A Web forum sanitised of profanity disguise by means of sequence alignment[[1]](#footnote-3)

Un foro Web libre de obscenidades enmascaradas utilizando alineación de secuencias[[2]](#footnote-4)

*Name and two last names[[3]](#footnote-5)*

*Name and two last names[[4]](#footnote-6)*

Abstract

Profanity is the use of offensive, obscene or abusive vocables or expressions in public conversations. A big source of conversations in text format nowadays are digital media such as forums, blogs or social networks where malicious users are taking advantage of their ample worldwide coverage to disseminate undesired profanity aimed at insulting or denigrating opinions, names or trademarks. Lexicon-based exact comparisons are the most common filter technique within these media; however, ingenious users are disguising profanity using transliteration or masking of the original vocable while still conveying its intended semantic (e.g. by writing *piss* as *P!55* or *p.i.s.s*), hence defeating the filter. Recent approaches to this problem inspired in the sequence alignment methods from comparative genomics in bioinformatics, have shown promise in preventing overlooking such guises. Building upon those results we have developed an experimental Web forum where user comments are screened against disguised profanity. In this paper we introduce the software (ForumForte) and describe briefly the technique and engineering behind it, as well as some empirical evidence of its filtering performance. Our software is open-source under the New BSD License and is available at: <http://tinyurl.com/ForumForte>.

Resumen

Las obscenidades son palabras o expresiones consideradas inapropiadas en conversaciones públicas por su carácter ofensivo o vulgar. Los medios digitales popularizados hoy en día como blogs, foros y redes sociales, son la principal fuente de conversaciones en forma de texto y es común encontrar en ellos el uso de obscenidades para insultar a o denigrar de opiniones, personajes o marcas, una anomalía cuyo agravante es mayor si se tiene en cuenta la amplia cobertura mundial que pueden alcanzar. El uso de diccionarios de palabras vetadas como mecanismo de filtrado es insuficiente debido a la versatilidad del lenguaje escrito, que permite a los usuarios inventar ingeniosas variantes mediante transliteraciones o enmascaramientos del texto (por ejemplo cambiar *mierda* por *m1erd@* o *m.i.e.r.d.a*). En recientes estudios se han reportado resultados interesantes para disminuir esta anomalía usando técnicas inspiradas en la genómica comparativa. Siguiendo esta misma linea de trabajo, hemos desarrollado un foro Web experimental donde los mensajes ingresados por los usuarios son inspeccionados y depurados de obscenidades transliteradas o enmascaradas. Este artículo presenta dicho software con una descripción breve de su diseño y su uso con datos reales provenientes de foros de noticias de periódicos y de trinos de una red social. El software es libre y se distribuye en línea en:<http://tinyurl.com/ForumForte>.

Keywords: Web forum, profanity detection, text analysis

Palabras clave: Foros Web, detección de obscenidades, análisis de texto.

# Introduction

An essential feature of Web2.0 digital media is their ability of crowdsourcing user–generating-content, motivating collaboration and mutual construction of scenarios so as to yield richer user experiences [1]. An illustrative example are digital forumsand blogs where multiple users generate written comments about their own or other’s opinions. Unfortunately some users abuse of this freedom of speech forinappropriate purposes such as insulting, degrading or boosting opinions, participants, brands or any other concept by menas of offensive or obscene language. For these reasons, usually this kind of digital services must be moderated by website administrators in order to guarantee profanity–free user–generated text content.

A naïve moderation tool are lexicon—based filters which screen text against a black–list of forbidden terms. A weakness in these filters is that they carry on exact comparisons missing variants with involuntary typos or misspellings, or more worryingly, variants disguised with transliterated or masking symbols written deliberated to circumvent the filter; the resulting variants still visually convey the actual meaning of the profanity. Take for example the vulgar slang term *piss* transliterated as *P!55* or masked as *p-i-s-s* or worse still, a combination of both, *P-!-5-5*. Any of these attacks would easily defeat a literal comparison filter, but the message would still be clear for most readers. It is evident that the number of guises of this type grows combinatorial in size; thus, the lexicon-based approach is impractical. The anomaly is illustrated in Figure 1.

**Figure 1. Disguised profanity in a Colombian newspaper forum.**

**Source: Forum pages of eltiempo.com.**

Similar anomalies have been identified in many other digital platforms [2], [3] and even more, has been characterised as a security threat [4]. Recent approaches to tackle this problem taking inspiration on bioinformatics techniques for sequence alignment of genomes from different organisms, have shown promising results [5], [6]. Building upon those results, we have developed an experimental profanity–safe Web forum (ForumForte). Our software was conceived as a concrete application of our previous results on the problems of revealing masked terms in spam email [5], automatic evaluation of fill-in-the-blank questionnaires [7] and automatic syntax verification of short blocks of code in programming languages [8]. An in-depth technical description of the filtering mechanism would be reported in a forthcoming paper.

In the following we describe the software, the technology behind it, and a proof-of-concept of its potential application for content moderation in mainstream applications such as newspaper forums and micro–blogging media platforms. This is an extended version of a short paper recently published in the *Proceedings of the 10th Colombian Computer Conference* -10CCC- that was held in Bogotá, September 2015 (see [16]). It is worth noting that ForumForte is free software under the New BSD License and is available online or for download at: <http://tinyurl.com/ForumForte>.

# Materials and methods.

# **2.1** Similarity trees of disguise profanity.

As it was observed before, the lexicon-based (exact comparison) approach against profanity disguising is not practical. A similar comparison difficulty was faced by bioinformaticians some decades ago in the field of comparative genomics,where the goal was to find common genetic motifs between different families of species. The “text” in that case was regarded as the sequences of DNA and protein molecules from the genome and proteome of living organisms [11], [12], written in an alphabet of letters representing the initials of their molecules (in the genome case {A, G, C, T} for (A)denine, (C)ytosine, (G)uanine and (T)hymine). Different organisms would have different genomes but when the sequences are aligned, similarities between sub–regions (genes) are found, except for a few places that differ. The small variations are due to mutations that insert, delete or substitute one molecule or the other. The mutation may imply a change on the function of the phenotype that the gene codes for.

An example of sequence alignment is shown in Figure 2, where a gene whose phenotype is expressed during synthesis of vitamin C is depicted for six species of mammals. The resemblance of the sequences is striking, although some differences, most surely due to mutations, are highlighted. As it can be seen, the last exon of the gene is very similar among cows, dogs, and rats, whereas it differs in one molecule deleted across humans and great apes. It is known that the former group can make their own vitamin C whereas the latter group have to take it from diet.

**Figure 2. Sequence alignment of genes coding for vitamin C synthesis in six species.**

**Source: [13].**

The evolutionist assumption that explains these variations in the genome is that mutations occurred during millions of generations of descendants from a common ancient genome, yielding the diversification of different species. Such diversification can be depicted as a phylogenetic tree, where branches grow every time a preserved mutation happens. In the previous example, the tree depicting mutations in the last exon is shown in Figure 3 (mutations highlighted in red). The genetic code of the rat is placed as the root, meaning that it would be the common ancestor of the six species (variants). From there, two branches diverge: one branch goes through the dog (substitution, A ➝ C) down to the cow (another substitution, different location, C ➝ T); the other branch goes to an unknown middle ancestor (a deletion, C) from where two branches open up, one for the orangutan (substitution, G ➝ A), and one for humans and chimpanzees (substitution, G ➝ T).

**Figure 3.Phylogenetic treeof the gene exon coding for vitamin C in Figure 2.**

**Source:** authors own creation**.**

We adapted the idea of phylogenetic tree diversification to the profanity disguise anomaly described earlier. That is, we assume that the guises of a profanity vocable grow down in a similarity tree from a common ancestor (its canonical text) to the variants obtained by recurring application of edits or corrections made on the predecessors (see Figure 4, edits highlighted in red). Instead of keeping all the possible trees whose depth increase combinatorially with all possible edits, the idea behind our profanity detection mechanism is to trace back the disguised variant up to its common ancestor via classical sequence alignment algorithms [11], [12] or alternatively, using approximate string matching algorithms [9]which were independentlydeveloped by computer scientists for the same purpose.The key concept of these algorithms is the edit distance between two texts [10], which is the number of character corrections (substitutions, deletions or insertions) that are needed to transform one text into the other. A special-purpose distance would be designed to correctly account for the edits that yield the profanity guises, as we shall describe later.

**Figure 4. Similarity tree of disguised variants of the offensive word BASTARD.**

**Source:** authors own creation**.**

The approximate sequence matching algorithm (seee.g. [9], [11], [12]) carries out a pairwise comparison of the characters in the two sequences (the user–generated text and the canonical profanity vocable), while accumulating the number of edits (insertions, deletions or substitutions) needed to transform one sequence into the other. The essential insight for detecting transliteration is to overlook the substitution of visually “twin” symbols (e.g. substituting ‘o’ by any of {0, °, ó, ò, ö, ô, Ø, θ, O}, see Figure 5), whereas for masking, the insight is to overlookthe insertion of bogus segmentation characters such as { . , \* , ~ , ¦ , ­­– , \_ , : , ; , “,”}. These couple of edits should add no value to the distance (or difference) between the two sequences, whereas edits such as deletion, insertion or any other substitution should count. Hence the design of the edit distance between two symbols in their respective sequences is outlined in Figure 6, where *d* indicates if the edit counts or not in each of the cases mentioned above (this function was originally introduced in [5]).

**Figure 5. An excerpt of the lists of twins substitutions.**

**Source:** authors own creation**.**

**Figure 6. Edit distance function used in the filter mechanism.**

**Source:** authors own creation**.**

# **2.2** Software design.

We embarked on the development of ForumForteas a test-bedto verify the robustness of the filter mechanism described above, within an easy-to-use open forum where comments can be writtenabout particular topics; no censorship is carried out, nor personal or usage information is collected. Simply put, ForumForte is a forum wall that screens comments against profanity; when profanity or disguised profanity is detected, the corresponding fragments are overwritten with a mask of asterisks, and both the original and filtered text sequences are posted to the wall. ForumForte was designed as a Web application based on the software architectural MVC pattern [14] and the Java EE platform [15]. In the following, we shall describe the most re-presentative design artefacts obtained during its development.

Let us start by summarising the use–case scenarios for the software (Figure 7). There are two kind of forum users: *visitors* and *administrator*. A *visitor* can inspect the fora pages as well as their contents, filter them by topics and indeed, post a comment in which case the profanity filter mechanism is activated and detection statistics would be collected. Lastly, he can download files for installation and upload a JSON file for batch processing of comments, as we will explain later on.

**Figure 7. ForumForte use-case diagram.**

**Source:** authors own creation**.**

The other type of user is the *administrator*. This user has a password–protected account which allows him to carry out basic maintenance tasks such as forum and subjectcreation, elimination, cleaning, password update, etc. His other usages are related to the filter mechanism: updating the profanity canonical lexicon, look at the performance statistics per forum or profanity vocable and fine-tunethe profanity tolerance parameters. These tolerances refers to the maximum edit distance for which two text sequences can be considered as equivalent (a value 𝜏 ∈ {0, 1, 2, 3}).

For the purposes of illustration, next we present the dynamical models of one of the main use cases in the forum, namely *posting a comment*. The activity diagram is shown in Figure 8, and the sequence diagram in Figure 9.All the other use case dynamical models are available on request.

**Figure 8.*Posting a comment*activity diagram.**

**Source:** authors own creation**.**

The structural model of the software was designed as an MVC-based class diagram organised in three packages, namely business, model, and controller. (see Figure 10). The business package include classes Forum, Admin, Filter and Bean; these classes implement the logics of the functionalities previously described. On the other hand, the model package encapsulates the ForumBoard, Subject, Threshold,Transliteration, Profanity; these classes are responsible of managing the visualisation of forum comments and user interface. Finally, the controller package consists of classesController, Login, Log4jInit and AdminControl; these classes control interaction with both kind of users. Detailed views of this general model are also available.

**Figure 9.*Posting a comment*sequence diagram.**

**Source:** authors own creation**.**

**Figure 10. ForumForte class diagram overview.**

**Source:** authors own creation**.**

Next we discuss briefly the persistence model of the software, which was implemented as a relational database using the Java Persistence API framework that carries out the mapping from the structural model. The corresponding ER diagram is shown in Figure 11, consisting of tables: forum, subject, comment, filteredComment, profanity, and tolerance. The latter were included because the administrator may fine–tune the edit-distance tolerance per profanity term, and therefore statistics per tolerance are collected.

**Figure 11. ForumForte persistence model.**

**Source:** authors own creation**.**

Finally, the deployment diagram of Figure 12, shows the sub-systems used to run the software as a Website application. It can be seen there, that the Catalina Server 2.4 provides the execution environment for the components AdministrationandWebForum included in the ForumForte.war package. System configuration is defined in the corresponding properties and XML files. This environment runs on aApache Tomcat 7.0.42 server within a Red Hat v6 LinuxOS. Data persistence is achieved through a socket connection to a MySQL 5.5 server within a ForumForte database session.

**Figure 12. ForumForte deployment diagram.**

**Source:** authors own creation**.**

# **2.3 Datasets**.

We gathered a profanity comment dataset consisting of a collection of 300 user comments (in Spanish) from the publicly–available news forums of a Colombian newspaper, traced during a time frame of 25 months (01-Jan-2011 to 31-Jan-2013). Every message in this dataset contains profanity that were not blocked by the forum filter. The dataset was prepared to be as diversified as possible; with respect to word length, 58% (176) comments are shorter than 30 words, 25% (74) are medium-long, between 31-50 words, and the remainder 17% (50) ranges between 51 and 150 words. The majority of comments include only one use of profanity (76% or 227 comments), followed by two uses of profanity (20% or 61 comments); there are extreme cases such as a comment containing seven obscenities, and cases where the full comment is precisely one profanity word. Other statistics of the dataset are shown in Table 1.

**Table1. Dataset summary statistics.**

**Source:** authors own creation**.**

On the other hand, we also built a lexicon of 60 spanish obscenities, varying in length, the mode being 4-6 letters long (60%), with extremes being three 9 letter-long and one 2-letter long words.Excerpts of the datasets are illustrated in Table 2 and Table 3.

**Table2. Example comments from dataset.**

**Source:** authors own creation**.**

**Table3. Example profanity from lexicon.**

**Source:** authors own creation**.**

# Results

# **3.1** Software execution.

*Online release and on-site installation*

ForumForte is installed an available online in <http://tinyurl.com/ForumForte> (last visit: November20th, 2015). The Web application works on any Web browser (Firefox, Chrome, Explorer and Safari). Alternatively, interested users can download and install the software in their own servers (user and installation guide are also available from the same URL).

*Forum visitor usage*

A visitor may browse the available fora by clicking the *Foros* option in the menu bar, which redirects to the page shown in Figure 13. Any choice here would display the list of subjects in each forumpreviously defined by the administrator. Then, by choosing one of the subjects, the visitor would be taken to the actual forum page, where comments made by other visitors would be shown (see Figure 14). We remark that interaction with the forum is anonymous and no private or network access information is collected by the system.

**Figure 13. List of available fora.**

**Source: s**napshot from ForumForte pages**.**

The layout of an actual forum is very intuitive and easy to use. Basically,there is a text box on the top of the page for the visitor to write his or her comment. The visitor can choose to either clean the current content of the text box, or to post the comment in which case, it would be screened by the filterengine. Once the text is processed, the original and filtered text would be posted to the forum wall as the most recent entry (the one just below the text box).

**Figure 14. The appearance of an arbitrary forum.**

**Source: s**napshot from ForumForte pages**.**

Each entry consists of the filtered text sequence aligned over the original text. If one or many disguised profanity are detected they would be overwritten with a mask of asterisks on the top line of the entry. Comments are kept in the wall in chronological order, most recent first. Lastly, the visitor may choose to display all comments in the forum, only profanity–marked comments, or only profanity–free comments (see Figure 15).

**Figure 15. Display options of forum comments.**

**Source: s**napshot from ForumForte pages**.**

*Forum administration*

The administrator main page shows the dash board of Figure 16,where typical maintenance operations in forums, subjects, profanity lexicon and statistics reports are carried out. For this purpose, the user should choose the *Administración* option in the menu page, and confirm his identity with a valid password. The software supports a unique administrator whose username and password can be updated at convenience using the *Login* choice. The remainder optionsaforementioned, redirect to Web pages where the administrator can create, eliminate or clean contents of the corresponding item.

**Figure 16. The administrator dash board.**

**Source: s**napshot from ForumForte pages**.**

For the sake of illustration, a sample subject administration page is shown in Figure 17 where the clean up and remove commands are visible. The administrator can navigate to the actual forums pages and visualise the comments or participate in the discussion, by clicking over the forum name. Similarly, the profanity lexicon module allows to create, remove or tune the transliteration tolerance of non-admitted vocables in the forum pages.In addition, the statistics module reports detection rates discriminated by forum or by profanity vocable(see Figure 18). The latter are furthermore broken down into individual rates per tolerance parameter, providing valuable information for tuning purposes with respect to particular vocables and transliteration attack patterns. We remark that forum statistics are computed instantly with its current comment contents whereas profanity statistics are historical, beginning at the moment they were created or assigned a different tolerance parameter. Statistics are lost when the associated item is removed.

*External input*

ForumForte features an interesting interface to apply its filtering mechanism to external sources of user–generated text. This feature takes advantage of the JSON format for content extraction and storing provided by theTwitter®social network trough its publicly–available API. This digital media platform is essentially a world–wide community forum for freeshort text messaging (comments no longer than 140 characters, known as tweets) with no moderation, and therefore a real–world practical scenario of the ideas motivating our development.

**Figure 17. A sample subject administrator page.**

**Source: s**napshot from ForumForte pages**.**

**Figure 18. The performance statistics module.**

**Source: s**napshot from ForumForte pages**.**

In order to try this feature, the user should prepare an input file with a JSON format. Such file can be obtained by logging in into the Twitter API console with a valid user account[[5]](#footnote-7)and then extracting tweets for a chosen user profile or trend topic. The file can then be processed in ForumForte entering the special–designed forum found in the path *Foros→Filtrado de trinos*. In contrast with the other fora, this one features two buttons to load the JSON file and to submit it to the filtering engine (see Figure 19), which would subsequently process each tweet in the file and post it to the forum wall as if it was originally typed in by a visitor.We highlight that by adhering to the Twitter JSON format, it is possible to filter user–generated text from other sources.

**Figure 19. The specially designed forum for processingexternal JSON Twitter files.**

**Source: s**napshot from ForumForte pages**.**

# **3.2** Filtering performance.

The detection performance of ForumForte on real-life data was tested with the dataset described in Section 2.3. Profanity ground truthwas obtained by manually labelling the occurrences of disguised obscenities from the profanity lexicon observed in each comment within the dataset. The experiments were carried out by processing the dataset using the JSON external input feature and statistics module of ForumForte. The 300 comments were screened against each profanity item in the lexicon, and detection rates were collected. For the sake of easier illustration of the results, we summarised the experiments by grouping profanities according to their character length. In each group, the experiments were repeated for the values 𝜏 = {0, 1, 2, 3} of the tolerance parameter.

The resulting detection rates are reported in Table 4, which demonstrate the feasibility and promise of the method in real–world scenarios. In there, greyed values denote detection rates closest to the ground truth (notice that values above the ground truth indicate occurrence of false positives). In profanityof short length (*m*≤ 6) a tolerance 𝜏 = 0 is more effective. This may be explained because in short sequences, insertion or deletion of even just a single character would result in a not immediately easy to interpret variant of the original word (e.g. *coo→coño, pQuta→puta*); therefore attackers prefer to use substitutions to obtain profanity guises in these groups. On the other hand, medium or larger sizeprofanity (*m*≥ 7), require higher tolerances 𝜏 = {1, 2} for better detection, since in these cases deletions and insertions are easier to interpret and thus attackers tend to use such edits rather than substitutions to disguise profanity. Lastly, a tolerance 𝜏 = 3 is clearly not recommended in any group since it yields extremely high false positive rates.

**Table4. Detection rates in the real-life dataset.**

**Source:** authors own creation**.**

# Conclusions.

Profanity disguising in user–generated text exploits the robustness of the human mind to visually interpret the semantics of a message overwritten with substitutions of twins symbols or insertion of bogus segmentations. Lexicon–based exact comparison filters, on the contrary, have limited ability to detect such variants. Taking inspiration on algorithms for sequence alignment widely–used in bioinformatics, we have built an academic profanity-proof Web forum that counter fights such anomaly, showing promising results. We anticipate this technology may have interesting applications for content moderation in Web2.0 digital services such as forums, blogs and social networks. It is worth to note that although the software described in this paper was targeted at Spanish obscenities, the mechanism is language–independent and requires no training before use other than setting up the lexicon of canonical forbidden terms.

A critical task during real-life operation of the forum would be tracking the behaviour of malicious users in coming up with new disguising stratagems that may imply periodic adaptation of the lexicon along with their tolerance parameters, so as to ensure high detection rates. Performing such tuning automatically is an interesting topic of further research. Besides, issues regarding speed and concurrency on high–volume content–generation environments are another [straightforward](http://www.thefreedictionary.com/straightforward)avenues of future study.

# References

[1] T. O’reilly, “What is Web 2.0: Design patterns and business models for the next generation of software,” Communications & strategies, no. 1, p. 17, 2007.

[2] S. Sood, J. Antin, and E. Churchill, “Profanity use in online communi-ties,” in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM, 2012, pp. 1481–1490.

[3] W. Wang, L. Chen, K. Thirunarayan, and A. P. Sheth, “Cursing in English on Twitter,” in Proceedings of the 17th ACM conference on computer supported cooperative work & social computing, 2014.

[4] M.-E. Maurer and L. Höfer, “Sophisticated phishers make more spelling mistakes: using URL similarity against phishing,” in Cyberspace Safety and Security. Springer, 2012, pp. 414–426.

[5] S. A. Rojas-Galeano, “Revealing non-alphabetical guises of spam-trigger vocables,” DYNA, vol. 80, pp. 15 – 24, 2013.

[6] X. Zhong, “Deobfuscation based on edit distance algorithm for spam filitering,” in Machine Learning and Cybernetics (ICMLC), 2014 Inter-national Conference on, vol. 1. IEEE, 2014, pp. 109–114.

[7] V. P. Cardona-Zea and S. A. Rojas-Galeano, “Recognising irregular an-swers in automatic assessment of fill-in-the-blank tests,” in Engineering Applications (WEA), 2012 Workshop on. IEEE, 2012, pp. 1–4.

[8] S. A. Rojas-Galeano, “Towards automatic recognition of irregular, short-open answers in Fill-in-the-blank tests,” Tecnura, vol. 18, 2014.

[9] R. A. Wagner and M. J. Fischer, “The string-to-string correction prob-lem,” Journal of the ACM, vol. 21, pp. 168–173, 1974.

[10] V. I. Levenshtein, “Binary codes capable of correcting deletions, inser-tions, and reversals,” Soviet Physics Doklady, vol. 10, no. 8, 1966.

[11] S. B. Needleman and C. D. Wunsch, “A general method applicable to the search for similarities in the amino acid sequence of two proteins,” Journal of Molecular Biology, vol. 48, no. 3, pp. 443 – 453, 1970.

[12] T. F. Smith and M. S. Waterman, “Identification of common molecular subsequences,” Journal of Molecular Biology, vol. 147, no. 1, 1981.

[13] D. Venema. (2013, June 28) Evolution basics: Genomes as ancient texts. The BioLogos Forum. Last visited: September 14th, 2015. [Online]. Available: http://biologos.org/

[14] A. Leff and J. T. Rayfield, “Web-application development using the model/view/controller design pattern,” in Enterprise Distributed Object Computing Conference, Proceedings. Fifth IEEE International, 2001.

[15] D. Alur, D. Malks, J. Crupi, G. Booch, and M. Fowler, Core J2EE Patterns (Core Design Series): Best Practices and Design Strategies. Sun Microsystems, Inc., 2003.

[16] C. Mogollón Pinzón, S. Rojas-Galeano, “A genomic-based profanity-safe Web forum,”Proceedings of 10th Colombian Computing Conference, IEEExplore, ISBN 978-1-4673-9461-1, 2015.

1. Submitted on: \_\_\_\_\_\_\_\_\_\_. Accepted on: \_\_\_\_\_\_\_\_\_\_\_\_. This article is derived from \_\_\_\_\_\_\_\_\_\_\_\_ investigation project. [↑](#footnote-ref-3)
2. Fecha de recepción: \_\_\_\_\_\_\_\_. Fecha de aceptación: \_\_\_\_\_\_\_\_\_\_. Este artículo se deriva de un proyecto de investigación denominado (*nombre en caso de ser un proyecto de investigación*), Código o Numero de registro (*En caso de tener*). Desarrollado por el grupo de investigación *(Nombre del grupo ó grupos)* de la Universidad \_\_\_\_\_\_\_\_, Cuidad, Colombia [↑](#footnote-ref-4)
3. Titulo de pregrado y universidad. Titulo de maestría y universidad. Titulo de Doctorado y Universidad. Cargo Actual y Filiación Institucional. Cuidad, Colombia. Correo electrónico: \_\_\_\_\_\_\_\_\_\_\_\_\_\_. [↑](#footnote-ref-5)
4. Titulo de pregrado y universidad. Titulo de maestría y universidad. Titulo de Doctorado y Universidad. Cargo Actual y Filiación Institucional. Cuidad, Colombia. Correo electrónico: \_\_\_\_\_\_\_\_\_\_\_\_\_\_. [↑](#footnote-ref-6)
5. The API Twitter is available at: https://dev.twitter.com/rest/tools/ console (last visit: November 20, 2015) [↑](#footnote-ref-7)