DETECTING CONTEXTUAL ANOMALIES IN LEXICAL REASONING MACHINES

Jonathan A. Zdziarski
jonathan@nuclearelephant.com

ABSTRACT

Modern day language classification employs conceptual machine learning which relies heavily on presented learning input. Most algorithms used today to classify text (Bayes, Chi-Square, Markovian Discrimination, and others) are inherently sound and accurate, however regardless of which is used, a great deal of any algorithm's accuracy is related directly to the quality of data provided – the Garbage In, Garbage Out rule. Many text samples contain a degree of malapropos data, or "noise". This paper outlines a statistical approach for detecting and removing these anomalies within a text sample by applying a series of machine-generated contexts to small subsets of text. The disposition of each context can then be learned to provide a medium of contrast against the disposition of the underlying data. This allows a classifier to identify material which is out of context. After removed, the remaining text provides a cleaner subset of data for classification and with a higher level of confidence.

Categories and Subject Descriptors

I.5.1 [**Pattern Recognition**]: Implementation – s*tatistical, structural*.

I.7.m [**Document Text and Processing**]: Miscellaneous

General Terms

Algorithms

Keywords

anomalies, contextual, noise, patterns, noise reduction, classification, confidence, text, filtering, pattern recognition

1.INTRODUCTION

Reasoning machines bear the responsibility of improving their responses regularly to provide better results. As part of this process, they should be able to detect, or try to detect, that their evidence is self-contradictory[5]. Lexical data can sometimes be misleading in that a data point (a word or token) can take on one disposition when in context, but represent a different or even inverse disposition when out of context. This occurs frequently in human communication, however today's text classifiers don't yet understand how to compensate for such occurrences. When analyzing texts created by humans, therefore, the input sample is likely to contain some degree of contradictory, out-of-context data (or noise) that doesn't belong. This noise can often lead to a misunderstanding (misclassification) if not removed. With these out-of-context elements removed from the sample, however, the classifier is left with more *good* data to perform its function. This, naturally, reduces the risk of a classification error and strengthens the classifier's confidence in its decision.

By placing subsets of text within a machine-generated context, noise can be detected by identifying anomalies between individual data points and the contexts they belong to. The detection of contextual anomalies can be applied to any type of text sample in most fields of research, however the examples provided in this paper will be applied to the most common function of language classifiers today, email classification. This being the ideal field of research for this type of algorithm, as noisy text runs rampant in all forms – from the noise in every day chit-chat to malicious word list attacks injected by spammers. If the detection algorithm is able to perform effectively in this type of environment, it is certain to operate effectively in many less aggressive fields of language classification.

The machine-generated contexts we'll discuss in this paper are based on each token's assigned probability (value), which the classifier should already be assigning. The foundation of a machine-generated context, however, could be based on other relevant tiers of data when applied to other, more foreign concepts in the field of machine learning.

The generated contexts themselves, as we'll see, are specific to a user's particular training set and are learned in training, providing an approach capable of adapting to individual behavior.

2.DETECTING ANOMALIES

In this section, we'll explore the three-step detection algorithm used for detecting malapropos text within a sample. The three steps involved in noise identification are:

a. Instantiating lexical contexts based on token probabilities
b. Training and distinguishing interesting contexts
c. Identifying statistical anomalies within each given context

2.1 Instantiating Lexical Contexts

The detection process begins with the instantiation of a series of lexical patterns (or contexts) which will be used as our chosen medium of contrast. The patterns instantiated are based on a window of individual per-token values, which have already been assigned to each token by the classifier. The value of each data point is assigned to a band (by rounding) with a width of $1/20^{th}$ of the range used by the implementor (in most cases, this is 0.05). For example, with a window size of 3, the following patterns may be formed.

Fig. 2.1 Instantiating Lexical Contexts

tokens:	Viagra	is	great	for
values:	0.92	0.64	0.34	0.71
bands:	0.90	0.65	0.35	0.70

contexts: 0.90_0.65_0.35
0.65_0.35_0.70

Each instantiated context consists of the value bands found inside its pattern window. If multiple input paradigms are used (for example, a series of single tokens and a series of biGrams), two separate sets of context patterns should be instantiated and assigned different names within the classifier. Prepending a unique identifier to the name of the contexts belonging to a paradigm could be used to differentiate. The actual name given to each pattern context is entirely at the discretion of the implementor.

After performing instantiations for an entire text sample, up to B^N different contexts will be instantiated per input paradigm, where B represents the number of bands within the value range of the classifier and N the window size of the pattern. Typically, values range from 0.00 – 1.00 providing 21 different 0.05-wide bands for a total of 9,261 possible patterns using the standard window-size of 3 tokens per pattern. Only active contexts (that is, those related to patterns found within the input sample) need be instantiated at the time of processing.

2.2 Training and Distinguishing Contexts

Once a series of pattern contexts has been instantiated, they must be learned through the classifier's existing statistical training mechanism. Each context should represent a *meta-token* in the classifier's lexicon, and patterns trained using a similar approach to tokens. This is usually through supervised and unsupervised training. The active contexts can be trained for every message processed, or fine-tuned to become more sensitive to uncommon text by training only on hard-to-classify messages. The higher the required threshold of uncertainty for training, the more sensitive the algorithm will become to noise in commonplace text (e.g. the pasting of a novel, as opposed to a collection of hot words). Depending on the level of filtering desired, it's up to the implementor to choose when to train. Should a classification error occur, the corrective training should also correct the counters of the active contexts in similar fashion as the standard data points are retrained by the classifier. Training approach ultimately defines the usefulness and accuracy of detection.

Each context is stored with a counter for each disposition available, for example *spamHits* and *innocentHits*. Once a minimum training threshold has been reached, the patterns may be assigned a value using P. Graham's[3] approach for calculating probability without bias. For example:

$$P_C = \frac{spamHits / totalSpam}{(spamHits / totalSpam) + (hamHits / totalHam)}$$

Standard rules for assigning values to hapaxes may be applied at the implementor's discretion. Hapaxes should be assigned a fairly neutral value well within the exclusionary radius (discussed next) that they will be ignored until sufficiently trained. After some initial training, the more interesting pattern contexts will take on the disposition of one particular class of text, for example:

Contexts Present in Guilty Text (Spam)

0.25_1.00_1.00 [0.99990]
0.35_1.00_1.00 [0.99990]
1.00_1.00_0.20 [0.99990]
1.00_0.40_1.00 [0.81868]
1.00_1.00_0.25 [0.99990]
0.55_1.00_1.00 [0.99990]
1.00_1.00_0.35 [0.99990]
0.25_1.00_1.00 [0.99990]
1.00_1.00_0.15 [0.99990]
0.15_1.00_1.00 [0.99990]
0.10_1.00_1.00 [0.99990]
0.20_1.00_1.00 [0.99990]
0.00_0.00_0.45 [0.99990]

Contexts Present in Innocent Text (Non-Spam)

0.65_0.20_0.00 [0.00900]
1.00_0.60_0.15 [0.21000]
0.00_0.80_0.55 [0.00900]
0.00_0.25_0.90 [0.00900]
0.15_0.05_1.00 [0.00900]
0.60_0.85_0.25 [0.12900]
0.00_0.60_0.90 [0.02000]
0.70_0.05_1.00 [0.17000]
0.85_0.95_0.10 [0.00900]
0.75_0.90_0.50 [0.00600]
0.65_0.65_0.75 [0.00600]
0.40_0.95_0.10 [0.16699]
0.95_0.25_0.65 [0.02000]

In the selected patterns above, notice each pattern is assigned a very strong disposition, yet many of the underlying data points within the pattern are contradictory (out of context).

2.3 Identifying Anomalies Within a Context

After an initial period of training, a pattern (0.65_0.35_0.70 for example) may resolve to a very strong disposition, such as 0.95. This denotes that the classifier found the presence of three adjacent tokens with values falling into 0.65, 0.35, and 0.70 bands, respectively, had a 95% probability of belonging to the disposition associated with that probability (in our case, spam). If, therefore, the context with which these data points fall into is considered guilty, then any members of the context resolving to a contradictory disposition are clearly outside of the context.

In order to identify these inconsistencies, it's first necessary to identify interesting pattern contexts. This can be done by creating an exclusionary radius from a neutral value (0.5 in our example) around each pattern present in the message. This exclusionary radius is defined based on the range of values considered 'useful' by the implementor. Most Bayesian implementations would use a radius of 0.25, as tokens falling within 0.25 – 0.75 are usually considered inert. For example:

ABS(0.99000-0.5) = 0.49 > 0.25 Interesting
ABS(0.15000-0.5) = 0.35 > 0.25 Interesting
ABS(0.35000-0.5) = 0.15 < 0.25 Not Interesting
ABS(0.65000-0.5) = 0.15 < 0.25 Not Interesting

Once interesting patterns have been identified, the next step is to identify inconsistencies between a context and the data points included within that context. This is accomplished by measuring a delta between the data point's probability (P_T) and context's (P_C) using ABS(P_C-P_T). We then identify the data points which fall far outside of the context's disposition by, using a second exclusionary radius. A reasonable radius for token distance for most implementations is approximately 1/3 of the value range used by the implementor. In Bayesian implementations, for example, 0.33 would be used. This radius may also be fine-tuned to reduce or increase the algorithm's sensitivity to anomalies. Increasing this radius will decrease the total number of anomalies identified and decreasing it will increase the number. Comparing the context's disposition against the values of each element, we see that the middle token's band in our 0.65_0.35_0.70 example is out of range as ABS(0.95-0.35) = 0.60 > 0.33.

		0.95			Context
[	0.65	**0.35**	0.70	]	Tokens
[	is	**great**	for	]	

Similarly, should the context have resolved to have a strong disposition closer to 0.00 (for example, 0.15), we would see that the two end tokens' bands are out of range as ABS(0.15-0.65) = 0.50 > 0.33 and ABS(0.15-0.70) = 0.55 > 0.33.

		0.15			Context
[	**0.65**	0.35	**0.70**	]	Tokens
[	**is**	great	**for**	]	

Once we have identified data points with an inconsistent disposition to the context they're included in, these tokens can be eliminated from the classification criteria. While eliminated from classification, they **should not be eliminated from training**.

2.4 The Noise Reduction Process Illustrated

The formula below describes the pseudo-code in Fig. 2.2 which illustrates the implementation of the noise reduction process using the values described in this paper. Let x represent an instance in X, the set of windows to evaluate. The goal is to approximate our hypothesis (H_x) based on the value of our window (P_w) and token (P_t) and their corresponding radii (R).

$$(\forall x \in X)[H_x = ((|0.5\text{-}P_w| > R_x) \rightarrow ((\forall t \in T)[(|P_w\text{-}P_t|) > R_t]))]$$

Fig. 2.2 The Noise Reduction Process Illustrated

```
let windowSize = 3
let windowRadius = 0.25
let tokenRadius = 0.33

function begin
begin loop[sample] (tokens in sample)
  instantiate context (WP,windowSize)
  load value PW for context
  if (ABS(0.5-PW)>windowRadius)
  begin loop[token] (tokens in context)
    load value PT for token
    if (ABS(PW-PT)>tokenRadius)
      eliminate token from classification
  end loop[token]
end loop[sample]
function end
```

3.FURTHER IMPROVEMENTS

In this section, we'll explore some potential improvements that could be made by the implementor to expand the functionality of this algorithm. These additional approaches are merely ideas which the implementor may find useful in certain circumstances, and should not be implemented as a standard function of the algorithm.

3.1 Expanding Use of Existing Data

With the additional data generated by this algorithm, additional uses may include:

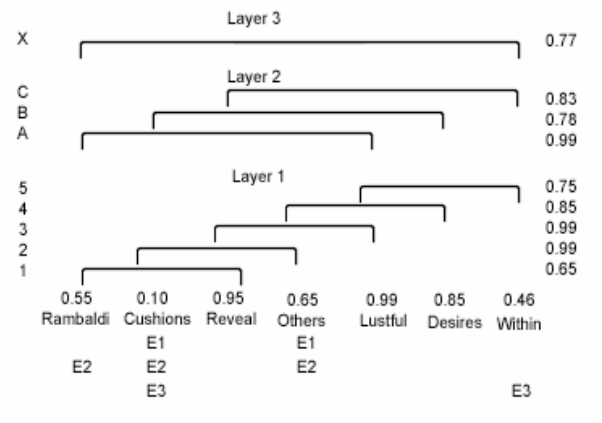
1. At the discretion of the implementor, the context names themselves may also serve as useful tokens in the statistical combination. This may further help the effectiveness of the interesting patterns identified.
2. It may be useful to perform a statistical combination of active patterns in the message to determine a "noise reduction result" to be consulted should the classifier find the message difficult to classify.
3. The number of eliminations can be used to calculate a signal-to-noise ratio for the message. This value could be factored in to any final decisions about the message.

3.2 Extended Detection via Layering

One particularly interesting innovation involves extended identification of anomalies using contextual layering. With layering, patterns of patterns of data points and patterns of patterns of patterns of data points are learned in the same fashion as the original top-level contexts. The extended detection takes place after the first detection pass has been completed. All second layer patterns including effective top-layer patterns (those with eliminations) are now processed against all tokens included in the active second layer. Any data points whose value falls outside of the exclusionary radius of this layer's pattern values are eliminated at this stage. Finally, any third layer patterns which include the effective second layer patterns are instantiated and the process repeats. The layered process is best explained when illustrated as shown in Fig. 3.1, where we see the following process take place:

1. Inconsistencies are discovered in patterns 1, 2, 3, and 4.
2. Pattern A, B, and C are instantiated because they include patterns with inconsistencies
3. Pattern A's value is compared to the token values included in pattern A, and additional inconsistencies are found (Rambaldi).
4. Pattern X is instantiated because it contains pattern A
5. Pattern X's value is compared to the token values included in pattern X, and additional inconsistencies are found (Within).
6. All inconsistencies are eliminated as classification datum

Fig 3.1 Extended Detection via Pattern Layering



4.SUPPORTING DATA / EXPERIMENTS

To evaluate this algorithm, we'll process some real-world samples of text. The samples were taken from an actual user's mailbox and processed against the algorithm after a sufficient training cycle for the same user as outlined in 2.2. It's important to note that this algorithm performs its function without any knowledge about the disposition of the sample it is classifying.

4.1 Results on Spam Classification

We see in Fig. 4.1 that we are presented with a message consisting of mostly irrelevant text. While some of the text is useful for identifying spam against the test user, there are an abundance of innocent and neutral tokens as well. These types of malicious spam often provide only a tiny bit of useful guilty information to identify them by. The remaining text is merely a flood of junk pseudo-conversational text in an attempt to fool spam filters.

Fig. 4.1 Real-world micro-spam with word-list attack

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML><HEAD><TITLE>Message</TITLE>
<META content="MSHTML 6.00.2800.1276" name=GENERATOR></HEAD>
<BODY> <DIV> </DIV> <DIV></DIV>
<DIV class=OutlookMessageHeader lang=en-us dir=ltr align=left><FONT face=Tahoma size=2>-----Original Message-----
<B>From:</B> cierra myers [mailto:sangglenna@techemail.com]
<B>Sent:</B> Tuesday, February 03, 2004 10:09 AM
<B>To:</B> Penny Kelly
<B>Subject:</B> >>Attract your mate

</FONT></DIV><FONT face="Courier New" size=1>My Saturdays nights are no longer spent by the fire "reading a good book". four seconds until picture isdownloaded
<A href="http://www.largeinfo.com/argo.com" wjyf.com mnhoavekpkjnimoixmomhioecwshvvl="http://emjwxbhol wirhkwwaoufvtcfamrydfnqx"><IMG src="http://www.netstarsite.com/argo.net" fnqusjjsqenrqgwn.com rqmjrqpkoccxavkbfhfrilkctbameurvfuepwjd="http://joydnx vnslffgngrejhustv" NOSEND="1"> </A>

No? drawled the dragonette; it seems to me very babyish
How old is your mother? asked the girl
Oh! I really think, continued the boy, nodding sagely, that it wouldn't be well to have these Records scattered around
Mother's about two thousand years old; but she carelessly lost track of her age a few centuries ago and skipped several hundreds Their use would givesome folks unfair advantage over others, you know </FONT></BODY></HTML>

Based on the user's training, the following text was determined to be out of context.

HTML PUBLIC HTML HTML TITLE TITLE OutlookMessageHeader us dir ltr left Tahoma cierra myers sangglenna techemail February To Attract Courier Saturdays nights spent good book four seconds isdownloaded href wjyf fnqusjjsqenrqgwn NOSEND No drawled dragonette babyish Oh! nodding sagely well these Records Mother's about two but she carelessly lost track her age few centuries skipped several Their use givesome folks advantage know HTML

0.16 1.00 0.16 0.16 0.43 0.43 0.40 0.27 0.31 0.40 0.44 0.24 0.40 0.40 0.40 0.40 1.00 0.23 0.40 0.32 0.40 0.30 0.22 0.09 0.25 0.39 0.09 0.40 1.00 0.40 0.40 0.40 0.28 0.40 0.40 0.40 0.40 0.40 0.40 0.17 0.26 0.40 0.40 0.24 0.22 0.49 0.61 0.40 0.43 0.16 0.76 1.00 0.41 0.40 0.16 0.39 0.24 0.19 0.40 0.11 0.61 0.38 0.16

Leaving the following text and values remaining:

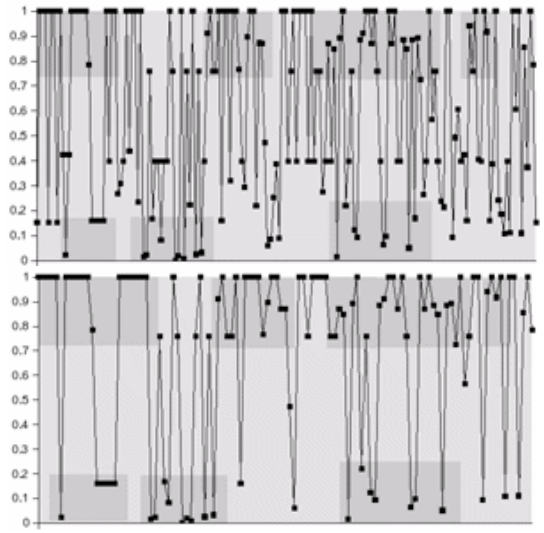
W3C DTD Transitional EN HEAD Message META content MSHTML name GENERATOR HEAD BODY DIV DIV DIV DIV DIV class lang en align FONT face size Original Message BR From mailto com BR Sent Tuesday AM BR Penny Kelly BR Subject your mate BR BR FONT DIV FONT face New size My are no longer by the fire reading until picture BR com IMG src com BR BR the it seems to me very BR How old is your mother asked the girl BR really think continued the boy that it wouldn't be to have scattered around BR thousand years old of ago and hundreds would unfair over others you FONT BODY

1.00 1.00 1.00 1.00 1.00 0.02 1.00 1.00 1.00 1.00 1.00 1.00 0.79 0.16 0.16 0.16 0.16 0.16 1.00 1.00 1.00 1.00 1.00 1.00 1.00 0.02 0.02 0.76 0.17 0.08 1.00 0.76 0.00 0.02 0.01 0.76 1.00 0.03 0.76 0.03 0.91 1.00 0.76 0.76 1.00 0.16 1.00 1.00 1.00 1.00 0.77 0.90 1.00 1.00 0.87 0.87 0.47 0.06 1.00 1.00 0.76 1.00 1.00 1.00 1.00 0.76 0.76 0.87 0.85 0.02 0.89 1.00 0.22 0.76 0.12 0.10 0.89 0.91 1.00 1.00 0.87 1.00 0.76 0.07 0.10 1.00 0.87 1.00 0.89 0.85 0.05 0.88 0.89 0.73 1.00 0.57 0.76 1.00 1.00 0.10 0.94 1.00 0.92 1.00 0.11 1.00 1.00 0.11 0.86 1.00 0.79

At first glance, we see what appears to be a lot of would-be junk text in the post-processed example, however a closer look at their underlying values shows that they are very useful data points for evaluating the message. What's left is a much cleaner, consistent set of data for processing. As illustrated in

Fig. 4.2, the general "shape" of the message is retained (similarities are highlighted), but much of the noise has been removed. What is left are the same peak data points without all of the irrelevant data.

Fig. 4.2 Pre and Post-Noise Reduction Filtering Waveform



4.2 Legitimate Message Classification

To further illustrate the statistically unbiased nature of the algorithm, we examine a legitimate (nonspam) message. The message used in this example includes noise from mailing list advertisements embedded and noise from informal conversation. In the resulting output, we see the same level of contextual symmetry from the algorithm.

Fig. 4.3 shows us that the noise reduction algorithm perceived inconsistencies this time to be primarily patterns with guilty features which may have otherwise led to a potential misclassification of the message. Unlike the example illustrated in 4.1 (where low-probability tokens were eliminated), the anomalies detected in this legitimate message lend themselves to correct classification by eliminating high-probability tokens. This takes place without the algorithm having any knowledge about the disposition of the message.

Fig. 4.3 Nonspam with common noise, list advertisements

```
<html><body>
<tt><BR>
-hey sassy canadian..I'll do it for ya..just email me.<BR>
I'm at mom's. We got caught in a snowstorm coming home
from <BR>
Susanville..I'm exhausted! lol<BR>
-- In clovergirls@yahoogroups.com, &amp;quot;Chris &amp;amp;
Heather Nish&amp;quot; <BR>
&amp;lt;hcnish@t...&amp;gt; wrote:<BR>
&amp;gt; Hey guys, <BR>
&amp;gt; I need one of you to email someone for me...<BR>
&amp;gt; My emails aren't getting to a potential customer
and<BR>
&amp;gt; now she's starting to get pissy with me...lol<BR>
&amp;gt; any volunteers?<BR>
<BR><BR><BR></tt>
```

```
<!-- |**|begin egp html banner|**| -->

<br><tt><hr width="500">
<b>Yahoo! Groups Links</b><br>
<ul><li>To visit your group on the web, go to:<br><a
href="http://groups.yahoo.com/group/clovergirls/">http://
groups.yahoo.com/group/clovergirls/</a><br> 
<li>To unsubscribe from this group, send an email
to:<br><a href="mailto:clovergirls-
unsubscribe@yahoogroups.com?subject=Unsubscribe">c
lovergirls-
unsubscribe@yahoogroups.com</a><br> 
<li>Your use of Yahoo! Groups is subject to the <a
href="http://docs.yahoo.com/info/terms/">Yahoo! Terms of
Service</a>.
</ul> </tt> </br>

<!-- |**|end egp html banner|**| -->

                    </body></html>
```

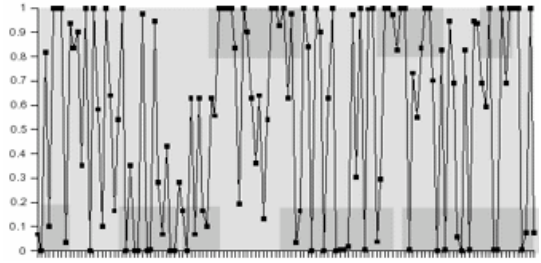
Eliminations:
I'm In com amp gt gt gt need one email for gt emails getting gt now get with gt any your on the from this send an email mailto com subject com nbsp Your use is subject the

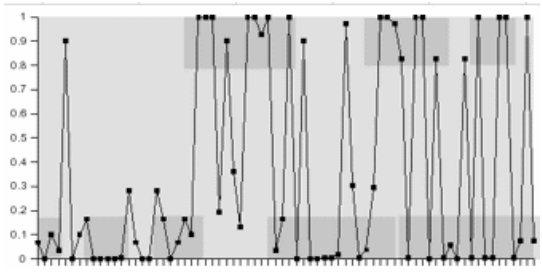
0.82 1.00 1.00 1.00 0.94 0.84 0.35 1.00 1.00 0.58 1.00
0.64 0.54 1.00 0.35 0.98 0.95 0.43 0.63 0.63 0.63 0.56
1.00 0.84 1.00 0.63 0.64 0.54 0.63 0.98 0.84 1.00 0.63
1.00 1.00 1.00 1.00 1.00 1.00 0.73 0.55 0.84 0.70 0.95
0.69 0.95 0.94 0.69 0.59 1.00 0.69 1.00

Remaining Text:
hey sassy I'll ya me mom's caught coming Susanville exhausted! lol clovergirls yahoogroups quot Chris Heather Nish quot lt hcnish wrote Hey guys of you to someone me My aren't to potential customer and she's starting to pissy me lol volunteers Yahoo! Groups Links To visit group web go to href To unsubscribe group to href clovergirls unsubscribe yahoogroups Unsubscribe clovergirls unsubscribe yahoogroups of Yahoo! Groups to href Yahoo! Terms of Service

0.07 0.00 0.10 0.04 0.90 0.00 0.10 0.17 0.00 0.00 0.00
0.00 0.01 0.28 0.07 0.00 0.00 0.28 0.17 0.00 0.07 0.17
0.10 1.00 1.00 1.00 0.20 0.90 0.36 0.13 1.00 1.00 0.93
1.00 0.04 0.17 1.00 0.00 0.90 0.00 0.00 0.01 0.01 0.02
0.97 0.31 0.01 0.04 0.29 1.00 1.00 0.97 0.83 0.01 1.00
1.00 0.00 0.83 0.01 0.06 0.00 0.83 0.01 1.00 0.01 0.01
1.00 1.00 0.01 0.07 1.00 0.08

Fig. 4.4 Pre and Post-Noise Reduction Filtering Waveform





4.3 Measure of Overall Effectiveness

A snapshot of activity was taken of two test subjects considered to have sufficient learning data. Tests were performed evaluating each sample (in this case, email messages) with and without the noise reduction algorithm. Confidence was then calculated using G. Robinson's geometric mean test[6] and inverted (1-P) for nonspam. The following chart shows the affect of the algorithm on classification confidence resulting from application to real-world scenarios.

Fig 4.5 Confidence Delta Measurements

Total	Improved Confidence	Decreased Confidence	Avg delta - Improved	Avg delta - Decreased
2280	1522	16	20.80%	4.00%
5828	3245	56	21.64%	5.39%

The metrics relevant to this detection process are based on strength in the classifier's overall confidence. Since we're not interested only in which samples polarity changes, but also which classifications can be strengthened, samples with an improved confidence were treated as true positives (e.g. successful), while samples with a decreased confidence were considered false positives. The difficulty lies in determining what denotes a false negative. Not all samples contain noise to filter, and so out of the samples that show N/C in confidence, it was appropriate to use samples whose overall confidence levels were low enough to denote a significant level of noise. Two measurements were taken at confidence levels of 65% and 75%, and the average of the two were used in the recall calculation. It was also necessary to take into account the proportional difference in average decrease vs. average increase. Since a decrease in confidence affected a sample's confidence in small proportion to the increase, a variable precision was calculated in addition to the traditional "static" precision of TP/(TP+FP). For example, the first test resulted in an average decrease by only 4% (or one fifth of the affect of a true positive), therefore the variable precision was calculated using TP/(TP+(FP/5)). The second test was considered proportional by a factor of 4, resulting in a variable precision of TP/(TP+(FP/4)).

Fig 4.6 Metrics

FN Avg	Recall	Static Precision	Variable Precision	Static FScore	Variable FScore
127	.9230	.9896	.9979	.9551	.9590
298	.9159	.9830	.9957	.9483	.9541

5.CONCLUSIONS

In the field of language analysis, the root cause of classification errors is an overabundance of data that does not adequately reflect the intended disposition of the sample. In this paper, we've discussed an algorithm to eliminate out of place data to promote better classification. This algorithm attempts to identify noise statistically by placing tokens within a machine-generated context. The noise reduction algorithm allows machine-learning systems to self-contradict their data, providing better overall confidence in the data, which can lead to fewer errors.

An implementation of this algorithm is presently being used in an open source email classifier called DSPAM with much success. The algorithm has already shown to prevent a significant number of erroneous classifications and improve the overall confidence in the results. An implementation is also available in the form of a shared library under the GNU General Public License.

5.1 Other Applications

This algorithm's primary function involves detecting statistical anomalies. While in language classification, these anomalies are undesirable, they may end up being extremely useful data points in other fields of research. The detection facets of this algorithm could be applied to many other fields including:

Steganography detection

Many existing Bayesian algorithms applied to image processing could be used to calculate probabilities of pixel pattern values based on their likelihood to appear together. The algorithm could be applied to identify patterns of pixels which are not likely to appear together and suspiciously out of place. The suspect pixels' least significant bit could be flipped to extract a bit pattern which may contain fragments of the hidden message, at which point a steganalytic attack could be performed.

Economic analysis

In analyzing financial transactions or other similar events, this approach could be applied to detect events which appear out of context. For example, a suspicious purchase of stock suspicious only because it was surrounded in other innocuous activities or vice-versa. Or to cater to the abstract nature of this algorithm, the same scenario could be identified with the purchase of a stock within a certain volume or value with two others of suspicious volume or value.

DNA research

Detecting anomalies and and unusual patterns has always been an interesting area of genetic research. This algorithm could be used to analyze samples of DNA from thousands of different subjects to identify small, unusual patterns seen only among a small group of patients. The algorithm could be used to identify patterns of base pairs which, alone have a fairly

uninteresting disposition, but together are very out of place.

6.ACKNOWLEDGEMENTS

Special thanks to Paul Graham, Bill Yerazunis, and Gary Robinson for their inspiring accomplishments in the field of language classification and for providing much of the base research that has led to the statistical functions this approach utilizes.

7.REFERENCES

[1] Graham-Cumming, J. *How to beat a Bayesian Spam Filter*, MIT Spam Conference 2004 http://www.jgc.org

[2] Graham, P. *So Far, So Good*, August 2003 http://www.paulgraham.com/sofar.html

[3] Graham, P. *Better Bayesian Filtering*, January 2003 http://www.paulgraham.com/better.html

[4] Yerazunis, W. *The Spam Filtering Accuracy Plateau*, MIT Spam Conference 2003 http://crm114.sourceforge.net/Plateau_Paper.pdf

[5] Jackson, C. *Introduction to Artificial Intelligence,* Dover Press 0-486-24864-X

[6] Robinson, G. *Spam Detection*, http://radio.weblogs.com/0101454/stories/2002/09/16/spamDetection.html