Keeping Your Online Community Safe

The Nuts and Bolts of Filtering User Generated Content

Overview

Branded online communities have become an integral tool for connecting companies with their clients and prospects. This comes in the form of forums, blogs, ratings and reviews, commenting, and social networks. A central aspect of these online communities is user generated content (UGC). When a visitor to a company's web property leaves a comment on the blog or writes a question in the forum they are producing user-generated content. This type of content allows customers to interact with the brand and each other. It adds color and credibility to an otherwise one-way conversation. For example, users can share stories about their experiences with a company's products or gain valuable information and insight about a product by reading other users' reviews before making a purchase. All of this information falls into the broad category of user generated content.

As companies allow more UGC and grow their online communities, the users begin to control the tone and image of the community. This can be both good and bad, depending on the users. Relinquishing this control to the users can empower them and drive further engagement by others. Companies do face a risk, however; as users do not always have the best interest of the companies in mind and might abuse the privilege of an open forum by posting inappropriate content. This abuse creates a strong argument for closer monitoring of user generated content. This is the point when most companies decide to invest in filtering technology or other solutions to protect their online presence. However, filtering UGC is not as simple as it seems. Users are sophisticated and highly imaginative when it comes to working around automated filters.

The move to online communities with user-generated content will continue to expand. So, how can companies confidently launch online communities without worrying their brands will be at risk? There is good news; intelligent filtering technology is available that protects the online community from unwanted and inappropriate content without sacrificing performance or security. It is important to point out that an intelligent filter is not intended for the sole purpose of blocking profanity; CleanSpeak is a customizable solution that can look for behaviors and questionable content as well as profanity. Good profanity filters are also fully customizable in real-time, which means that new words and phrases can instantly be added to be filtered or to generate alerts for moderators. For a closer look at the challenges of filtering user-generated content, we discuss some classic problems in UGC filtering, the techniques users employ to circumvent the filter and how an intelligent filter succeeds in protecting the online community.

Classic Filtering Problem

A classic filtering problem is called "word sense disambiguation." A filter is challenged to determine the correct meaning of a word based on its usage and placement within a

sentence, because language is full of ambiguous cases of word usage. For example, one word may have five meanings. The problem exists because filters might incorrectly determine the meaning of the word based on the context or might ignore the context completely. Additionally, contemporary users have become more creative and are now creating new words spontaneously. For example, online games and communities allow users to create usernames to identify themselves. In some cases, these new usernames contain profanity. Whether or not the user was intending to be profane and create a new profanity or was simply unaware of the profanity in their username can only be determined by a thorough examination of the text. These types of situations still fall into the broad category of word sense disambiguation and require that a filter inspect the context of the word to determine its meaning.

For the purpose of this discussion, the category of problems known as word sense disambiguation has been divided into three main problems. These are:

- Stand-alone words
- Embedded words
- Non-dictionary words

Each of these problems is solved using similar tactics, but each problem is distinctly different than the others. Let's look at each problem and discuss the possible solutions.

Stand-alone Words

Stand-alone words only occur when a word is both profanity and a dictionary word or proper name. The most common example of a stand-alone word is given in the sentence in Example 1.

The movie is a **Dick** Van Dyke classic.

Example 1: *Stand-alone word*

As you can see, this sentence contains a word that could be considered profanity. However, in this context it is clearly a person's name. Context-based word sense disambiguation exists when profanity is also a dictionary word or name and, only based on the context of the word, can it be determined whether or not it is, in fact, profanity.

Stand-alone words pose a difficult problem for filters. The issue is that the filter must examine the adjacent words to determine the meaning of the questionable word. In some cases the adjacent words might flag the word as profanity, but it was not intended as such. This is also known as a false-positive.

This type of problem often requires a sophisticated solution that scores adjacent words based on how frequently they indicate the word is a profanity. Often the solution requires an initial "learning" process that teaches the filter how to score adjacent words. There currently isn't a steadfast solution to this problem. Implementing an AI system would be one of the better approaches if not for the excessive cost to build, train, and regularly re-train the system. Instead, Inversoft developed a gray list and employs techniques that combine filtering phrases like "you are a dick" and ignoring other phrases like "dyke van dyke" to manage these edge cases.

Embedded Words

Embedded words occur when a dictionary word or proper name contain profanity. Example 2 illustrates this situation.

Don't **ass**ume you will win the game.

Example 2: *Embedded word*

As you can see, this sentence contains a common dictionary word, "assume". This word contains profanity, but is not itself profanity. This case only occurs when dictionary words contain profanity. In general there are very few profanities that exist in dictionary words. However, those that do exist in a large number of dictionary words, pose a challenging scenario for filters.

This case is quite simple to handle. Filters can simply look for dictionary words that contain profanity and safely ignore them. This can be done preemptively or during the filtering process. Poorly written filters will often get caught up on these simple cases and flag large numbers of dictionary words as profanity. Inversoft's CleanSpeak filter pulls from a large set of dictionary words to correctly handle this situation and minimize false positives.

Non-Dictionary Words

Non-dictionary words occur in a variety of cases, the most common of which is when users become creative while deciding on a username. There are other cases when non-

dictionary words might occur by user error, such as misspelling of dictionary words. Example 3 illustrates these two cases of non- dictionary words.

My username is black**ass**h.

Don't **ass**oom you will win the game.

Example 3: *Non-dictionary words*

As you can see, both of these sentences contain profanity embedded in another word. This problem is made worse by the fact that neither of the words the profanity is embedded is in a dictionary word. This makes it difficult to determine if the user was being creative or purposefully being profane. Example 4 illustrates the latter.

Cases of non-dictionary word embedding are difficult for filters to correctly handle. They require that the adjacent characters are analyzed in the same manner that the adjacent words are analyzed for the Stand-Alone Words case. Filters normally fall into two categories when considering non-dictionary words. The first category of filters ignores these cases because the profanity is embedded inside other words. This approach will eliminate false-positives, but it also increases the number of misses, meaning that profanity might slip through and be displayed for all users to read on the website. The second category of filters will always identify these cases as profanity. This increases the number of false-positives while decreasing the number of misses.

That guy is an **ass**hat.

Example 4: *Non-dictionary profanity*

Inversoft has addressed this problem with a feature called ‘filter mode.’ This feature allows for adjustment of the filter aggressiveness for each word or phrase in the blacklist. By enabling the site owner to change the settings in filter mode, CleanSpeak is able to minimize the number of false positives without increasing the number of misses.

Filter Circumvention

Besides the common cases of word sense disambiguation that we discussed above, there are a number of other cases that UGC filters must consider. All of these cases occur when a user is attempting to fool the filter and allow profanity through. This is also known as

attacking the filter or a filter attack. The six most common cases of filter attacks that users employ are:

- Character replacement (AKA “leet” speak)
- Swapping and collapsing
- Repetition
- Abbreviation
- Separators
- Image processing

To get a better understanding of each of these cases, let's cover them individually and discuss the problem and possible solutions.

Character Replacement

Character replacement is the process of replacing certain characters with other characters, usually symbols, that look the same or similar. This is a method that is used to attack filters that ignore characters that aren't alphabetic letters. Example 5 illustrates some cases of possible character replacement.

#1 **$**ally is my neighbor

#2 |**)**on't be a menace

#3 |**<**nive**$** can be dangerous

#4 \/\/hat are you doing?

Example 5: *Character replacements*

Case #1 in Example 5 illustrates a single character replacement. Here, a user is replacing the “S” character with the “$” (dollar-sign) character. Since the dollar-sign character has the same overall shape as the “S” character, it is simple for the reader to discern that text contains the word “Sally”.

Case #2, #3 and #4 illustrate multiple character replacements. Here, a user is using multiple characters to replace a single character. #2 is using the “|” (pipe) character and the “)” (right-parenthesis) character to create a capital “D” character. #4 is using a combination of forward and backward

slashes to create a capital “W” character. Example #3 goes a step further. It is replacing two different characters in the text. Both the “K” and “S” characters are being replaced.

Character replacement is often used to attack filters because the majority of the filters in use today don't understand character replacements. In most cases, character replacements can be identified by swapping back in the original characters prior to filtering. However, this could produce false-positives if the user was correctly using a character such as a dollar sign. Therefore, a filter must be careful not to assume that all non-alphabetic characters are filter attacks . Inversoft's CleanSpeak is capable of handling character replacements without incurring false-positives.

Swapping and Collapsing

Character swapping and collapsing is the process of replacing characters with other alphabetic characters or removing unnecessary characters while still retaining the phonetic structure of the word. This tactic is often used to attack filters that do not understand phonetics. Example 6 illustrates some cases of character swapping and collapsing.

#1 T**ee**ch me guitar

#2 Atta**k** the main castle gate

Example 6: *Swapping and collapsing*

Case #1 illustrates how the replacement of the letters “ea” with “ee” retain the same phonetic structure and allow the reader to infer the word. This is an example of character swapping since the “a” has been swapped with “e” in the word “Teach”.

Case #2, on the other hand, is an example of character collapsing. In this example the “ck” in the word “Attack” has been collapsed to a single “k” character. This collapsing still retains the same phonetic structure of the word and allows the reader to infer the original word.

In some cases characters can't be collapsed without changing the meaning of the word. For example, the word “been” can't be collapsed to “ben”. Therefore, a filter can't simply ignore multiple characters that are phonetically the same. It has to understand if the word can be collapsed.

This type of filter attack is a common tactic of users due to the fact that most filters have no knowledge of phonetics. CleanSpeak is able to correctly identify the majority of these types of phonetic attacks. It understands that many characters can be swapped, and in some words, multiple characters can also be collapsed. It also knows which words can't be collapsed without impacting the meaning of the word.

Repetition

Repetition is another commonly used filter attack that involves the repetition of characters in a word. This tactic fools filters which are not designed to ignore multiple instances of the same character. Example 7 illustrates some cases of character repetition.

heeeeeeeeelllllllllllooooooo

Example 7: *Repetition*

This example is somewhat more sophisticated since multiple "E" characters are normally pronounced differently than a single "E". However, the brain can easily detect the pattern within the text and extract the correct word from it.

Most simple filters are unable to detect repetition of this kind and extract the word from it. Therefore, many uses will employ this type of attack to circumvent a filter. CleanSpeak is capable of detecting this type of filter attack and will correctly identify words regardless of repetition.

Abbreviation

Abbreviations are a relatively new addition to tactics used to attack UGC filters. In most cases, abbreviations are harmless. However, there are a number of modern abbreviations that are used to shorten common inflammatory sayings. Example 8 illustrates one of these modern abbreviations.

WTF (What The F-word)

Example 8: *Abbreviation*

This example illustrates that there are abbreviations that contain profanity or other unwanted words and therefore should be handled by the filter.

In most cases, abbreviations can be handled by the majority of filters by adding any unwanted abbreviations to the list of words being filtered. CleanSpeak contains a number of the most common abbreviations that contain profanity and is able to filter those abbreviations out. In addition, certain types of applications may prefer to allow abbreviations like this as they are less inflammatory than the full phrase. CleanSpeak is also capable of ignoring abbreviations.

Separators

Probably the most sophisticated attack that users employ against UGC filters is known as inserting separators. This process involves inserting separators, such as spaces or periods, between characters of a word such that the word can still easily be read. Example 9 illustrates some cases of using separators.

These examples illustrate how the simple process of inserting additional non-alphabetic characters between the characters of the word does not interfere with the reader's ability to identify the word correctly.

h....e....l....l....o

h e l l o

h....e l l....o

Example 9: *Separators*

Most filters view characters such as spaces and periods as word boundaries and often stop filtering when they reach a word boundary. The major issue with adding separator support to filters is that if the filter is not careful, it can easily introduce false-positives.

Example 10 provides an example of what a naïve filter might consider profanity.

I'm going to sma**sh it**

Example 10: *Separator false positive*

It might be difficult to see the profanity in Example 10, but if you look at the last 4 characters on their own, you'll see it. Filters that are not intelligent about separators will incorrectly identify this sentence as inflammatory. Therefore, the filter must understand how word separators behave within sentences and how they can be used as an attack.

HaaEaaLaaLaaO

Example 11: *Complex separators*

Separators can also be extremely complex. In some cases, other alphabetic characters can be used as separates without impacting the ability of the reader to identify the word. Example 11 illustrates this principle.

Furthermore, by changing various characters in a sentence from lowercase to uppercase, a user is capable of combining characters and words as separators. Example 12 illustrates this type of attack.

Obviously, there is a breaking point at which the reader can no longer disassociate the word from its surroundings. However, in most cases, the brain can eventually see the pattern in the text and pull from it whatever the user has intended to write.

CleanSpeak is capable of handling basic separators that use common punctuation marks such as periods, semi-colons and asterisks. It is also capable of handling spaces without introducing false- positives. It understands that word boundaries are used correctly more often than not, but that spaces can still be used to attack the filter. Complex attacks that use other alphabetic characters or uppercase letters as an attack can pose a challenge for even the most sophisticated filters. Inversoft continues to solve these and other technical challenges through a rigorous product development process.

He Eats a Lot of oLd pOwerbars

Example 12: *Combination separators*

Image Processing

The last method that users sometimes employ to attack UGC filters is known as image processing. The attack involves the same principle as a picture montage. A picture montage is constructed of many small images that are arranged in such a manner that together they form a larger image. In the same manner, users can create words by scattering characters on the page. Example 13 illustrates this principle. As you can see the word "run" is spread over three lines.

see spot r

u

n

Example 13: *Image*

This situation is somewhat easy for filters to handle since it is identical to the separator attack discussed above. The only difference here is that the separators are a combination of spaces and new lines (carriage returns). This means that any filter that is capable of handling separators can also handle these types of attacks.

However, users further expand on this theme by adding additional words between the characters of the word "run" to create complex forms of separators. Example 14 illustrates this form of attack.

See spot r

good u

n dog

Example 14: *Complex image processing*

This form of attack can become exceptionally complex, but simple for the reader to look at and read. Advertising, signs, logos and product labeling has made use of vertical word placement and other image processing techniques long enough for the public to become not only accustomed to this type of image, but also good at quickly interpreting and reading them.

The majority of the UGC filters on the market can only handle a subset of these situations. However, any filter that provides a comprehensive solution for the either the separators or image processing forms of attack will also successfully handle the other form of attack. Inversoft is unique in that its filter handles some of these image processing attacks.

Summary

We have covered the major types of problems that UGC filters encounter as well as many of the most common attacks used against UGC filters. A sophisticated filter can no longer simply look for unwanted words in a text. It now has to understand sentence and word structure and apply a variety of tactics when filtering. The vast majority of filters currently available are not capable of handling most of the cases we have presented here.

Inversoft's CleanSpeak is one of the most comprehensive filter solutions available on the market today. With the prevalence of user generated content growing in customer facing online communities and users becoming more adept at dodging basic filters, most companies require a solution that will continue to evolve. Inversoft continually invests in its filtering technology to ensure that it works as quickly and as effectively as possible.

For more information on the range of filtering products available from Inversoft, visit our website at http://www.inversoft.com or call us toll-free at 1-888-423-7814.