Privacy-Preserving Community-Aware Trending Topic Detection in Online Social Media

Theodore Georgiou, Amr El Abbadi, Xifeng Yan

Privacy-Preserving Community-Aware Trending Topic Detection in Online Social Media

Theodore Georgiou, Amr El Abbadi, and Xifeng Yan

Department of Computer Science, University of California, Santa Barbara
{teogeorgiou,amr,xyan}@cs.ucsb.edu

Abstract. Trending Topic Detection has been one of the most popular methods to summarize what happens in the real world through the analysis and summarization of social media content. However, as trending topic extraction algorithms become more sophisticated and report additional information like the characteristics of users that participate in a trend, significant and novel privacy issues arise. We introduce a statistical attack to infer sensitive attributes of Online Social Networks users that utilizes such reported community-aware trending topics. Additionally, we provide an algorithmic methodology that alters an existing community-aware trending topic algorithm so that it can preserve the privacy of the involved users while still reporting topics with a satisfactory level of utility.

1 Introduction

With the explosive growth of Online Social Networks and the consequential unparalleled creation of an enormous amount of user generated content, algorithms that can extract meaningful insights and summarize this content have been widely studied and used. Specifically, the concept of *Trending Topics* has been popularly utilized in the detection of breaking news, hyper-local events, or memes, and also significantly contribute as marketing and advertising mechanisms. In its broader definition, a trending topic is a set of words or phrases that refer to a temporarily popular topic. Trending topics are used to understand and explain how information and memes diffuse through vast social networks with hundreds of millions of nodes. However, due to the open-access nature of Online Social Networks like Twitter, where everyone can see who says what, and depending on how much information a trending topic contains, novel notions of privacy emerge.

As a concrete example, Twitter reports Trending Topics by location, even at the city resolution. Their service also offers a search functionality which enables the discovery of all social postings (tweets) that contain certain keywords, and those tweets are always associated with a user of the social media service. When Twitter reports that a topic is trending in Athens, Greece, anyone can find the users that mentioned this topic through Search and may, therefore, assume that they live in Athens, Greece. The location of a user could be considered a sensitive attribute, if for example they post provocative political opinions and are afraid of physical repercussions. As we will show later, an attacker can easily infer the location of hundred of thousands of Twitter users through a simple crawling of Location-based trending topics using the official Twitter API. These users do not geocode their tweets neither publicly display their location on their profile. Thus, the

correlation between trending topics and attributes like location can lead to privacy leaks. Building smarter trending topic extraction algorithms, which contain richer demographic information of the involved users can further increase the privacy risk of any reported topic. It is important that any algorithm that extracts multiple correlated user attributes takes privacy seriously into account.

In [8] we proposed an efficient method to identify trending topics on Twitter where the underlying user population (the users that mention the topic) share common attributes like age, location, gender, political affiliation, sports teams, etc. Through human-based evaluations we showed that topics correlated with surprising attribute values tend to be 27% more interesting and informative than trending topics that are extracted purely based on their raw frequency or burstiness. We call such an algorithm a *community-aware trending topic extraction algorithm* since the involved users in each topic form homogeneous groups (communities), even if they are not linked directly.

Due to the public nature of Online Social Networks like Twitter, apart from identifying the real identity of a user, an attacker will usually try to *infer* sensitive attribute values of certain users utilizing knowledge of the social network (who is a friend with whom, or who follows who). Furthermore, a sensitive attribute inference attack is also a significant risk in the context of community-aware trending topic reporting and to the best of our knowledge has not been studied before. At the same time, large Social Media websites like Facebook and Twitter already have proprietary methods for inferring social attributes of their users that are not explicitly provided by them. Recently, it was revealed that Facebook is able to learn a user's political preference between values like "Liberal", "Very Liberal", "Moderate", or "Conservative". This is a particularly interesting case since user content on Facebook is usually not accessible to anyone except the user's immediate social network. However, if sensitive attribute information, like political preference, is used in the context of enriching other features which are publicly known, like Facebook's Trending section, then this feature could start leaking sensitive information to virtually anyone.

To demonstrate how sensitive attribute inference could be applied as an attack in the context of trending topics, we provide a hypothetical example in Figure 1 where users mention certain topics that were reported as trending from a community-aware algorithm (listed in the table at the top of the figure). The information in the table is public to everyone, similarly to the lists of Trending Topics that Facebook and Twitter already publish to their users in general. The main difference is that each topic is also linked with values for specific attributes like gender, age, location, political preference, etc. The association of an attribute value with a topic indicates that this specific attribute value is a characteristic for the majority of the users that mentioned the topic (but not necessarily all of them). For an attacker, this means that they cannot be 100% confident that every user mentioning topic T_1 lives in Boston. However, when users discuss *several* topics, the attacker's confidence may increase. As shown in Figure 1 Alice and Bob each mention some of the topics that happen to be listed in the table of trending topics. Since the attacker can obtain a list of the users that mentioned each topic (e.g., Twitter provides such search functionality), they can also *increase* their confidence (note the difference between cases (a) and (b)) in inferring Alice and Bob's sensitive attributes like political preference or gender without even accessing their posts or network.

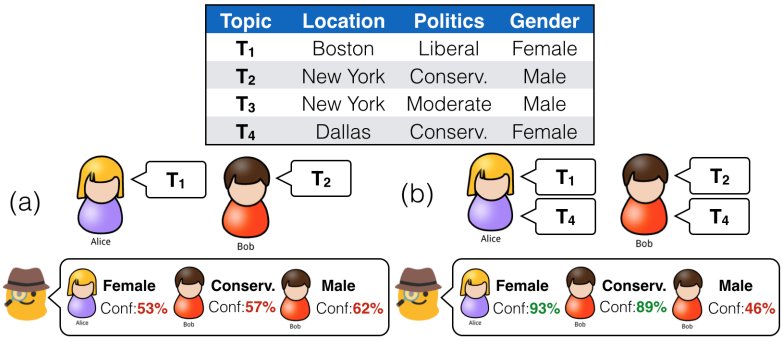


Fig. 1: Alice and Bob are two users who have discussed some topics. These topics were reported as trending and additionally, for each topic certain demographic information was extracted for 3 attributes: Location, Political Preference, and Gender. These values indicate that a significant *portion* of all the users that mentioned each topic, belong to the community defined by those values. An attacker can observe these values and can also find which topics Alice and Bob have discussed. Based on this knowledge, the attacker can infer certain attribute values of Alice and Bob with certain confidence. In case (a) (left), where Bob and Alice have only discussed a single topic, the attacker has low inference confidence. In case (b) (right), Bob and Alice have also discussed topic T_4 which increases the confidence of the attacker for Alice's gender and Bob's political preference but at the same time decreases the confidence for Bob's gender because T_2 and T_4 have mostly male and female communities correspondingly.

In Table 1 we list some real examples of topics and their corresponding community characteristics (attribute values) that we extracted from Twitter data. The communities are characterized by values for several attributes including Location, Gender, Age, Political party (US only), or even Sports teams. Note that these attribute values are temporal and might change over time, even for the same topics. Each topic has a frequency (how many unique users mentioned it) and a community defined by the attributes that describe a significant part of the users that mentioned the topic. In practice, it is impossible to observe topics where the entirety of their population forms a homogeneous community on some attribute values, therefore, the reporting algorithm will only guarantee that at least some percentage of this user population shares the reported attribute values. Note, that a community is not necessary to have a value for every attribute, as it happens for "#NFL" where the user population is homogeneous only on Gender and Location and not in Age or Politics. In the last column of the table we provide the number of privacy violations for each topic, i.e. the number of social media users that will have *at least one attribute* exposed to an attacker if the corresponding trending topic is publicly reported.

An attacker similar to the one in Figure 1 can peruse the rows of Table 1 and attempt to infer sensitive attribute values for the involved users. If there is a user that mentioned both topics #ObamaCare and #ObamaInThreeWords then the attacker can be very confident that the user supports the Republican party, that they are located in the United States, and moderately confident that they are male and a young adult. This is becomes more important In the presence of even more sensitive attributes like sexual

Table 1: Real examples of community-aware trending topics

Topic	Frequency	Community characteristics	Size	Violations
#NavyYardShooting	5427	Location: USA, Age: 19-22	5218	2561
#NFL	1534	Gender: Male, Location: USA	1212	389
#FreeJustina	54	Location: Boston, Gender: Female, Political party: Democrats	51	13
#OscarTrial	1242	Location: Johannesburg:ZA, Gender: Female	1133	345
#ObamaCare	5090	Location: USA, Politics: Republicans	4818	1002
#ObamaIn3Words	246	Location: USA, Age: 19-22, Gender: Male, Politics: Republicans	224	76
#RedSox	528	Location: Boston, Age: 19-22, Gender: Male, Team: Red Sox	411	256

orientation, religion, or race. Note that this kind of attack is different from existing privacy scenarios where the attacker infers sensitive attributes through the user's local social graph (e.g., [26]). In the case of community-aware trending topics, membership to a community is implicit and happens just by mentioning certain topics. Therefore, even if a user is careful with which groups they subscribe to or become members of, sensitive information can still be exposed simply through the mention of a topic.

We tested how easily we can attack private attributes in existing Trending Topics reports. As mentioned earlier, Twitter provides Trending Topics by location (a total of 401 cities in the world). We crawled these topics through the Twitter API, and managed to infer the location of approximately 300k users that mentioned topics which were trending only in a single location just within a single day of crawling. 11.8% of these users had public location and from a sample we estimated that this location inference attack was 82.33% successful. This proved how easy an attacker can exploit existing Trending Topics to infer the location of thousands of users. Therefore, altering trending topic algorithms to protect the sensitive attributes of OSN users is an important area to study.

Main constributions: In this research we formally introduce a novel privacy model that captures the notion of sensitive attribute inference in the presence of community-aware trending topic reports where an attacker can increase their inference confidence by consuming these reports and the corresponding community characteristics of the involved users. We discuss a basic attack and provide an efficient algorithm that preserves the privacy of each individual user so that sensitive attributes can not be successfully inferred. To the best of our knowledge we are the first to address this notion of privacy and introduce an algorithm that uses the idea of attribute generalization in combination with Artificial Intelligence techniques to efficiently defend against such attacks.

In the next sections we provide related literature on the subject of sensitive attribute inference in Social Media (Section 2), discuss the data, attack, and privacy models (Section 3), and provide an analysis of the basic attack that is based on Naive Bayes inference (Section 4), which is commonly used in this line of research. We then present a novel approach to preserve privacy while maintaining topic reports with high utility (Section 5). Finally, we provide experimental results on the algorithm's performance and utility (Section 6).

2 Related Work

Data privacy is a thoroughly studied area and several families of algorithms have been proposed to deal with different kinds of attacks, mostly on published anonymized datasets. Most notably, the concepts of k-anonymity [16, 17, 20], l-diversity [11], t-closeness [10], and Differential Privacy [6] include methodologies to preserve data privacy and information anonimity. However, privacy in Online Social Networks follows a different data model where most of the information is publicly available: the Twitter social graph, the set of online postings by every user in Twitter, user membership in Facebook pages, etc. What is not accessible though, is information about sensitive characteristics that users might want to keep hidden from the general public. An attack to discover these characteristics is known as sensitive or private attribute inference.

There are studies and published algorithms for inferring user demographics based on the content posted by social media users or their social network. Schwartz et al. [18] developed language models to identify the gender and age of Facebook users. [5] describe a method to infer user demographics by utilizing external knowledge of website user demographics and correlating it with a social media service. Their approach mainly differs from Schwartz et al.'s in its ability to infer the user characteristics without analyzing the content of postings. Nazi et al. [12] proposed a methodology to discover hidden information from Social Media by exploiting publicly accessible interfaces like the search functionality. While all the aforementioned work provides useful data mining tools and models, the privacy implications of the proposed methods are not examined.

Zheleva et al. [26] were the first to study the privacy of sensitive attributes in the context of Online Social Networks. They describe a variety of attack models to infer sensitive user attributes but the model most related to the current work, is the model that utilizes the membership of users in Facebook pages. This model is similar to the "membership" of a user to a trending topic's community. However, they do not provide any algorithmic solution since it is the choice of the user to subscribe to a page. A system called Privometer [21] measures how much privacy leaks from certain user actions (or from their friends' actions) and creates a set of suggestions that could reduce the risk of a sensitive attribute being successfully inferred, like "tell your friend X to hide their political affiliation". Similar to Privometer, in [3], and then in [14], a method is proposed for the prevention of information leakage by introducing noise, through the removal of edges or addition of fake edges, to the social graph. This idea was then extended to a finer-grained perturbation in [2] where edges are only added *partially*. Eunsu et al. [15] built a system called "curso" that identifies when a user's privacy is violated through the analysis of their local network. There are also studies that focused on the anonymization of network data where the attacker tries to statistically infer the relationship between members of the social network. Most prominent works in this area include [25] and [4]. Tassa et al. [22] also studied the same problem but specifically consider *distributed* social networks.

Dealing with privacy on a virtually infinite stream of data poses its own challenges and most of the aforementioned techniques focus on static datasets. Dwork et al. have studied privacy in streaming environments and proposed a family of algorithms called Pan-Private Streaming Algorithms [7]. The main focus of these algorithms is to deal with

attackers with control of the machine(s) where the algorithm is running but no access to the stream, while in our case they have access to every social post.

3 Data and Attack Models

3.1 Data Model

The users of a Social Media service are represented as a set $U = \{u_1, u_2, ..., u_n\}$. Each user u is associated with a vector v of k *sensitive* attributes (e.g., location, age, etc.). The attribute a_i of a user u ($u.v.a_i$) can take on one of a set of possible values $\{a_{i1}, a_{i2}, ..., a_{im_i}\}$, where m_i is the corresponding attribute's total number of unique values. The values of an attribute form a *hierarchy* which for some attributes can have a significant depth (e.g., for location: cities, to regions, to countries, to continents, to wordwide) or be trivial (e.g. for gender: from male and female to any gender). An attribute value can be *generalized* by being replaced with an ancestor value from the hierarchy. A user can mark a set of attributes as sensitive and keep them private. Or depending on the nature of an attribute, e.g., race, which the social media service might infer using its own proprietary inference algorithm, it could be considered as sensitive for everyone.

The content of the Social Media service is represented as an infinite stream P of posts. Every post $p \in P$ has a unique author (user) $p.u$ and contains an arbitrary number of topic keywords $p.T = \{t_1, t_2, ...\}$. We define a publicly available search function $SEARCH$ that returns all the users mentioning a given topic keyword t: $SEARCHt = \{p.u | t \in p.T\}$. The number of users mentioning t is referred to as *topic population* and its size is equal to $|SEARCHt|$ and referred to as *topic frequency* (second column in Table 1). We can assume that each user that mentions topic t is counted only *once* to avoid bias from spamming. The search function $SEARCH$ is defined for multiple topics as well, and returns the *intersection* of the users that mention all the given topics.

We define a *homogeneous community* as a group of users with identical values in some of their attributes, but not necessarily connected in the social graph. More formally, a *homogeneous community* contains users that share the same values for a combination of attributes $C \in \wp\{a_1, a_2, ..., a_k\}$ where $\wp$ is the powerset symbol and a_i is a user attribute (e.g., location, age, etc.). Users that live in San Francisco, are 25 years old, and are male, form a homogeneous community that contains all the users identified by these values for the attribute combination {location, age, gender}. Users in New York form another homogeneous community defined by the singleton attribute combination {location}.

A *community-aware* trending topic algorithm (referred to as *CATT* [8]) identifies topic keywords mentioned by a *homogeneous community* that has at least size ξ of the total topic population ($0 < \xi \leq 1$). For example, if ξ = .7, a topic with frequency 1000 will have *at least* 700 users forming a homogeneous community. The CATT algorithm reports records in the form of a stream of tuples: t_i, C_i, where C_i is the set of attribute values that define the homogeneous community CATT identified for topic t_i. If a topic t has no homogeneous community of size $\xi|SEARCHt|$ or larger associated with it then it isl not reported by CATT. We will refer to homogeneous communities simply as *communities* and to topics extracted via a community-aware algorithm as *community-aware topics*.

CATT extracts trending topics using a *batch-based* sliding window on the stream of social postings of the service. At the end of each window, CATT reports a set of pairs t_i, C_i) which includes all the extracted topics from the current window. We refer to the output of CATT for each window of social postings as a *batch*. Table 1 shows an example of such a batch that contains 8 pairs. Through the definition of community-aware trending topics, the users of the social media service inherit an implicit membership to communities just by mentioning certain topics. Using a single reported pair t_i, C_i one can infer that at least $\xi\%$ of the users in $SEARCHt_i$ are characterized by the values of C_i. This constantly increasing knowledge enables an attacker to gradually improve their inference confidence for a given user's sensitive attribute(s).

Note that execution of CATT requires the knowledge of community attributes for the involved users. Realistically, CATT is executed by the Social Media service itself which has access to private user information or even its own proprietary method to extract attributes. Attackers lack access to the necessary information to execute CATT themselves.

3.2 Attack Model

A CATT algorithm reports a stream of batches of pairs t_i, C_i. The attacker knows CATT's threshold ξ, as it is public knowledge, has access to the output stream, and to the search function $SEARCH$ which returns the set of users that have mentioned the provided topic(s). It is also safe to assume that the attacker has general knowledge of each attribute's prior distribution. For example, such knowledge might include the location distribution based on a Census, the age distribution based on published statistics from the social media service, the gender distribution based on users that have this information public, etc. We can safely assume that the attacker is omnipotent and can indefinitely store the pairs (t_i, C_i) and the corresponding sets of users $SEARCHt_i$. The goal of the attacker is to infer a user's sensitive attribute by exploiting the knowledge of each topic's community C_i and the users associated with it. In the presence of an omnipotent attacker a privacy preserving algorithm must maintain all previous trending topics and communities to accurately calculate the probability distribution of the sensitive attribute values, of each user.

In related literature on sensitive attribute inference [14, 21, 26], an attacker would train a Naive Bayes Classifier to choose the value of a sensitive attribute L that maximizes the probability distribution $PL|u.T$. However, though Naive Bayes is known to be a decent classifier, it is also known to be a bad estimator [24]. For the inference process to be accurate, a high probability bound is necessary, so we consider that attack to be successful only when the inference probability of an attribute value is greater than a set threshold θ (e.g., θ = .75 or .85). We will be using a global value for θ across all attributes and users, but the proposed model and algorithm support different values for each attribute and user.

4 Privacy Model

4.1 Sensitive Attribute Inference

Having established the models for the data (social stream) and the attacker (inference of sensitive attributes) we can now formally define the privacy model. For every user in the social network that discusses several topics in a streaming fashion, we want to protect against having their sensitive attribute values leaked through the continuous reporting of community-aware trending topics. Specifically, any attacker that has access to current and historical reports of community-aware trending topics should not be able to infer any user's sensitive attribute with confidence that is higher that a set value θ. At no point should an attacker be able to infer a *lower bound* for the distribution $PL|u.T$ (probability distribution of sensitive attribute L of a user u given the topics T of u), that is higher than θ.

Definition: If there is even a single case where a user's sensitive attribute can be inferred with confidence larger than θ, this comprises a *privacy violation*. A community-aware trending topic algorithm that is capable of maintaining a record of zero privacy violations while it continuously reports new batches of topics is called θ*-private*.

Referring back to the example of Figure 1, if θ is set to $.75$ then an algorithm that reports the topics in the table of the figure is *not* θ-private in case (b), since the attacker can infer the gender of Alice and the political preference of Bob with confidence that is higher than θ. To make the algorithm θ-private we would need to obfuscate the gender and political preference associated with topics T_1, T_2, and T_4. If Alice and Bob had only discussed topics T_1 and T_2, as in case (a), then the algorithm would be θ-private for this specific instance.

The inference of a sensitive attribute involves estimating the probability of a specific value given some background knowledge. As already discussed, the attacker has access to prior attribute probabilities and the output and settings of CATT. The Naive Bayes classifier is a powerful and simple technique to calculate the probability of a sensitive attribute value. Arguably, if the attacker has additional information of other sensitive attributes (e.g., already knows that Alice is a woman because she has her own photo in her profile) then they can get a better estimation of the probability of another sensitive attribute, like her location, than they would from Naive Bayes. In the following subsection we focus on the calculations necessary to get a lower bound of the probability $PL|u.T$ using Naive Bayes. The end goal is to anticipate what values the attacker can successfully infer so that they can be kept private. This is typically easy since the attacker's knowledge is generally based on publicly available information and the privacy model can incorporate it if necessary. To keep things simple, for the rest of the paper we assume that the attacker has no existing knowledge of sensitive attribute values and therefore the Naive Bayes Classifier can set a precise upper bound. The introduced privacy model is independent of how $PL|u.T$ is calculated by an attacker and the privacy preserving algorithm proposed later can be easily adjusted to calculate these distributions differently.

4.2 Naive Bayes Inference

Given a collection of topic and community tuples t_i, C_i (the output of CATT) and a search function $SEARCH$, an attacker may attempt to infer the sensitive attributes of

users that mention at least one of the topics t_i. Let u be a user that has *mentioned* k topics $t_1, t_2, ..., t_k$ and let L be one of the user's sensitive attributes (e.g., location). The probability distribution of L, given that the user mentioned some topics $t_1, t_2, ..., t_k$ is:

$$PL|t_1, t_2, ..., t_k = \frac{Pt_1, t_2, ..., t_k|LPL}{Pt_1, t_2, ..., t_k} \tag{1}$$

by applying the Bayes Rule. PL is the prior multinomial distribution of the attribute L and can be assumed to be known to an attacker based on their general knowledge on such information. The probability distribution of a user mentioning topics $t_1, t_2, ..., t_k$ given L, $Pt_1, t_2, ..., t_k|L$, is equal to the number of users u that mention all the k topics and have a specific value for L, over the total number of users with that value of L. For example, for $L = a$:

$$Pt_1, ..., t_k|L = a = \frac{|\{u|u.v.L = a, t_1 \in u.T, ..., t_k \in u.T\}|}{|\{u|u.v.L = a\}|} \tag{2}$$

where $u.v.L$ is the attribute L in the user's vector of attributes v. Similarly, the prior probability of topics $Pt_1, t_2, ..., t_k$ is equal to the number of users that mentioned these topics over the total number of users n: $|SEARCHt_1, t_2, ..., t_k|n$

While an attacker might have knowledge of the attribute's multinomial distribution and the ability to calculate the prior probability of any topic combination (using the search function $SEARCH$), they cannot compute the set of users that have a specific attribute value $L = a$: $\{u|u.v.L = a\}$. Instead, they can obtain an approximate value of the probability distribution $Pt_1, t_2, ..., t_k|L$ based on the reported tuples from CATT. The attacker can exploit the guarantees provided by CATT that a reported trending topic t_i has a population of size $|SEARCHt_i|$ with a homogeneous community C_i with size at least $\xi|SEARCHt_i|$.

More specifically, if the attribute L is not part of C_i, then the topic population of t_i follows the prior distribution of L: $Pt_i|L = PL$. If $L \in C_i$ and has a value $L = a$, then applying the Bayes Rule we get:

$$P_{approx}t_i|L = a = \frac{PL = a|t_iPt_i}{PL = a} = \frac{\xi}{PL = a}Pt_i \tag{3}$$

Similarly, the probability that a user with value $L = b$ mentions topic t_i is:

$$P_{approx}t_i|L = b = \frac{PL = b|t_iPt_i}{PL = b} = \\ = \frac{1 - \xi PL = b|SEARCHt_i|}{PL = bn} = 1 - \xi Pt_i \tag{4}$$

The attacker can now approximate the probability distribution (2) by assuming topic independence given L: $P_{approx}t_1, t_2, ..., t_k|L = \prod_{i=1}^{k} Pt_i|L$ where each factor of the product can be computed using the probability formulas from (3) and (4). Note that topic independence given L is an assumption that can be true when the number of topics k is large. An attacker can use the following formula to approximate $PL|u.T$:

$$P_{approx}L|u.T = \frac{nPL^{\,t_i \in u.T}\, Pt_i|L}{|SEARCHu.T|} \tag{5}$$

If for any value of $L = l$, the probability $PL = l|u.T$ becomes larger than the threshold θ then we assume that the privacy of this user for L is violated.

5 Privacy Preservation Methodology

A *community-aware* trending topic algorithm is also θ*-privacy-preserving* if its output does not enable the inference of sensitive user attributes with a confidence greater than a threshold θ, for any of the users involved. We will refer to this modification of the CATT algorithm as θ*-CATT*. At the same time, the goal is to keep reporting trending topics with maximum *utility*. Maximizing the utility of the results is a competing goal with preserving privacy since the algorithm could report an empty result set and the privacy leakage would be zero. Issues arise when the algorithm reports at least one trending topic t_i and its community C_i and for all users in $SEARCHt_i$ some statistical information is leaked. Especially challenging is the fact that users continuously discuss new topics which results in a constant stream of information that an attacker can use to increase their inference confidence of sensitive attribute values (as demonstrated in Figure 1).

We now introduce a novel approach that utilizes the concept of generalization in combination with Artificial Intelligence to efficiently solve the exponentially expensive anonymization problem while preserving significant utility.

5.1 Utility of Trending Topics

The goal behind extracting trending topics that certain communities focus on is to provide additional insight into why certain topics end up trending, understand which user demographics are interested in an event, product, etc., and generally provide more interesting, surprising and personalized trending topics to the users of the social media service. Using the notion of Self-information from Information Theory [19] we provide a measure of the information content for community-aware trending topics. Self-information can capture how surprising an event is based on the probability of the event. The total utility of θ-CATT's results is equal to the self-information sum of every reported topic's community. The *self-information of a community* C_i is $IC_i = -log_2 PrC_i$. Intuitively, the less likely a community is to be observed, the higher its self-information. Since we are using the logarithm with base 2, self-information is measured in bits. This metric provides a systematic way to measure the utility of the reported topics and can be used to calculate the information/utility loss when anonymization is applied. We define a *utility function* $util$ which returns the utility over a set of tuples (t_i, C_i). Other metrics can be used as well without alterations to θ-CATT.

5.2 Community Attribute Anonymization

θ-CATT needs to constantly monitor the maximum confidence of a hypothetical attacker to infer every sensitive attribute of every user in the service. When θ-CATT identifies a trending topic t_i with a homogeneous community that involves $|SEARCHt_i|$ users, it has to make sure that none of the users $u \in SEARCHt_i$ will have their sensitive attributes leaked by publishing (t_i, C_i). To ensure that, it calculates the probability of

each sensitive attribute for every user u: $PL|u.T$ and checks if the value becomes greater than θ. If it does not, then the pair (t_i, C_i) is published. If it does, θ-CATT will anonymize the sensitive attribute of the community before publishing, while preserving as much utility as possible.

We utilize the method of *attribute generalization* to achieve anonymization similarly to k-anonymity [9, 16, 17, 20]: if the city of a user can be inferred, θ-CATT reports location at the state level instead, which will alter the inference probability since a much larger population is described by this value. Generalization of categorical attributes is achieved by moving up a level in the attribute hierarchy (as described in earlier section). Depending on the depth of an attribute's hierarchy, a single generalization (moving up a single level in the attribute's value hierarchy) might lead to complete anonymization which also means zero utility for this attribute. For example, generalizing the value "male" will result to "any gender" (or "*").

The θ-CATT algorithm encapsulates the privacy-agnostic CATT algorithm which just extracts the community-aware trending topics by consuming the social stream. θ-CATT receives the batch of topics and attributes pairs (t_i, C_i) (as described in earlier section), and combined with the knowledge of every user's sensitive attributes and the topics they have previously mentioned ($u.T$), calculates if any user's privacy would leak with the publication of the batch.

5.3 Finding the Best Anonymization Strategy

In order to output a list of trending topics that contains no privacy violations, a decision must be made that involves choosing which topic communities should be anonymized without sacrificing too much utility. There are many solutions to this problem, each with a different level of utility loss. To avoid solving this problem in exponential time by trying all possible combinations and choosing the one that minimizes the utility loss, we propose an algorithm that efficiently finds the best strategy for identifying a near optimal combination to anonymize. The θ-CATT algorithm is able to identify the privacy risk each new topic-community pair poses before publishing it, ideally in real time. To achieve this computation, θ-CATT needs to store: (1) the history of trending topics previously reported by the algorithm, that each user u has mentioned, and (2) the communities that were reported to be correlated with those topics. With this information θ-CATT can simulate an attacker and identify privacy violations before they even occur.

Batch-based Anonymization When a batch of pairs (t_i, C_i) is reported by CATT, θ-CATT will iterate through all pairs, apply necessary anonymizations and publish the altered set of pairs. A naive approach to identify which pairs require anonymization, is to iterate through them one by one, and if a pair violates the privacy of at least one user, appropriately anonymize the community's sensitive attribute(s) before moving to the next topic. However, the iteration order might lead to non-optimal results where more communities get anonymized than necessary to preserve privacy and utility loss is not minimal. For example, it might be better to anonymize a single community C_3 instead of anonymizing two communities C_1 and C_2 and achieve the same privacy gain. Occasionally, the combination of two topic communities can enable their publication

without anonymization while if we each pair is individually considered, then neither of them would get reported. For this reason, θ-CATT considers the privacy and utility of the whole batch to identify the best anonymization strategy which minimizes the required attribute generalization and utility loss.

Assume for simplicity that there is a single sensitive attribute L and let S be a batch of k pairs (t_i, C_i) with communities that have a value for attribute L. Since the generalization of an attribute in a community C_i lowers the total utility of the batch, we want to generalize L in the least possible number of communities. An *anonymized batch* S' is a *modified* version of S with an arbitrary number of the communities in S anonymized (a community is anonymized when its attribute L is generalized at least once as described earlier). If a community does not contain a value for attribute L, it is ignored since it will not alter any user's inference probability for L. Therefore, there is a total of 2^k different anonymized batches S' ranging from the case where nothing is anonymized to the case where all k communities are anonymized and every possible combination in between.

The goal for θ-CATT is to find the batch S' that has greater utility than any other S'': $utilS' \geq utilS''$ while at the same time S' preserves the privacy of every user's sensitive attribute. For example, in Table 1, k = 7 and S contains the eight topic-community pairs listed in the table. If reporting these 7 pairs violates the privacy of any of the involved users, then θ-CATT will identify an anonymized version of the batch that does not leak sensitive attributes.

A* State Encoding To find the best anonymized batch S', a naive approach would be to enumerate all 2^k possible batches and keep the batch with the maximum utility, which at the same time does not leak any sensitive user attributes. However, this approach has exponential complexity $O2^k$. Instead, we propose a customized version of the A* algorithm, which is an Informed Search method [13], to identify a good batch S' *efficiently*. A* is a search algorithm, hence, it requires a search tree with a starting node and a goal node to reach. Each node of the tree is called a *state* and corresponds to a batch S'. The starting state would be the non-anynomized batch S while the goal state would be the anonymized batch S' that preserves the privacy of all involved users. There are many acceptable goal states, so additionally a cost function is needed to indicate the amount of sacrificed utility to reach a specific state.

Each anonymized batch S' corresponds to a state and all possible states form the search tree. We encode S' as a k-digit binary number where the i-th digit corresponds to the pair $t_i, C_i \in S'$. A value of 0 as the i-th digit indicates that the sensitive community attribute L in t_i, C_i is generalized, while a value of 1 indicates that it is not. Ideally, we would like to report the batch S' that corresponds to the value 111...1 (no anonymization). A batch S' is an *ancestor* of batch S'' in the search tree if their encoding differs in exactly one digit, where this digit is 0 in S' and 1 in S''. Using this notion of ancestors a search tree can be defined where the encoding 111...1 is the root node and a node's children contain all descendant encodings. For example, for k = 4, the children of root node 1111 are: 1110, 1101, 1011, and 0111. The children of 1110 are: 1100, 1010, and 0110, etc. A visual example for k = 3 is shown in Figure 2(a). All search tree branches will have

$00...0$ as the common leaf node which corresponds to a fully anonymized batch and is the least desirable result since its utility is minimal.

As the *starting state* of A* θ-CATT selects the batch S (original, non-anonymized output of the CATT algorithm) which has encoding $111...1$. The *goal state* will be the first state that has no privacy leaks (all sensitive attribute inference probabilities are below θ). Given a random state S', the neighbors are generated by flipping a single digit with value 1. If there are no such digits left, the search tree has reached its end. Given that the algorithm is stable across batches (all probabilities are below θ before a new batch), an acceptable goal state will always exist. In the worst case this will be the state with encoding $00...0$ at the bottom of the search tree (Figure 2(a)).

A* Cost Function A* requires a cost function that returns the cost of visiting each state. θ-CATT utilizes the following cost function $f.$: $fS' = gS' + hS'$. Function gS' returns the total utility loss: $gS' = utilS - utilS'$, where S is the original non-anonymized set of topics and communities. Function hS' is the *heuristic* that estimates how close the current state is to the goal state and we use the following measure: hS' = # users with a privacy violation. The number of users with a privacy violation is obtained by iterating through all the involved users in the batch and calculating the probability of inferring their sensitive attribute(s) with confidence higher than θ (equation 5). The function g measures the cumulative cost to reach a node in the search tree (how much utility has been sacrificed) and function h *estimates* the remaining distance of the goal state, where there is no privacy violation for any user. Note that this specific heuristic is not *admissible* (it might overestimate the cost to reach the goal state), which means that A* might not find the optimal path. Not finding the optimal path means that some additional utility might be sacrificed in order to greedily reach a goal state in less steps. Since the two functions g and h measure different units we normalize them with two weights α and β: $fS' = \alpha gS' + \beta hS'$ where $\alpha + \beta = 1$. The exact values of α and β depend on the total number of users (for g) and the specific utility function used (for h).

Algorithmic Complexity A* checks recursively if the current node is an acceptable goal state — number of privacy violations is equal to zero — and if it is not, it expands its children nodes and adds them in a priority queue to visit them next. Priority is calculated using the $f.$ function. This strategy enables θ-CATT to find a path to a batch S' that does not violate the privacy of any user, while reducing the number of necessary steps. The only trade-off is that the utility of the reached S' might not be optimal. For multiple sensitive attributes, the same process can be executed in parallel.

Let V be the set of sensitive attributes, k the size of the batch with pairs of topics and communities, T the set of all topics in the batch, and n the total number of users in the social network. The *time complexity* of the algorithm is:

$$O|V| \cdot k \cdot |SEARCHT| + |SEARCHT| \cdot |u.T|$$

The main bottleneck of the algorithm is the calculation of the inference probability (Equation 5) for a specific attribute and every involved user. First, the whole process must be repeated for every sensitive attribute. This entails linear complexity to the number

of sensitive attributes. Second, probability calculations must be repeated every time the cost of a state in the search tree is valuated. While there are 2^k states to explore, the customized A* with the proposed greedy heuristic can reach a local optimum in logarithmic complexity. $log_2 2^k = k$, thus, the algorithm scales linearly (amortized) with the number of topics in the batch. Finally, we need to calculate probabilities for every involved user, so the time complexity will also be proportional to $|SEARCHT|$. The inference probability formula (Equation 5) contains the product of the empirical probabilities $Pt_i|L$ where t_i is an old topic the user has mentioned and L is a sensitive attribute. To avoid calculating this product every time the inference probability is measured, we can instead store in memory the products for all topics the user has mentioned so far. The prior probability of PL needs to be calculated only once per batch and n is a fixed number (at least in the context of a batch). The only "problematic" term is the denominator of the fraction, $|SEARCHu.T|$, which requires the calculation of the intersection of every set of users that mentioned the same topics with user u. However, this value needs to be calculated only once per user, per batch. Therefore, the time complexity of the inference probability calculation is constant.

The necessary *space complexity* to store the probability products for each user and sensitive attribute is: $On|V|$.

6 Experimental Results

For our experiments we used a real Twitter dataset that contains a uniform 10% sample of the complete Twitter Firehose stream from a 39 day period between April 16 and May 24, 2014. Each tweet also contains the information of its author (user). The extracted topics include unigrams, hashtags or capitalized entities from the tweets' raw text. The four extracted user demographics include location, gender, age, and US political party preference. Location extraction was done on (1) the tweet level using Twitter's geo-tagging mechanism, and to further improve the recall, on (2) the user level using a user-provided raw text field (similarly to [23] and [1]). To extract gender and age we applied existing language models extracted from [18] on social media data. The hierarchy for gender includes the leaf nodes "male"/"female" and the top level of "all genders" or "*". Similarly, the hierarchy for age includes the leaf nodes "13-18"/"19-22"/"23-29"/"30+" and the top level "*". Finally, for political party affiliation we gathered the official Twitter accounts associated with the three most popular US political parties: Democratics, Republicans, and Libertarians. Then, a user's political affiliation was determined based on the simple majority of interactions (@-replies) with these accounts. More extensive details can be found in [8].

We consider all four attributes to be *sensitive* for every user. Then we ran two versions of our algorithms (simple CATT and θ-CATT) and compared the results. The algorithm settings are: θ = .7 (attacker's inference confidence), ξ = .5 (community size as a ratio of the topic population), utility $util\{t_i, C_i\} = \sum_{i=1}^{k} IC_i$ (self-information sum), α = .999, and β = .001. The selected values were empirically chosen to reflect a realistic scenario with a plethora of violations.

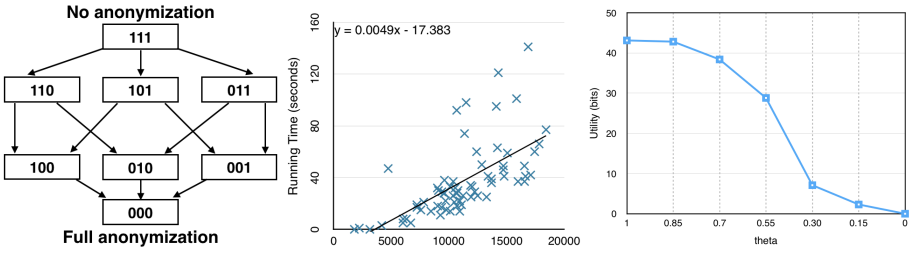
The average number of extracted trending topics and community pairs in the dataset is 112 per window (a window of data corresponds to a single batch of trending topics

Table 2: Examples of communities and the corresponding anonymized versions.

Topic	Original Community	Size	Viol/ns	Anonymized Community
#OscarTrial	Location: Johannesburg,ZA, Gender: Female	1133	345	**Location: ZA**, Gender: Female
#FreeJustina	Location: Boston, Gender: Female, Politics: Democrat	51	13	Location: Boston, **Gender: ***, Politics: Democrat
Bruins	Location: Boston, Gender: Male, Age: 19-22	196	58	**Location: ***, Gender: Male, Age: 19-22
#ObamaIn3Words	Location: USA, Age: 19-22, Gender: Male, Politics: Republican	224	76	Location: USA, **Age: ***, Gender: Male, **Politics: ***

as described in earlier section). We focus on the topics that have a specific city-level location, or age, or gender, or political party preference values, which on average is $k = 21.57$ topics per batch. The per-batch average number of unique location values is 15.2, number of unique gender and political party values is 2, and number of unique age values is 2.8. The average number of involved users is 8162. The average utility without any anonymization (simple CATT) is 43.1 bits but also contains an average of 213.2 privacy violations. Privacy violations were counted by identifying users that have inference probabilities (equation 5) for either location, age, gender, or political party preference, that is higher than θ. To preserve the privacy of the location attribute, θ-CATT anonymized on average 4.3 communities to bring the number of privacy violations to 0. The average utility of the anonymized results published by θ-CATT is 38.37 bits, so there is a total utility loss of 4.73 bits.

Examples that demonstrate cases where a community got anonymized to preserve the involved users' privacy are listed in Table 2. The 4th column lists how many privacy violations would occur if the original community was published. The 5th column shows how the proposed algorithm decided to anonymize the community by generalizing at least one attribute. *After anonymization, θ-CATT managed to bring all privacy violations to 0 so that the reported results are θ-private*. For the topic #OscarTrial the location attribute was generalized to hide the location of 345 users. For the topic #ObamaInThreeWords both age and party preference are generalized to preserve the privacy of 76 users.



(a) (b) (c)

Fig. 2: (a) Full search tree ($k = 3$). "No anonymization" is the starting state of A*. (b) Running time for $k = 21.57$. (c) Utility loss for different values of θ.

In Figure 2(c) it can be seen how the utility loss scales for different values of θ. As expected, when $\theta = 1$, an attacker must be 100% confident when inferring a sensitive attribute which in reality is practically impossible and results in maintaining the *full*

utility of the results (equal to the utility of CATT's output). On the other end, for $\theta = 0$, no information leakage is permitted at all, therefore, full anonymization of the communities is necessary and utility becomes equal to 0. These two extremes are equally not practical for a meaningful and realistic combination of trending topics with utility and preserved privacy. Based on the values in Figure 2(c) we observe that choosing a value of θ above $.6$ can maintain at least 73% of CATT's original utility of community-aware trending topics. This curve is a useful guide for choosing the desired privacy-utility trade-off.

Figure 2(b) shows the running time of our privacy preservation algorithm. All running times are recorded on a personal laptop with a 2.6GHz Intel Core i5 processor and 16Gb of RAM. There were 70 datapoints each corresponding to randomly sampled batches of topics. Since the complexity of the algorithm is mainly affected by the number of *involved users* (users mentioning one of the topics in the batch) the plot demonstrate how the running time is affected by this number. Each datapoint is an execution time (y-axis) of a single batch and corresponds a certain number of involved users (x-axis). The number of topics with sensitive attributes (batch size) was quite stable throughout our experiments with a mean of $k = 21.57$ and a standard deviation of 3.35. The plot also contains the corresponding least-square linear trendline and its equation. All reported running times are within the range of 0 seconds (no anonymizations were necessary for these batches so A* immediately found the goal state to be the starting state) and 160 seconds. Note that the time necessary to stream-in the data of a single batch takes around 3-4 minutes based on the rate of new tweets being created on Twitter, therefore, an average running time of 39.56 seconds is more than sufficient to produce results before the new batch is even ready for processing. This means that the algorithm can be used in a real-time fashion, a strong requirement for any streaming algorithm.

To examine if the running time is affected by the size of a batch k we also performed an experiment where we forced the number of topics to be always equal to 15—an arbitrarily selected value that is less than 21.57—by randomly dropping some topics. We observed that the running time is also increasing *linearly* with the number of users, as expected. Altering k had no apparent effect on how the running time scales with the number of users, similar to the slope of the trendline in Figure 2(b), which proves that the greedy heuristic of A* has sublinear amortized complexity. Based on the projected trendlines in Figure 2(b), we estimate that the running time for $100K$ users, which is a number that can be observed for trending topics on the Twitter web-page, would be approximately 490 seconds which is again acceptable based on the rate of generated tweets. Therefore, our algorithm satisfies the efficiency requirement of a practical real-world setting.

7 Conclusions and Acknowledgments

With the introduction of algorithms that extract trending topics that correlate with user demographics (community-aware topics), novel ways emerge to attack sensitive user information through attribute inference. We are the first to address privacy concerns in this context, by demonstrating how an attacker can statistically infer sensitive attribute values and introducing a privacy model for the preservation of these sensitive values of each individual user that discusses trending topics in a social network. Towards this end, we propose a new algorithmic approach that utilizes Artificial Intelligence methods in

a novel way to efficiently identify when a privacy violation may occur and remedy all violations by efficiently extracting an optimal anonymization strategy which maximizes the utility of the reported trending topics and corresponding community characteristics.

Acknowledgments: This work is supported by NSF grant CNS 1649469.

References

1. Achrekar, H., Gandhe, A., Lazarus, R., Yu, S.H., Liu, B.: Predicting flu trends using twitter data. In: Computer Communications Workshops. pp. 702–707 (2011)
2. Boldi, P., Bonchi, F., Gionis, A., Tassa, T.: Injecting uncertainty in graphs for identity obfuscation. Proc. VLDB Endow. 5(11), 1376–1387 (Jul 2012), http://dx.doi.org/10.14778/2350229.2350254
3. Bonchi, F., Gionis, A., Tassa, T.: Identity obfuscation in graphs through the information theoretic lens. In: Proceedings of the International Conference on Data Engineering. pp. 924–935. ICDE, Washington, DC, USA (2011), http://dx.doi.org/10.1109/ICDE.2011.5767905
4. Campan, A., Truta, T.M.: Data and structural k-anonymity in social networks. In: PinKDD 2008. pp. 33–54 (2009), http://dx.doi.org/10.1007/978-3-642-01718-6_4
5. Culotta, A., Ravi, N.K., Cutler, J.: Predicting the demographics of twitter users from website traffic data. In: Proc. of the Conference on Artificial Intelligence. pp. 72–78 (2015), http://dl.acm.org/citation.cfm?id=2887007.2887018
6. Dwork, C.: Differential Privacy, pp. 1–12. Springer (2006), http://dx.doi.org/10.1007/11787006_1
7. Dwork, C., Naor, M., Pitassi, T., Rothblum, G.N., Yekhanin, S.: Pan-private streaming algorithms. In: Innovations in Computer Science - ICS 2010, Tsinghua University, Beijing, China, January 5-7, 2010. Proceedings. pp. 66–80 (2010), http://conference.itcs.tsinghua.edu.cn/ICS2010/content/papers/6.html
8. Georgiou, T., El Abbadi, A., Yan, X.: Extracting topics with focused communities for social content recommendation. In: Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing, CSCW 2017, Portland, OR, USA, February 25 - March 1, 2017. pp. 1432–1443 (2017), http://dl.acm.org/citation.cfm?id=2998259
9. LeFevre, K., DeWitt, D.J., Ramakrishnan, R.: Incognito: Efficient full-domain k-anonymity. In: Proceedings of the 2005 ACM SIGMOD international conference on Management of data. pp. 49–60. ACM (2005)
10. Li, N., Li, T., Venkatasubramanian, S.: t-closeness: Privacy beyond k-anonymity and l-diversity. In: ICDE 2007. pp. 106–115 (2007)
11. Machanavajjhala, A., Kifer, D., Gehrke, J., Venkitasubramaniam, M.: L-diversity: Privacy beyond k-anonymity. ACM Trans. Knowl. Discov. Data 1(1) (Mar 2007), http://doi.acm.org/10.1145/1217299.1217302
12. Nazi, A., Thirumuruganathan, S., Hristidis, V., Zhang, N., Shaban, K., Das, G.: Query hidden attributes in social networks. In: 2014 IEEE International Conference on Data Mining Workshop. pp. 886–891 (Dec 2014)
13. Nilsson, N.J.: Problem-Solving Methods in Artificial Intelligence. McGraw-Hill Pub. Co. (1971)
14. Raymond, H., Murat, K., Bhavani, T.: Preventing private information inference attacks on social networks. IEEE Trans. Knowl. Data Eng. 25(8), 1849–1862 (2013), http://dx.doi.org/10.1109/TKDE.2012.120

15. Ryu, E., Rong, Y., Li, J., Machanavajjhala, A.: Curso: Protect yourself from curse of attribute inference: A social network privacy-analyzer. In: Proceedings of the ACM SIGMOD Workshop on Databases and Social Networks. pp. 13–18. DBSocial '13, New York, NY, USA (2013), http://doi.acm.org/10.1145/2484702.2484706
16. Samarati, P.: Protecting respondents' identities in microdata release. IEEE Trans. on Knowl. and Data Eng. 13(6), 1010–1027 (Nov 2001), http://dx.doi.org/10.1109/69.971193
17. Samarati, P., Sweeney, L.: Generalizing data to provide anonymity when disclosing information. In: PODS. vol. 98, p. 188 (1998)
18. Schwartz, H., Eichstaedt, J., Kern, M., Dziurzynsk, i.L., Ramones, S.: Personality, gender, and age in the language of social media: The open-vocabulary approach. In: PLoS ONE 8(9) (2013)
19. Shannon, C.E.: A mathematical theory of communication. SIGMOBILE Mob. Comput. Commun. Rev. 5(1), 3–55 (Jan 2001), http://doi.acm.org/10.1145/584091.584093
20. Sweeney, L.: Achieving k-anonymity privacy protection using generalization and suppression. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10(05), 571–588 (2002)
21. Talukder, N., Ouzzani, M., Elmagarmid, A.K., Elmeleegy, H., Yakout, M.: Privometer: Privacy protection in social networks. In: Workshops Proceedings of the International Conference on Data Engineering, ICDE. pp. 266–269 (2010), http://dx.doi.org/10.1109/ICDEW.2010.5452715
22. Tassa, T., Cohen, D.J.: Anonymization of centralized and distributed social networks by sequential clustering. IEEE Transactions on Knowledge and Data Engineering 25(2), 311–324 (Feb 2013)
23. Vieweg, S., Hughes, A.L., Starbird, K., Palen, L.: Microblogging during two natural hazards events: what twitter may contribute to situational awareness. In: Proceedings of the SIGCHI conference on human factors in computing systems. pp. 1079–1088. ACM (2010)
24. Zhang, H.: The optimality of naive bayes. AA 1(2), 3 (2004)
25. Zheleva, E., Getoor, L.: Preserving the privacy of sensitive relationships in graph data. In: International Conference on Privacy, Security, and Trust in KDD. pp. 153–171 (2008), http://dl.acm.org/citation.cfm?id=1793474.1793485
26. Zheleva, E., Getoor, L.: To join or not to join: The illusion of privacy in social networks with mixed public and private user profiles. In: Proceedings of the International Conference on World Wide Web. pp. 531–540 (2009), http://doi.acm.org/10.1145/1526709.1526781