanced handling, which may include training data updating.

3.1.2. Collecting data

On some social media, a user needs to log in to the system before posting or viewing comments. Once the user is authenticated, for example using the user name and password, the user is free to post comments on any topics of interest. As we analyse the authors’ writing styles, we need to collect the user name and their comments. Generally, each comment is relatively short (e.g. several dozens of words). As a single short comment is not enough to accurately characterise a user’s writing styles, we accumulate each author's comments to form a larger text.

We designed and implemented a tool in C# for capturing posts and other data from designated websites checking them hourly to determine whether there are new posts on the websites. If it finds

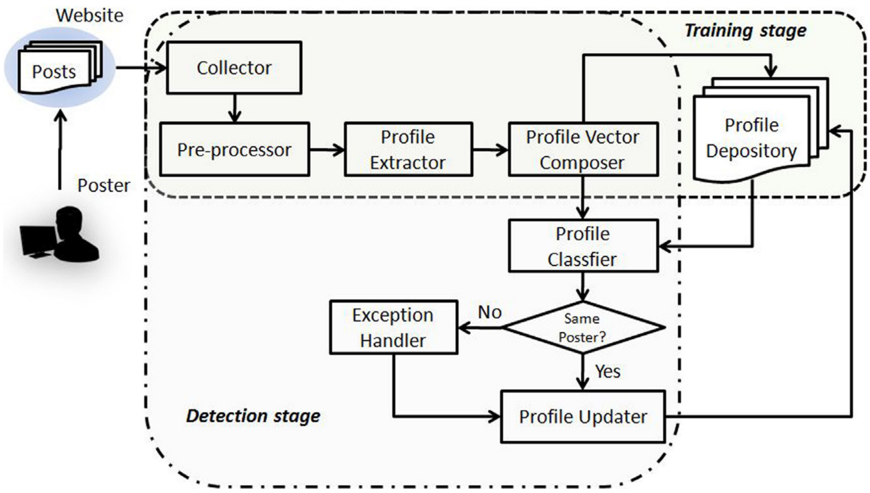

Fig. 1. System architecture.

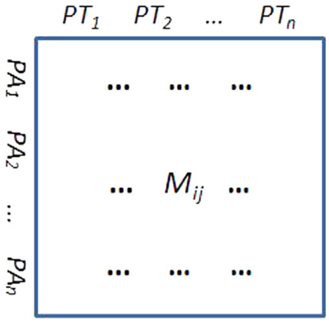

Fig. 2. Attribution matrix.

any, it automatically downloads them to the local host for the Pre-processor. The data collected includes website name and address, topics, user name, and post contents (text).

3.2. *Data analysis*

This section firstly briefly introduces some basic analysis techniques used with the *n*-gram analysis method to create the posting attribution system. Then, it details the data pre-processing. We then explain how the *n*-gram profiles are extracted individually and collectively from a text, and how these *n*-gram profiles are used for our author attribution.

3.2.1. *Basic analysis*

3.2.1.1. Euclidean distance. The Euclidean distance is a commonly-used method to measure the similarity/distance between two points in a Euclidean space. For two points x and y in an m-dimensional space, where $x=(x_1, x_2,\ldots, x_m)$ and$y=(y_1, y_2,\ldots, y_m)$, the Euclidean distance is defined as follows:

$$d_{L2}(x, y)=\sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

When assessing the similarity between any two author profiles, we use the Euclidean distance as a measurement of similarity. If the distance between any two authors is sufficiently small, we may conclude they are the same author. This Euclidean distance is used in the analysis methods of the KNN and the outlier classification.

In the context of this work, the two profiles being compared at any time are the stored profile already attributed to a user through a training phase, and a candidate profile which is being tested for eligibility to be attributed to a known author.

3.2.1.2. K-nearest neighbours algorithm (KNN). The K-Nearest Neighbours algorithm is a non-parametric analysis method often used for classification and clustering analyses in image processing, pattern recognition, data mining, etc. If $k=1$, it means to find the minimum distance between concerned points. Here, we use the Euclidean distance above as a metric of the distance. In this context, the KNN algorithm groups together authors whose profiles are sufficiently similar to potentially be the same author. In particular, for $k=1$, it finds the single author whose profile is closest to the first author.

3.2.1.3. Outlier classification. An outlier is something that lies outside of the main body or group and may be perceived to not belong to that group. In this context, an outlier of'similarity' will be an author's profile that appears to be sufficiently similar to one or more other authors' profiles and considered as the same author. We use the Interquartile Range (IQR) to determine outliers. IQR is defined as "the difference between the upper and lower quartiles" (Graham Upton, 1996). A commonly-used method for outlier identification is defined as those values which are 1.5 times IQR lower than the first quartile or 1.5 times IQR higher lower than the third quartile. As we are seeking similarity, we only focus on the lower-end outliers which are those closest to the target profile. Based on the analysis of our training results, we set the value 0.8 times IQR instead of 1.5 for a better performance.

3.2.1.4. Detection performance. We use accuracy, false positive (FP) and false negative (FN) as the detection performance metrics. Accuracy includes both true positive and true negative; in other words, that the author is correctly attributed and another person will not be wrongly attributed as the author. A false positive implies that another person is incorrectly attributed as the author, and a false negative implies that the true author is incorrectly rejected as the author.

As stated before, an author's text is divided into halves, and one is for training and the other is for attribution. For a specific text by author i, we assume the profile $\mathrm{PT_i}$, $(i = 1, \ldots, n)$ is obtained from the part of the text during the training stage and the PA_i, $(j = 1, \ldots, n)$ is obtained from the other part of text in the attribution stage. Hence, the attribution process is to match the profile $\mathrm{PA_j}$ against every PT_i by the KNN and outlier methods. In the case of KNN, the PA_j is attributed to PT_i if the distance between is the smallest in the case of KNN, $k=1$ or smaller than a pre-defined (obtained during training stage) in the case of the outlier analysis approach. If there are n profiles that need attributing, we can easily have an attribution matrix with size of $n \times n$ (Fig. 2). It is noted that there can be more than one matches in each of two cases as there can be several ones with the same smallest values or smaller than the preconfigured value.

Then we have the following algorithm to calculate the True Positive (TP), the True Negative (TN), the False Positive (FP), and the False Negative (FN).

```
Scan each row of the matrix {
  if ($M_{ij}$ is selected) {
    if (i=j) // correctly attributed
      tp++;
    else
      fp++;
  }
  else {
    if (i=j)
      fn++
    else
      tn++;
  }
}
```

Once we obtain the TP, TN, FP, and FN from above algorithm, we define performance metrics by the following standard formulae (https://en.wikipedia.org/wiki/Sensitivity_and_specificity):

a) True Positive Rate: $TPR=\frac{TP}{TP+FN}$

b) False Positive Rate: $FPR=\frac{FP}{FP+TN}$

c) False Negative Rate: $FNR=\frac{FN}{FN+TP}$

d) Accuracy Rate: $AR=\frac{TP+TN}{(FN+TP)+(FP+TN)}$

3.2.2. *Pre-processing*

Because most website postings are relatively short, individually these posts do not contain sufficient text for meaningful statistical

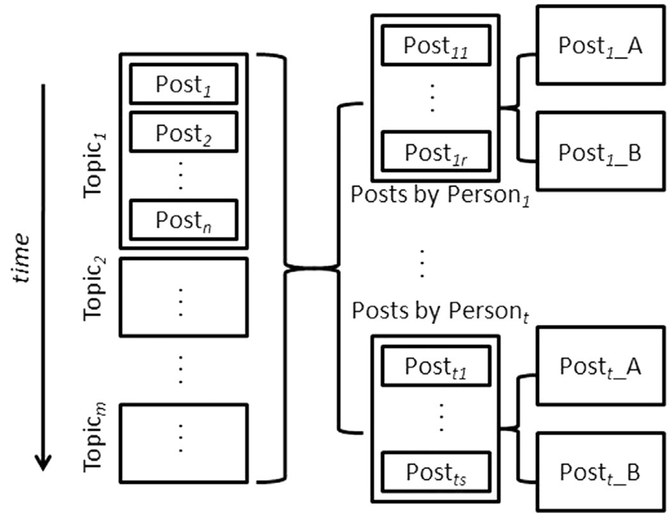

Fig. 3. Pre-processing.

analysis. Therefore, downloaded posts are accumulated for each author by chronologically concatenating the posts by topics for each author (based on the username). These concatenated posts are then used for analysis. In the following experiment, each author's accumulated text is segmented in half (at the end of a sentence) to form two blocks of text ($Post_i\_A$) and ($Post_i\_B$). One is for the training data and the other for classification (Fig. 3).

3.2.3. n-Gram profiles

An *n*-gram is a sequence of *n* adjacent units. The units range from bits, bytes and characters to whole words or terms. *n*-Grams are extracted from the text by taking a current 'window' which contains the n adjacent units, then sliding the window along to take the next n adjacent units. The windows may or may not overlap with each other. Changing the value of *n* gives different window sizes, e.g. setting $n=1$ gives a unigram, while $n=2$ and $n=3$ yield bigrams and trigrams.

For example, suppose we have a text to analyse with word-based *n*-grams: "I am in the UK":

The unigrams (1-grams): "I", "am", "in", "the" and "UK".
The bigrams (2-grams): "I am", "am in", "in the" and "the UK".
The trigrams (3-grams): "I am in", "am in the" and "in the UK".
The 4-grams (4-gram): "I am in the" and "I am in the UK".

The above is an example of the word-based *n*-gram model, since the units are whole words. By contrast to the bit-, byte- and character-based *n*-grams, the byte length of the window for the word-based *n*-gram method is flexible as the lengths of words are not fixed although the number of words per *n*-gram is fixed.

Beside the word-based *n*-gram models above, there are other types of *n*-gram models either, for example, character-based model, byte-based model, and bit-based model. Word-based *n*-gram models can better present author's writing styles, but as word lengths are variable and cannot be fixed, their analysis algorithms are complex and need more time and space for processing. In contrast, the other *n*-gram models with fixed lengths are more effective. In some situations, the character-based models are the same as the byte-based models as long as one character is represented by a single byte. This applies to the languages based on Latin alphabet characters. However, they are different while processing some Asian languages, such as Chinese, Japanese, and Korean, as they generally use two bytes to represent a single character. For example, if we employ the byte-based approach, it sometimes splits words into half for analysis. In English, a word would be split into two words in the case of word-level model, e.g. the word "friend" is divided into the words "fri" and "end". When we put a text into its 7-bit ASCII binary representation, the text becomes a string of "0"s and "1"s. The bit-based *n*-gram analysis is based on this type of binary string. Due to its fixed unit length (1 bit) and small number of possible values (2 cases), it is extremely efficient for text analysis.

We perform binary *n*-gram analysis over the text by viewing every character with its ASCII code. For example, if the text string is "ab", its ASCII binary string is "11000011100010". Its 3-grams are shown in the Table 2

Table 2 3-gram of binary string of "ab".

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

In the same way, we can obtain an *n*-gram profile for a given text, which can be expressed in a vector $(g_0, g_1,\ldots,g_{2^n})$ in a 2^n-dimensional space where each $\mathrm{g_i}$ represents the number of occurrences of each binary string of length n in the text. This vector is used as a profile in our authorship analysis (see Fig. 4).

3.2.4. Collective profiles

As the value n and the number of authors become larger, the computational time and storage space show exponential growth. Therefore, we need to find a way to mitigate these problems. Here we use the weighted average of r consecutive *n*-grams to profile a text. Suppose we have the *r n* -gram profiles and starting from *m*: $N_m, N_{(m-1)}, \ldots, N_{(m-r+1)}$ as follows:

$$N_m=(g_{m1}, g_{m2},\cdots,g_{m2^m}),$$

$$N_{(m-1)}=(g_{(m-1)1}, g_{(m-1)2},\cdots,g_{(m-1)2^{(m-1)}}),$$

$$N_{(m-r+1)}=(g_{(m-r+1)1}, g_{(m-r+1)2},\cdots,g_{(m-r+1)2^{(m-r+1)}}),$$

Assuming that larger *n*-gram profiles include more of the author's writing styles than in smaller *n*-gram profiles, we put more weight on larger *n*-gram profiles for the collective *n*-gram profiles and keep the sum of all weight to be 1. For the above case, we assign a weight to each profile as follows:

- For N_m, its weight: $W_m=\frac{2^m}{(2^m+2^{(m-1)}+,\ldots,+2^{(m-r+1)})}$;
- For $N_{(m-1)}$, its weight: $W_{(m-1)}=\frac{2^{(m-1)}}{(2^m+2^{(m-1)}+,\ldots,+2^{(m-r+1)})}$;
- …,
- For $N_{(m-r+1)}$, its weight: $W_{(m-r+1)}=\frac{2^{(m-r+1)}}{(2^m+2^{(m-1)}+,\ldots,+2^{(m-r+1)})}$.

Here, we have

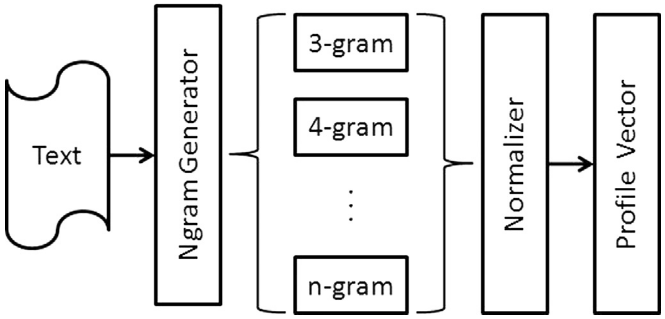

Fig. 4. Profile extraction.

$W_m+W_{(m-1)}+\ldots+W_{(m-r+1)}=1$

Then, we have the following collective n-gram profiles $N_{(m,r)}$ as follows, which starts from N_m with length of r.

$N_{(m,r)}=(W_m N_m,\ W_{(m-1)}N_{(m-1)},\ldots,W_{(m-r+1)}N_{(m-r+1)})$

Here, we can replace either N_m or $N_{(m+1)}$ with $N_{(m,r)}$. In the first case, we expect a better detection performance by using a more complex collective profile for the single n-gram profile. In the other case, we expect a speed improvement by introducing a lower order of collective n-gram profiles:

- Speed priority: as the name suggests, the goal is to speed up the classification. Intuitively, we use lower-order collective n-grams to replace the more time-consuming high-order n-grams. For example, for a given n-gram where $n=\text{k}$, we collectively use r consecutive lower-order n-grams to represent the n-grams, e.g. (($k-1$)-gram, ($k-2$)-gram, …, ($k-r$)-gram).
  Assuming it takes T time for processing ($k-r$)-gram, then we roughly have
- $2T$ for ($k-r+1$)-grams,
- …,
- $2^{(r-1)}T$ for ($k-1$)-grams, and
- 2^rT for k-grams.
  The total time spent for these r consecutive grams is

  $(2^{(r-1)}T+\ldots+2^1T+2^0T)$

  Savings in time is

  $2^rT-(2^{(r-1)}T+\ldots+2^1T+2^0T)=T.$

  Therefore, the speed gain we can achieve is $\frac{1}{2^r}$. If $r=3$ in our case, then the classification can run 12.5% (1/8) faster.
- Accuracy priority: here we focus on maximizing the classification accuracy. Our method is that, beside this n-gram, we consider the previous ($r-1$) consecutive lower-order n-grams (r n-grams in total). We use (($k-1$)-gram, ($k-2$)-gram, …, ($k-r$)-gram), k-gram) to represent k-gram. The cost for it is that more time is required to calculate the extra data (previous consecutive ($k-1$) lower-order grams). We calculate the cost to be $\frac{2^{(r-1)}-1}{2^{(r-1)}}$. As the r increases, the cost is nearly double. Therefore, unless necessary, it is not recommended to choose very large r for this method to improve accuracy. However, it is a very effective for smaller values of r, for example, if $r=2$ and there is only a 50% increase.

3.2.5. *n-Gram analysis*

In the experiments, we calculate n-grams from $n=3$ to 15. As the length of the available text varies for different people, the first step is to normalize these n-grams by converting them from absolute numbers to a proportional value using a normalization procedure. The normalized n-grams then are converted into a profile vector $(N_3, N_4,\ldots,N_{15})$, where each N_i is another 2^i-dimensional n-gram vector $N_i=(g_0, g_1,\ldots,g_{2^i})$ (see Fig. 4).

We now describe what a typical profile vector looks like, including the range of possible values, whose sum of the'heights' of each number in the vector should be 100%.

Table 3
Summary of the raw data.

Topic	Posters	Comments	Period
475	8587	94,274	2014/06/10–2014/11/17

Table 4
Results for KNN classification.

Single n-gram profiles	No. of matches	Accuracy (%)	FN (%)	FP (%)
N_3	17	42.50	15.33	57.50
N_4	31	77.50	6.00	22.50
N_5	33	82.50	4.67	17.50
N_6	35	87.50	3.33	12.50
N_7	39	97.50	0.67	2.50
N_8	38	95.00	1.33	5.00
N_9	38	95.00	1.33	5.00
N_{10}	37	92.50	2.00	7.50
N_{11}	40	100.00	0.00	0.00
N_{12}	40	100.00	0.00	0.00
N_{13}	40	100.00	0.00	0.00
N_{14}	40	100.00	0.00	0.00
N_{15}	40	100.00	0.00	0.00

4. Experiment findings

The raw data of posts are obtained from the website http://drum.abc.net.au by a purpose-designed tool and shown in Table 3.

We selected the 40 authors who generated the most posts during this period. Then, as stated in the Section 3.2, all posts were concatenated into a single text for each author, and then the text was equally divided into two parts, one to be used for training and the other for classification.

We conducted the following experiments.

4.1. *Experiment 1: KNN classification*

In this experiment, the assumption is that the author to be attributed is someone in the sample set. Therefore, the question is “Who in the sample set is the current author?”. Here, we use the KNN ($k=1$) analysis technique to determine the similarity. For each set of author data in the classification set, we calculated the n-gram profile for each of $n=3$–15. This collection of n-gram profiles was compared with the same profiles for all authors in the training set, and the attributed author chosen was the one with the smallest deviation from the current classification author data, using an average of differences to calculate the deviation.

The results are shown in condensed form in Table 4. The results range from the weakest accuracy with the lowest value of n set to 3, through to credible accuracy with $n=6$ where 35 of the 40 author data from the classification set were correctly matched (giving accuracy of 87.50%, and FN and FP of 3.33% and 12.50%, respectively) and concluding with fully accurate matching for $n=11$ and above.

4.2. *Experiment 2: outlier classification*

The KNN method assumes that the author data to be classified can be attributed to a known author in the training set. This may not always be the case, but the KNN method would nevertheless attribute the authorship to the nearest training set author regardless. In this experiment, we seek to firstly determine whether the current author is someone in the training set, and if and only if so, to then attribute the author.

We need to predefine a threshold for confirming the whether the author data to be classified is indeed in the training set of authors, and a commonly-used method for defining a threshold is outlier analysis. If the distance between the current author and any other in the sample set is smaller than the predefined value, then we assume that the author data to be classified is in the training set and that the nearest neighbour from amongst those below the threshold is the corresponding author in the training set.

We use the IQR method to determine outliers for this problem. Here, we only consider the outliers at the lower side. If such outliers exist, it means there are extremely close distances between the author data to be classified and author data in the training set, suggesting they are probably the same author. In our experiments, we find the coefficient $k=1.5$ is much stricter, and we took a less conservative view and reduced it to $k=0.8$. The results are shown in Table 5.

Table 5
Results for outlier classification in percentages.

Single *n*-gram profiles	No. of matches	Accuracy (%)	FN (%)	FP (%)
N_3	13	32.50	11.33	67.50
N_4	24	60.00	7.33	40.00
N_5	32	80.00	8.67	20.00
N_6	28	70.00	4.67	30.00
N_7	24	60.00	5.33	40.00
N_8	28	70.00	0.00	30.00
N_9	38	95.00	5.33	5.00
N_{10}	35	87.50	1.33	12.50
N_{11}	35	87.50	1.33	12.50
N_{12}	38	95.00	12.00	5.00
N_{13}	40	100.00	12.67	0.00
N_{14}	40	100.00	14.00	0.00
N_{15}	39	97.50	15.33	2.50

4.3. Experiment 3: collective classification

For larger values of n, the computational time and storage space required for classification grow. In order to alleviate these problems, we can use collective lower-order grams to replace the higher-order grams, because it is less time and space consuming to process lower-order grams. In our experiment, we use three consecutive lower-order grams for the current order grams. For example, for faster classification we can replace 7-gram with collective *n*-grams of 4-, 5- and 6-grams, and for higher accuracy we can substitute 7-gram by *n*-grams of 5-, 6- and 7-grams (more details are discussed in Section 3.2). The results are shown in Table 6.

We conducted several experiments with higher *n*-grams for $n=16.21$ and verify that the findings for the higher *n*-grams are the same (Table 7).

Table 7
Results for KNN classification for $n=16$–21.

Single *n*-gram profiles	No. of matches	Accuracy (%)	FN (%)	FP (%)
N_{16}	40	100.00	0.00	0.00
N_{17}	40	100.00	0.00	0.00
N_{18}	40	100.00	0.00	0.00
N_{19}	40	100.00	0.00	0.00
N_{20}	40	100.00	0.00	0.00
N_{21}	40	100.00	0.00	0.00

5. Discussion

In this section, we discuss the results of the experiments, and then outline some limitations in the collected data that will be addressed in future work.

5.1. Interpreting the results

We evaluate the performance of the classification methods firstly by accuracy, followed by efficiency in terms of time consumption.

1) The KNN classification results in Table 4 show that the approach is effective when $n=7$ (one byte in ASCII), as the accuracy is up to 97%. The accuracy then increases to 100% when *n* is larger than 9 (one bit more than 1 byte). Generally, the accuracy increases and FN decreases as *n* becomes larger, except for $n=10$ where there is a noticeable decrease to 92%. This may be an artefact of the data, and further data is being collected from alternative sites for comparison (see Section 5.2).
2) As we had expected in the outlier classification approach, the average classification rate decreases from 90% in the KNN method to 85.7%, and both FN and FP increase by 4.6% and 4.3% respectively. For the 9-gram, the accuracy is significantly higher than 10-gram and 11-gram, and even equal to that of the 12-gram analysis. One explanation might be that it captures much more writing styles due to the shorter length of the “informal” language. This shows that the discrepancy of “similarity” becomes much smaller. A strange phenomenon associated with the FN is that it is high at the beginning (3-gram) and decreases gradually to the middle and then goes back up at the end (15-gram). This might be explained by the characteristics of language. As stated before, the text can be regarded as a stream of characters. In a similar way, every character is composed of a fixed length (7bits) of binary string in our case. Whenever the value of n is around the multiples of 7, the FN reach one of its valleys or minimums, and if n is around the middle between any these two consecutive multiples, the FN becomes one of its peaks.
3) When we apply collective *n*-grams to our data, there are two modes for classification:
   a) For KNN classification
   When we apply the KNN for speed-priority classification, there is a slight performance improvement with 0.23% of accuracy increase and 0.06% of decreases of FN at the cost of a 1.5% accuracy loss and 0.4% of FN up. In other words, there is

Table 6
Results for KNN and outlier classification.

Collective profiles		$N_{(5,3)}$	$N_{(6,3)}$	$N_{(7,3)}$	$N_{(8,3)}$	$N_{(9,3)}$	$N_{(10,3)}$	$N_{(11,3)}$	$N_{(12,3)}$	$N_{(13,3)}$	$N_{(14,3)}$	$N_{(15,3)}$
KNN	No of matches	33	37	38	38	38	38	39	40	40	40	40
	Accuracy (%)	82.50	92.50	95.00	95.00	95.00	95.00	97.50	100.00	100.00	100.00	100.00
	FN (%)	4.67	2.00	1.33	1.33	1.33	1.33	0.67	0	0	0	0
	FP (%)	17.50	7.50	5.00	5.00	5.00	5.00	2.50	0	0	0	0
Outlier	No of matches	32	32	28	32	37	36	36	38	40	40	40
	Accuracy (%)	80.00	80.00	70.00	80.00	92.50	90.00	90.00	95.00	100.00	100.00	100.00
	FN (%)	9.33	6.67	7.33	4.00	5.33	2.67	2.67	7.33	8.67	15.33	17.33
	FP (%)	20.00	20.00	30.00	20.00	7.50	10.00	10.00	5.00	0	0	0

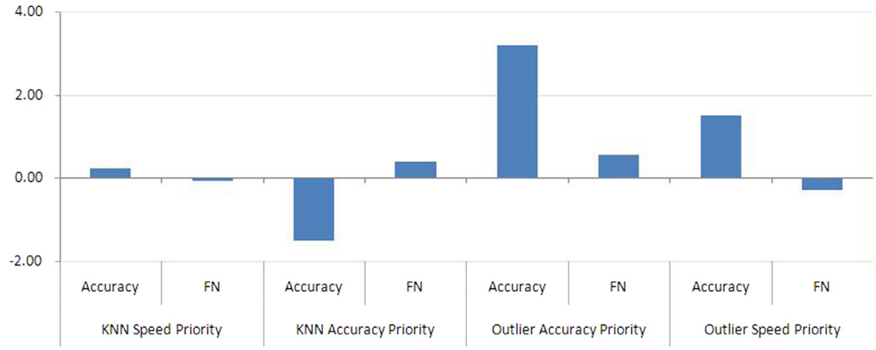

Fig. 5. Average differences of accuracy and FN between collective *n*-grams and individual *n*-grams.

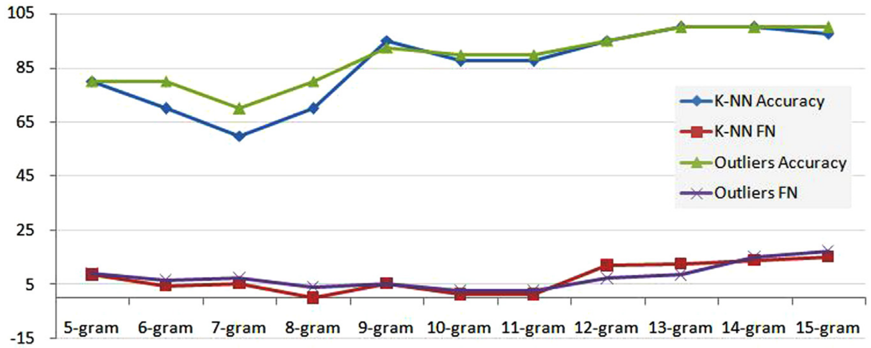

Fig. 6. Comparisons of accuracy and FN between KNN and outlier methods.

not a noticeable improvement for both classification cases (Fig. 5). This may imply that the collective approach is not very effective for improving the KNN algorithm.

b) For outlier classification
Although the collective method does not improve the performance for the KNN algorithm, it contributes to the classification of outliers. Inaccuracy-priority classification, we gain an average accuracy by 3.2% at the cost of mere 0.5% up of FN (Fig. 5). Even for speed-priority classification, we achieve an increased accuracy of 1.5% with a decrease of 0.3% FN (Fig. 5). These show that the collective method is an effective way to improve the outlier classification technique.

The collective method can increase the accuracy, as well as smoothing the results of both accuracy and FN (Fig. 6). This feature is important, especially for systems that require stability and robustness.

5.2. Limitations

The experimentation above gives a clear indication that posted text in public fora can be assessed to determine whether any two sets of text appear to be written by the same or a different author. However, a limitation of the work is that the collected data cannot enable validation of the results against known authorship. The postings collected were *bona fide* contributions made on a public site where posters' true identities cannot be confirmed from the public data. As such, it is not possible to be certain that the results above are entirely accurate, although the different methods provide mutual support. We addressed this by using equal quantities of each author's data as training set and analysis set, even if this does then assume that the author is the same individual throughout the duration of the data collection. This cannot be confirmed either, although it seems unlikely that this would be common enough to compromise the results for all 40 subjects.

A second limitation is that the pattern of generally-increasing accuracy as n increases punctuated by apparent weakness around n=10 may only be an artefact of this forum site, or even of the 40 authors.

To address these limitations, future work will be extending the original experimentation in two ways:

- Firstly we will collect a new set of data from a different site with two additional features:
  1. the full set of postings for each author is readily-available by viewing the author's personal profile, and;
  2. the metadata for each author is available.
  This means that instead of restricting the data collected to the specific period of collection, we are able to collect the entire history of postings from the author. We can also inspect the explicit metadata, including date of joining, “up” votes, and times, dates, thread and full text of posts. It is also possible to derive other features such as frequency of posting, posting versus date/time patterns, average post length, and so on. Analysing the metadata will permit cross-checking of results between metadata-based analyses and the *n*-gram linguistic analysis above. Also, with significant quantities of text, it will be feasible to perform more traditional linguistic analysis such as stylometrics to further confirm the findings.

- Secondly, we are in the process of planting control data in this new site, with known individuals posting under two author names. The relationships between these two'authors' with each other as well as with all other authors are thus known and this will provide some confirmation of the methods' accuracy.

The data set used in this paper was collected from http://www.abc.net.au/news/thedrum/, a news analysis and opinion site which frequently enables discussion fora within each article. The new data will be collected from http://www.theguardian.com/ which publishes similar material with the same discussion forum opportunities, but with the full user history and metadata as described above.

6. Conclusion and future work

In this paper, we proposed a bit-level based *n*-gram analysis approach to attribute users by their posts on social media. First, the authors' linguistic features were extracted from the text of their postings and accumulated into profiles using *n*-gram techniques. Then, these profiles were transferred into profile vectors. The KNN algorithm and IQR range techniques were employed as classifiers to determine whether any anomaly exists between two authors, i.e. whether they were different authors or not. We found that the KNN algorithm with a single *n*-gram is very effective for classification purposes. However, there is no noticeable improvement when this algorithm is associated with the collective *n*-gram data.

When we applied the outlier (IQR) classification method to our single *n*-gram data, the performance decreases in comparison to the findings using the KNN approach. However, when IQR is applied to the collective *n*-gram data, there is a significant performance enhancement.

In spite of the limitations outlined in Section 5.2, it appears that the experimental results do yield accurate results about authorship in public discussion fora. We had originally been motivated to discover whether there were any astroturfers present in the forum but there is little to indicate that there are any amongst the assessed authors, rather the analyses indicate that the posters are distinct individuals. It seems plausible that the selected data was not extensive enough to capture any astroturfer activity, and it is possible that astroturfers will not generate such extensive text for each account they operate (due to time limitations) so the capture of extensive new data is clearly indicated.

Hence, future work will include collecting more extensive data for analysis for analysis as described in Section 5.2. There is also scope for calculating higher order *n*-grams to determine if there are trends that are not evident from the current analysis.

One further line of investigation in this research is the scope for using it over languages other than English. We conjecture that for languages similar to English, such as European languages, that the outcomes would be similar. For other languages, in particular Asian languages, we would expect that larger values of n would be needed due to the more semantically-rich character set. So while in principle the *n*-gram analysis methods would be expected to work, the parameters for their success would likely be different.

Acknowledgements

The authors thank the editor and the three anonymous reviewers for their constructive and generous feedback.

References

Abou-Assaleh, Tony, Cercone, Nick, Keselj, Vlado, Sweidan, Ray, 2004a. Detection of new malicious code using *n*-grams signatures. In: Proceedings of Second Annual Conference on Privacy, Security and Trust, pp. 193–196.

Abou-Assaleh, T., et al., 2004b. *n*-Gram-based detection of new malicious code. In: Proceedings of the 28th Annual International Computer Software and Application Conference, Workshop and Fast Abstracts, pp. 41–42.

Abou-Assaleh, T., Cercone, N., Keselj, V., Sweidan, R., 2004c. Detection of New Malicious Code Using *n*-Grams Signatures.

Ali, A.M.E.T., Abdulla, H.M.D., Snasel, V., 2011. Survey of plagiarism detection methods. In: Proceedings of the 5th Asia Modelling Symposium (AMS'05), pp. 39–42.

Afroz, S., Brennan, M., Greenstadt, R., 2012. Detecting hoaxes, frauds, and deception in writing style online. In: Proceedings of the 2012 IEEE Symposium on Security and Privacy (Sp), pp. 461–475.

⟨Behavior-based modeling and its application to Email analysis.pdf⟩.

Barrón-Cedeño, A., Basile, C., Degli Esposti, M., Rosso, P., 2010. Word length *n*-grams for text re-use detection. Comput. Linguist. Intell. Text Process., 687–699.

Bailey, K.O., Okolica, J.S., Peterson, G.L., 2014. User identification and authentication using multi-modal behavioral biometrics. Comput. Secur. 43, 77–89.

Bergsma, S., Post, M., Yarowsky, D., 2012. Stylometric Analysis of Scientific Articles.

Cavnar, W.B., Trenkle, J.M., 1994. *n*-Gram-based text categorization. In: Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval, pp. 161–175.

Cesare, S., Xiang, Y., Zhou, W., 2014. Control flow-based malware variant detection. IEEE Trans. Dependable Secur. Comput. 11 (4), 307–317.

Chen, C.M., et al., 2013. Battling the internet water army: detection of hidden paid posters. In: ASONAM'13 Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, pp. 116–120.

Colombini, C.M., Colella, A.,Mattiucci, M., Castiglione, A., 2012. Network profiling-Content analysis of users behavior in digital communication channel. CD-ARES 2012, pp. 416–429.

Colombini, C.M., et al., 2012. The Digital Profiling Techniques Applied to the Analysis of a GPS Navigation Device. IMIS 2012, pp. 591–596.

Crawford, H., Renaud, K., Storer, T., 2013. A framework for continuous, transparent mobile device authentication. Comput. Secur. 39, 127–136.

Escalante, H.J., Solorio, T., 2011. Local histograms of character *n*-grams for authorship attribution. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, Portland, Oregon, pp. 288–298.

Frantzeskou, G., et al., 2006. Effective identification of source code authors using byte-level information. In: ICSE '06 Proceedings of the 28th International Conference on Software Engineering, pp. 893–896.

⟨https://en.wikipedia.org/wiki/Forensic_linguistics⟩.

⟨https://en.wikipedia.org/wiki/Sensitivity_and_specificity⟩.

Hirst, G., Feiguina, O., 2007. Bigrams of syntactic labels for authorship discrimination of short texts. Lit. Linguist. Comput. 22 (4), 405–417.

Harpalani, M., Hart, M., Singh, S., Johnson, R., Choi, Y., 2011. Language of Vandalism: Improving Wikipedia Vandalism Detection via Stylometric Analysis.

Holmes, D.I., 1994. Authorship attribution. Comput. Humanit. 28 (2), 87–106.

Houvardas, J., Stamatatos, E., 2006. *n*-Gram feature selection for authorship identification. Artif. Intell.: Methodol. Syst. Appl. Proc. 4183, 77–86.

Hovold, J., 2005. Naive Bayes spam filtering using word-position-based attributes. In: Proceedings of the Second Conference on Email and Anti-spam, CEAS, Stanford University.

Huang, L., Stamp, M., 2011. Masquerade detection using profile hidden Markov models. Comput. Secur. 30 (8), 732–747.

Huang, X., Xiang, Y., Bertino, E., Zhou, J., Xu, L., 2014. Robust multi-factor authentication for fragile communications. IEEE Trans. Dependable Secur. Comput. 11 (6), 568–581.

Iqbal, F., et al., 2008. A novel approach of mining write-prints for authorship attribution in e-mail forensics. Digit. Investig. 5, S42–S51.

Iqbal, F., et al., 2010. Mining writeprints from anonymous e-mails for forensic investigation. Digit. Investig. 7 (1–2), 56–64.

Keselj, V., et al., 2003. *n*-Gram-based author profiles for authorship attribution. In: Proceedings of the Pacific Association for Computational Linguistics, pp. 255–264.

Koppel, M., Schler, J., Argamon, S., 2009. Computational methods in authorship attribution. J. Am. Soc. Inf. Sci. Technol. 60 (1), 9–26.

Kapardis, A., Krambia-Kpardis, M., 2004. Enhancing fraud prevention and detection by profiling fraud offenders. Crim. Behav. Ment. Health, 189–201.

Li, W.-J., Wang, K., Stolfo, S.J., Herzog, B., 2015. Fileprints: identifying file types by *n*-gram analysis. In: Proceedings of the 2005 IEEE Workshop on Information Assurance and Security, pp. 64–71.

⟨Linguistic profiling for author recognition and verification.pdf⟩.

Martin, S., Sewani, A., Nelson, B., Chen, K., Joseph, A.D., 2005. Analyzing behaviorial features for email classification. In: Prodeedings of the IEEE Second Conference on Email and Anti-Spam (CEAS 2005).

⟨US Military to Replace Passwords with_Cognitive Fingerprints_-Infosecurity Magazine.pdf⟩.

Maurer, H., Kappe, F., Zaka, B., 2006. Plagiarism – a survey. J. Univers. Comput. Sci. 12 (8), 1050–1084.

Masud, M.M., Khan, L., Thuraisingham, B., 2007. A Hybrid Model to Detect Malicious Executables.

Mazhelis, O., Puuronen, S., 2007. A framework for behavior-based detection of user substitution in a mobile context. Comput. Secur. 26 (2), 154–176.

Ngugi, B., Tremaine, M., Tarasewich, P., 2011. Biometric keypads: Improving accuracy through optimal PIN selection. Decis. Support Syst. 50 (4), 769–776.

Juola, P., Noecker Jr., J.I., Stolerman, Ariel, Ryan, Michael V., Brennan, Patrick, Greenstadt, Rachel, 2013. Keyboard behavior based authentication for security. IT Prof. 15 (4), 8–11.

Pfleeger, S.L., Caputo, D.D., 2012. Leveraging behavioral science to mitigate cyber security risk. Comput. Secur. 31 (4), 597–611.

Ratkiewicz, J., et al., 2011a. Detecting and tracking political abuse in social media. In: Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media.

Ratkiewicz, J., et al., 2011b. Detecting and tracking the spread of astroturf memes in microblog streams. In: WWW'11 Proceedings of the 20th International Conference Companion on World Wide Web, pp. 249–252.

Riseman, E.M., Hanson, A.R., 1974. A contextual postprocessing system for error correction using binary *n*-grams. IEEE Trans. Comput. 23 (5), 480–493.

Shrestha, P., Solorio, T., 2013. Using a variety of *n*-grams for the detection of different kinds of plagiarism. In: Notebook for PAN at CLEF 2013, Valencia, España.

Stamatatos, E., 2007. Author identification using imbalanced and limited training texts. In: Proceedings of the 18th International Workshop on Database and Expert Systems Applications, pp. 237–241.

Stamatatos, E., 2009. Intrinsic plagiarism detection using character n-gram profiles. In: SEPLN 2009 Workshop on Uncovering Plagiarism, Authorship, and Social Software Misuse (Pan 2009), pp. 38–46.

Stein, B., Lipka, N., Prettenhofer, P., 2010. Intrinsic plagiarism analysis. Lang. Resour. Eval. 45 (1), 63–82.

Ullmann, J.R., 1977. A binary *n*-gram technique for automatic correction of substitution, deletion, insertion, and reversal errors in words. Comput. J. 20 (2), 141–147.

Upton, G., 1996. I.T.C., Understanding Statistics. Oxford University Press.

Wen, S., Jiang, J., Xiang, Y., Yu, S., Zhou, W., Jia, W., 2014. To shut them up or to clarify: restraining the spread of rumors in online social networks. IEEE Trans. Parallel Distrib. Syst. 25 (12), 3306–3316.

Wen, S., Haghighi, M., Chen, C., Xiang, Y., Zhou, W., Jia, W., 2015. A sword with two edges: propagation studies on both positive and negative information in online social networks. IEEE Trans. Comput. 64 (3), 640–653.

Wressnegger, C., et al., 2013. A close look on *n*-grams in intrusion detection, pp. 67–76.

Wu, W., et al., 2013. How to achieve non-repudiation of origin with privacy protection in cloud computing. J. Comput. Syst. Sci. 79 (8), 1200–1213.

Yang, Y.H., 2010. Web user behavioral profiling for user identification. Decis. Support Syst. 49 (3), 261–271.

Yang, P., Fang, H., 2013. Opinion-based User Profile Modeling for Contextual Suggestions, pp. 80–83.