Journal of the American Medical Informatics Association, 26(1), 2019, 37–43
doi: 10.1093/jamia/ocy139
Research and Applications

Research and Applications

Cranky comments: detecting clinical decision support malfunctions through free-text override reasons

Skye Aaron,[1] Dustin S McEvoy,[2] Soumi Ray,[1,3] Thu-Trang T. Hickman,[4] and Adam Wright[1,2,3,5]

[1]Division of General Internal Medicine and Primary Care, Brigham and Women's Hospital, Boston, Massachusetts, USA, [2]Partners eCare, Partners HealthCare, Boston, Massachusetts, USA, [3]Department of Medicine, Harvard Medical School, Boston, Massachusetts, USA, [4]Partners Community Health, Partners HealthCare, Boston, Massachusetts, USA, [5]Department of Biomedical Informatics, Harvard Medical School, Boston, Massachusetts, USA

Corresponding Author: Adam Wright, PhD, Brigham and Women's Hospital and Harvard Medical School, 1620 Tremont St., Boston, MA 02115, USA (awright@bwh.harvard.edu)

Received 17 July 2018; Revised 24 September 2018; Editorial Decision 2 October 2018; Accepted 8 October 2018

ABSTRACT

Background: Rule-base clinical decision support alerts are known to malfunction, but tools for discovering malfunctions are limited.
Objective: Investigate whether user override comments can be used to discover malfunctions.
Methods: We manually classified all rules in our database with at least 10 override comments into 3 categories based on a sample of override comments: "broken," "not broken, but could be improved," and "not broken." We used 3 methods (frequency of comments, cranky word list heuristic, and a Naïve Bayes classifier trained on a sample of comments) to automatically rank rules based on features of their override comments. We evaluated each ranking using the manual classification as truth.
Results: Of the rules investigated, 62 were broken, 13 could be improved, and the remaining 45 were not broken. Frequency of comments performed worse than a random ranking, with precision at 20 of 8 and AUC = 0.487. The cranky comments heuristic performed better with precision at 20 of 16 and AUC = 0.723. The Naïve Bayes classifier had precision at 20 of 17 and AUC = 0.738.
Discussion: Override comments uncovered malfunctions in 26% of all rules active in our system. This is a lower bound on total malfunctions and much higher than expected. Even for low-resource organizations, reviewing comments identified by the cranky word list heuristic may be an effective and feasible way of finding broken alerts.
Conclusion: Override comments are a rich data source for finding alerts that are broken or could be improved. If possible, we recommend monitoring all override comments on a regular basis.

Key words: clinical decision support, electronic health records, alerts, override reasons, sentiment analysis

BACKGROUND

Clinical decision support (CDS) has the potential to improve patient safety and quality of care.[1–3] One commonly used subset of CDS is rule-based alerts, which analyze patient data in the electronic health record (EHR) and alert providers when specified conditions are met.[3–5] For example, a CDS alert may notify the provider of a dangerous situation (such as potential drug interactions) or of a useful clinical action that the provider might take (such as prescribing a medication or adding a problem to the problem list). CDS alerts are now used widely in healthcare organizations with uptake increasing over time.[6] Studies investigating the effect of CDS alerts on healthcare quality have had positive but mixed results, suggesting that CDS alerts can improve provider behavior and patient outcomes,[7–9] but often do not.

In particular, there is a growing body of evidence to suggest that CDS systems and alerts are vulnerable to malfunctions.[5,10,11] CDS alerts can malfunction in a variety of ways: an alert might break; the way an alert is designed might not be helpful to users; or an alert might leave out clinically important inclusions or exclusions, causing it to be triggered for the wrong population of patients or providers.[4,12–14] Malfunctioning alerts may pose a safety risk to patients if they encourage providers to take clinical actions that are not appropriate for the patient. Furthermore, flooding providers with clinically inappropriate alerts may cause providers to lose faith in CDS systems and begin to ignore alerts that matter—a phenomenon often referred to as alert fatigue.[15] Additionally, when alerts are not triggered for users who should receive the alerts, there is a lost opportunity to provide a meaningful clinical intervention to patients and, if the user was depending on the alert, an elevated risk for harm.[15,16] Thus, finding and fixing CDS alert malfunctions is an important part of maintaining safe and helpful CDS systems that improve patient safety and quality of care.

In prior work, we developed a taxonomy of CDS alert malfunctions based on 68 cases of alert malfunctions collected from healthcare institutions in the United States.[13] One of the axes of this taxonomy is the method by which a malfunction is discovered. Although the most common discovery method in the sample was user reports via traditional channels such as the help desk, safety reports, and contacting CDS maintenance personnel, 5 malfunctions were discovered by reading free-text override reasons written by providers when overriding alerts.

These malfunctions were discovered at our own institution, by chance, when a student researcher was reviewing alert overrides and noticed that some comments seemed to express strong frustration and disagreement with the alerts. After the discovery, we investigated a sample of rules with a large number of comments seeming to express user frustration to see if there was any merit to the comments, and whether they highlighted an actual issue with the alert. A brief description of 3 illustrative cases and the results of the investigation are provided in Table 1. We found that not only were some of the rules broken, but the comments were helpful for determining why the rules were broken, and helped us fix the rules.

As has been noted in the literature before, users sometimes provide information in override comments or other text fields in the EHR that seem to be communicating information to someone else, but in fact are not sent to the implied recipient and are never acted upon.[17] Override reasons are an example of this kind of text field: although intended for audit purposes, override reasons are typically not intended for communication of clinical information or issues with the CDS itself. Nonetheless, users sometimes provide meaningful information in their override reasons.[18,19] Free-text override reasons have been used in previous research studies to evaluate the clinical appropriateness of user overrides, commonly for medication alerts such as drug–drug interaction checking.[20] A study analyzing the appropriateness of overrides suggested that override reasons could potentially be used to improve alert logic.[20] To our knowledge, this has not become common practice, nor have free-text override reasons been studied directly for their use in the detection of alert malfunctions.

The objective of this study was thus to determine whether free-text override reasons can be systematically used to identify CDS alerts that are malfunctioning. A secondary hypothesis was that features of override comments can be used to identify alerts that are broken.

METHODS

We built our dataset by extracting records for all best practice advisories (BPAs)[1] displayed to users at Partners Healthcare from May 30, 2015, to March 20, 2017, in all settings (both inpatient and outpatient), excluding alerts for test patients (patient profiles in the EHR used for software testing purposes that do not represent real patients). We included the rule identification number (ID), the description of the rule, the date that the rule was triggered, and the override comment if one was given. We excluded rules that received fewer than 10 comments.

In Part 1, we developed a method to systematically review override comments to identify rules that were broken, and we applied this method to classify all rules in the dataset. In Part 2, we applied several ranking methods that attempt to sort broken rules to the top of the list. We used the classification from Part 1 to evaluate the rankings in Part 2.

Methods Part I: rule-level investigation

Each rule was manually classified into one of 3 categories based on a detailed analysis of its logic and a sample of comments: "broken," "not broken, but could be improved," or "not broken."

To begin, a random sample of 50 comments was drawn for each rule. If a rule had fewer than 50 comments, all available comments were included in the sample. A human reviewer (author SA) analyzed the comments for each rule. The reviewer had access both to the back-end Enterprise Data Warehouse consisting of data such as orders, labs, and notes, as well as to the patient's chart in the front-end EHR, and finally to the rule logic available in the training version of the EHR.

Rules were classified as "broken" if there was at least 1 instance in which an alert was triggered contrary to the stated rule logic. Rules were classified as "not broken, but could be improved" if the rule was not classified as "broken" and there was at least 1 instance in which the rule logic had omitted an important piece of clinical data that was in the patient's chart. Rules were classified as "not broken" if none of the alert instances met either criterion. The process is summarized in Figure 1 and described in greater detail in Supplementary Appendix A.

After the comment-level reviews were completed, a second reviewer (author AW) assessed the overall findings for each rule to determine whether he agreed with the classification. When there was disagreement, the 2 reviewers discussed the evidence together to reach consensus about the classification.

Methods Part 2: ranking methods

Since there are many override comments and it may not be possible to review them all, sentiment analysis is a natural place to turn for automatically identifying high-yield comments. Sentiment analysis uses natural language processing (NLP) techniques to extract affective states from text.[21] For example, sentiment analysis can be used to identify angry tweets,[22] positive movie reviews,[23] or popular sentiment towards politicians.[24] The goal of Part 2 was to identify and evaluate methods that use linguistic features of override comments to find rules that are more likely to have had a malfunction. Successful methods for automatically identifying rules with malfunctions may be helpful for organizations with many CDS rules or limited resources to review them.

[1] Best practice advisories (BPAs) are a subset of rule-based CDS alerts at our organization. For the remainder of this paper we will refer to these entities as "rules" and will use "alert" to refer to specific instances in which a rule was triggered and an alert was generated for an EHR user.

Table 1. Three rules are described that had override comments indicating frustration. These rules were investigated further to determine if there was anything wrong with the alerts. Tangible improvement to the alerts was sought when possible

Rule 200002 - Potassium and Digoxin	
Classification	Broken
Alert text	"Patient is taking digoxin and potassium level is low (less than 3.0) or a patient has not had a potassium level in the last 90 days. Consider potassium supplementation and recommend repeating electrolytes."
Representative comments	"BPA misfiring: no potassium on file, but there was a K done!" "Inappropriate warning as K is 4.3"
Investigation	The alert fired when a non-numeric value (eg, "hemolyzed") was stored in the potassium field. Thus, if a patient had non-numeric text stored in the potassium field, the alert would fire even if the patient did not have a potassium level less than 3.0 in the past 90 days.
Resolution	The knowledge management team updated the rule logic so that the rule would be triggered only for patients who had a numeric lab value with $K < 3.0$. After the change, alert volume for the rule fell from 977 alerts per week to 474 per week.
Rule 1005 - Coronary Artery Disease and Beta Blockers:	
Classification	Broken
Alert text	"Patient has CAD-equivalent on problem list and a beta blocker is not on the medication list. Recommend beta blocker."
Representative comments	"He is on beta blocker!" "you are stupid" "this is an inappropriate rec" "On carvedilol!!!!!!!"
Investigation	The alert uses a drug class reference to identify beta blockers. Carvedilol was classified as an alpha/beta blocker, and this class was not included in the rule logic.
Resolution	The knowledge management team had previously updated the rule logic to include alpha/beta blockers such as carvedilol. Several other changes were made to the rule, so it was not possible to assess the effect on the firing rate.
Rule 1852 - Cyclosporine Level Monitoring	
Classification	Not broken, but could be improved
Alert text	"Patient is due for a Cyclosporine Level. Please use the following SmartSet to enter order or go to the Health Maintenance activity to update modifier."
Representative comments	"NOT ON CYCLOSPORINE!!!!!!!!!!!!!!!!!!!!!!!!!!!!!," "cyclosporine is eye drops!," "stupid EPIC reminder-N/A for ophthalmic CyA"
Investigation	The rule did not include the route of administration for cyclosporine orders, so it matched both systemic and ophthalmic preparations, even though ophthalmic administration does not require cyclosporine level monitoring.
Resolution	The knowledge management team updated the rule to exclude ophthalmic cyclosporine orders. There was no significant drop in alert firing rate after the change.

For this section, we combined "broken" and "not broken, but could be improved" into 1 larger category called "malfunction" and compared that to the "not broken" category. We evaluated 3 methods of ranking rules to see whether any of the methods brought rules with malfunctions to the top of the list.

Ranking method 1. Frequency of override comments

In this simple ranking method, rules were ranked by the proportion of alerts that received an override comment (number of comments divided by number of alert firings), with the hypothesis that rules that got comments more frequently may be more likely to exhibit a malfunction.

Ranking method 2. Cranky comments heuristic

We created a preliminary list of words, phrases, and punctuation that seemed likely to indicate frustration or annoyance with an alert. Using this list, we queried override comments containing any features on the list. We then used a snowball method to look for other words and punctuation that appeared in these comments that we felt indicated frustration or brokenness, and added them to the list. We deleted features that brought up many comments that did not indicate frustration or brokenness. The final list is shown in Supplementary Appendix B. For each rule, a ratio was computed by dividing the number of comments for that rule that matched the heuristic by the total number of comments for that rule.

Ranking method 3. Naïve Bayes classifier

A random sample of 10 comments was drawn for each rule. (As in Part I, we excluded rules with fewer than 10 comments available.) A human annotator (author SA) classified each comment into 2 bins, based on whether the comment warrants investigation ("worthy of investigation") or not. The first bin included all cases in which the comment either suggested the rule was broken or clinically inappropriate (that is, any comment that would have warranted investigation in Part I was flagged). The annotator did not have access to any information besides the alert ID, alert description, comment text, and alert date. (Note that although presented later in the text, this process was completed before Part I, to avoid bias in annotating comments when the outcome was already known.)

The Natural Language Toolkit (nltk) package[25] in Python[26] was used to tokenize each comment. Unigrams and bigrams were used as features, and capitalization was removed. The nltk Naïve Bayes classifier was then trained on the sample of annotated override comments. Using the classifier trained on the annotated comments, all remaining comments were classified as "worthy of investigation" or not, based on which category was more likely, given the features of the comment. For each rule, a ratio was computed by dividing the number of comments for that rule that the Naïve Bayes classifier flagged as "worthy of investigation" by the total number of comments for that rule. A 10-fold cross-validation was performed, with the average of scores used for the final score. Rules were ranked by their final score.

Evaluation of ranking methods

The ranking methods were compared using 3 performance metrics:

1. The median ranking of rules without malfunctions was compared to the median ranking of rules with malfunctions.

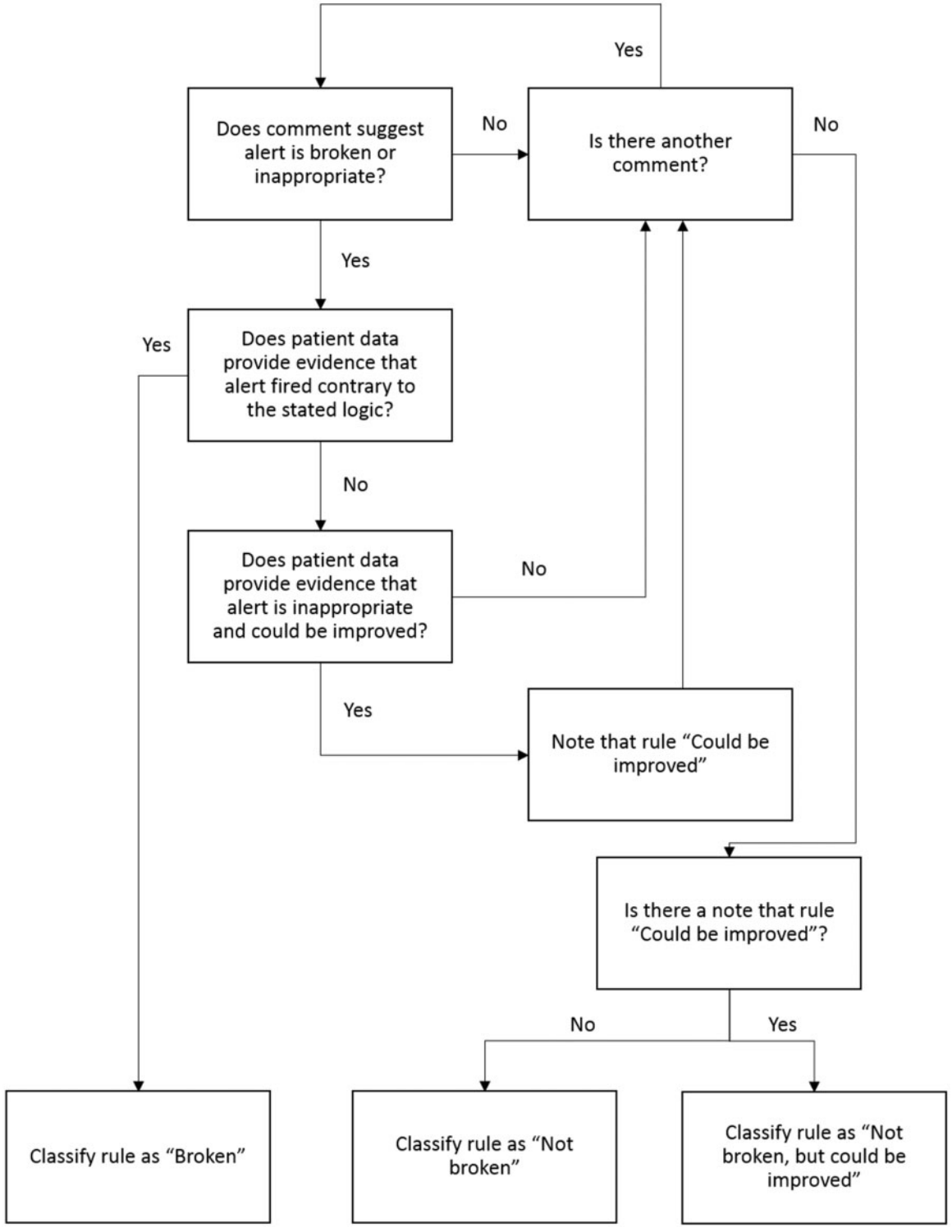

Figure 1. Procedure for classifying rules based on comments.

Statistical significance for the comparison was calculated using the Mann–Whitney U test.

2. Precision at 10, 20, and 30 was calculated.
3. The receiver operating characteristic (ROC) curve was plotted and the area under the curve (AUC) calculated.

The performance evaluations were conducted using the stats,[27] pROC,[28] and ggplot2[29] packages in R.[27]

RESULTS

Results Part 1: rule level investigation

There were 282 rules that triggered an alert shown to a user at least 1 time during the study period. Of these, 120 rules had at least 10 comments, and thus qualified for inclusion in the dataset, with a total of 36 369 associated free-text override comments. For the purposes of the rule level investigation, 4868 comments were sampled.

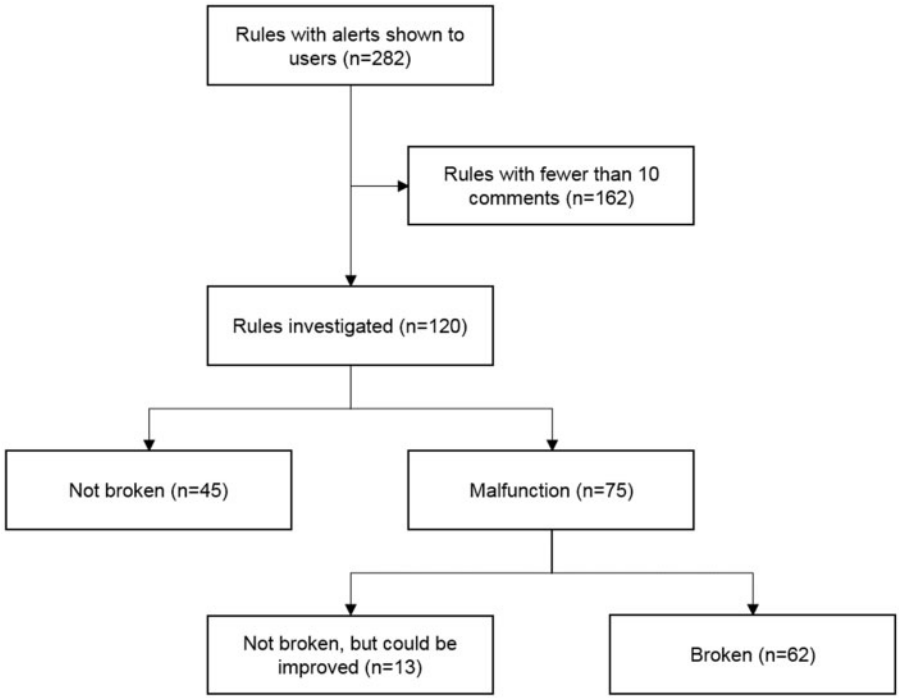

Figure 2. Results of rule-level classification. 120 rules qualified for the rule-level investigation and were sorted into 2 categories and 2 subcategories. 62.5% of rules investigated had some kind of malfunction.

As summarized in Figure 2, 62 of the rules were classified as "broken," 13 were classified as "not broken, but could be improved," and the remaining 45 were classified as "not broken." Thus, reviewing the sample of override comments led to the identification of 75 rules with a malfunction (were broken or could be improved), comprising 62.5% of rules with at least 10 comments (and 26.6% of all rules in the EHR).

Results Part 2: ranking methods

Each method for ranking rules assigns a continuous value from 0 to 1 to each rule, and then sorts the rules based on their score. Since there were 120 rules in the dataset, the rankings range from 1 to 120, where a lower rank corresponds to a higher score. Rankings for ties were averaged. We hypothesized that rules with malfunctions would have higher scores and thus a lower ranking.

Three metrics were calculated for each ranking. Precision at n reports the number of rules that were actually broken out of the top n rules. Since there are more rules with malfunctions than rules without, we would expect precision at 20 to be good for a baseline random ranking; on average 12.3 out of the top 20 rules will have a malfunction. However, there should be no statistically significant difference on average between the median rankings for rules with malfunctions and rules without. The Mann–Whitney U test was used to test for significance, with group sizes 75 and 45. Finally, the AUC reports the AUC of the ROC curve, shown in Figure 3. Random rankings will converge to an AUC of 0.5, whereas a perfect ranking (all rules with malfunctions are sorted ahead of all rules without malfunctions) would have an AUC of 1.

The performance of each method is compared in Table 2. In brief, sorting by the Frequency of Override Comments performed worse than a baseline random sort would have. Sorting by the Cranky Comments Heuristic performed well. Sorting by the Naïve Bayes Classifier performed best.

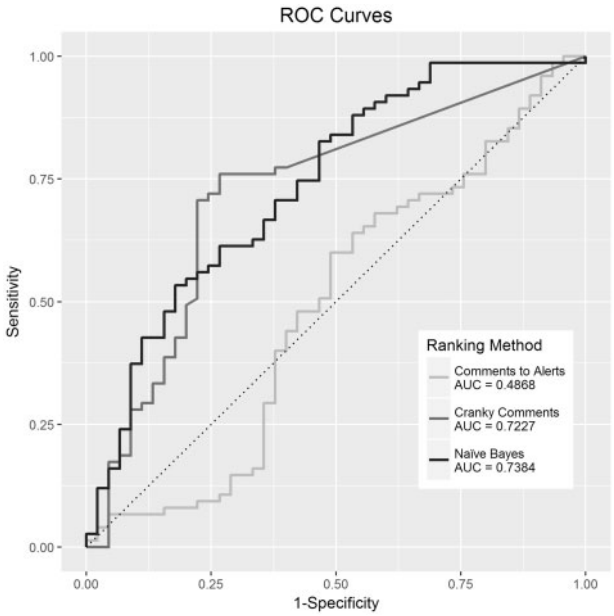

Figure 3. ROC curves and AUC for each of the 3 ranking methods.

DISCUSSION

We found a substantial number of rules at our institution that were broken or could be improved by examining override comments: almost two-thirds of rules with at least 10 comments fit into 1 of these categories. This represents 27% of all rules that were active during the study period, which is much higher than expected and unacceptable from a safety perspective.

Of the 3 sentiment analysis methods evaluated for their use in ranking rules, 2 performed better than a random ranking, and 1 did

Table 2. Summary statistics of 3 methods for ranking rules. Each method assigns a continuous score of 0 to 1 to each rule, and then ranks rules based on their score. Lower ranks correspond to a higher score

	Frequency of Override Comments	Cranky Comments Heuristic	Naïve Bayes Classifier
Median rank of rules with malfunctions	56.5	47.5	46
Median rank of rules without malfunctions	71	98.5*	86*
Mann–Whitney U statistic	1732	936	883
Precision at 10	5/10	8/10	9/10
Precision at 20	8/20	16/20	17/20
Precision at 30	14/30	24/30	26/30
Area under the receiver operating characteristic curve (AUC)	0.487	0.723	0.738

*Median rank of rules with malfunctions is different from median ranking of rules without malfunctions with $P < .0001$

not. The frequency of override comments out of total alert instances was not a good indicator of malfunctions. However, both the cranky comments heuristic and the Naïve Bayes classifier performed well—investigating the top 20 rules selected by each method would have yielded 16 and 17 true malfunctions, respectively. Annotating the training set for Naïve Bayes required substantially more work than the cranky comments heuristic, with little improvement from a practical standpoint. Given the very short length of comments (25 characters on average), our organization decided not to continue with the classifier approach, since a daily manual review of all comments is feasible.

Our results suggest that user feedback provided through override comments has been an underutilized but valuable data source for improving CDS. For organizations that can afford to do so, reading all the override comments may be worthwhile. At our own organization, we now send a daily email to the CDS knowledge management team with all override comments from the previous 24 hours and cranky comments highlighted (as determined by the cranky word list). We also recommend putting contact links within alerts so that users who would like to provide feedback are encouraged to do so. Giving users channels to provide feedback may increase the likelihood that they do so.

One strength of this methodology is that it can be used to find rules that have malfunctioned since inception. Other methods for detecting malfunctions that we have prototyped rely on changes in the alert firing rate, and thus cannot be used to identify rules that have a stable firing rate but are nonetheless broken. In contrast, override comments can be used to find malfunctions even when the rule has a consistent firing rate. When it is possible for an organization to do so, we recommend using multiple monitoring strategies that complement each other to identify as many malfunctions as possible. At our institution, in addition to the daily review of override comments, we also monitor the firing rate of rules, changes to rule logic, and changes to underlying codes used in rule logic.

One limitation is that the study did not examine changes in classification over time. When changes are made to rule logic, the EHR system is upgraded, or underlying codes such as medication codes and lab codes are revised, it is possible that rules that were previously broken can begin to function correctly, and vice versa. Another limitation of the study is that it provides only a lower bound on the malfunction rate at our institution—rules that did not receive comments or that did not have comments indicating a malfunction were not investigated and may be broken. Additionally, the 2 successful methods for ranking rules predict which rules were likely to be investigated, which predicts malfunctions only by proxy. Finally, the methods may not be generalizable to other EHRs or institutions—the methodology requires that override comments can be retrieved and that users are providing useful information in their comments—conditions that may not be met at all institutions.

CONCLUSION

This study develops a novel method for detecting CDS malfunctions and contributes to the growing body of evidence that malfunctions in CDS are common and often go undetected. Analyzing override comments was an effective strategy for detecting malfunctions. Furthermore, in several cases, comments provided a guide to diagnosing how rules were broken and could be fixed.

The specific methods used to analyze override comments can be adapted to suit organizations with different resources and monitoring needs. A simple heuristic list of words that expressed frustration or brokenness was effective at identifying broken alerts with high precision. This is a low-cost method of finding malfunctions that could be useful even for organizations with relatively few resources to devote to the maintenance of CDS. For organizations with more resources to devote to monitoring CDS, analysis of override reasons can be made more comprehensive and/or sophisticated. When it is possible and seems fruitful to do so, we recommend that CDS knowledge management personnel review *all* override comments on a daily basis, and we have recently deployed this approach at our own organization.

Finally, the methods developed in this study may be useful for identifying CDS malfunctions that cannot be found via other means. When possible, we recommend using multiple malfunction detection techniques that complement each other.

FUNDING

Research reported in this publication was supported by the National Library of Medicine of the National Institutes of Health under Award Number R01LM011966. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

Conflict of interest statement. The authors do not have any competing interests.

CONTRIBUTORS

AW is responsible for study concept and design. SA, DM, and AW conducted rule investigation and classification. SA conducted the data analysis. SA wrote the manuscript. SA, DM, SR, TH, and AW provided critical revisions of the manuscript for important intellectual content.

ACKNOWLEDGMENTS

We would like to thank student researcher Dinesh Banjara for discovering that override comments may be a rich data source for improving alerts.

SUPPLEMENTARY MATERIAL

Supplementary material is available at *Journal of the American Medical Informatics Association* online.

REFERENCES

1. Teich JM, Merchia PR, Schmiz JL, *et al*. Effects of computerized physician order entry on prescribing practices. *Arch Intern Med* 2000; 160 (18): 2741–7.
2. Evans RS, Pestotnik SL, Classen DC, *et al*. A computer-assisted management program for antibiotics and other antiinfective agents. *N Engl J Med* 1998; 338 (4): 232–8.
3. Byrne C, Sherry D, Mercincavage L, *et al*. Advancing clinical decision support: key lessons in clinical decision support implementation. Technical Report. Prepared for Office of the National Coordinator ARRA Contract, Task No. HHSP23337009T; 2011.
4. Krall M, Gerace A. a metrics taxonomy and reporting strategy for rule-based alerts. *Perm J* 2015; 19 (3): 11–9.
5. McCoy AB, Thomas EJ, Krousel-Wood M, *et al*. Clinical decision support alert appropriateness: a review and proposal for improvement. *Ochsner J* 2014; 14 (2): 195–202.
6. Charles D, Gabriel M, Searcy T. Adoption of electronic health record systems among U.S. non-federal acute care hospitals: 2008–2014. *ONC Data Brief* 2015; 23: 1–10.
7. Page N, Baysari MT, Westbrook JI. A systematic review of the effectiveness of interruptive medication prescribing alerts in hospital CPOE systems to change prescriber behavior and improve patient safety. *Int J Med Inform* 2017; 105: 22–30.
8. Pell JM, Cheung D, Jones MA, *et al*. Don't fuel the fire: decreasing intravenous haloperidol use in high risk patients via a customized electronic alert. *J Am Med Inform Assoc* 2014; 21 (6): 1109.
9. Rind DM, Safran C, Phillips RS, *et al*. Effect of computer-based alerts on the treatment and outcomes of hospitalized patients. *Arch Intern Med* 1994; 154 (13): 1511–7.
10. Wright A, Hickman TT, McEvoy D, *et al*. Analysis of clinical decision support system malfunctions: a case series and survey. *J Am Med Inform Assoc* 2016; 23 (6): 1068–76.
11. Kassakian SZ, Yackel TR, Gorman PN, *et al*. Clinical decisions support malfunctions in a commercial electronic health record. *Appl Clin Inform* 2017; 08 (03): 910.
12. Shah NR, Seger AC, Seger DL, *et al*. Improving acceptance of computerized prescribing alerts in ambulatory care. *J Am Med Inform Assoc* 2006; 13 (1): 5–11.
13. Wright A, Ai A, Ash J, *et al*. Clinical decision support alert malfunctions: analysis and empirically derived taxonomy. *J Am Med Inform Assoc* 2018; 25: 496–506.
14. McCoy AB, Waitman LR, Lewis JB, *et al*. A framework for evaluating the appropriateness of clinical decision support alerts and responses. *J Am Med Inform Assoc* 2012; 19 (3): 346–52.
15. van der Sijs H, Aarts J, van Gelder T, *et al*. Turning off frequently overridden drug alerts: limited opportunities for doing it safely. *J Am Med Inform Assoc* 2008; 15 (4): 439–48.
16. van der Sijs H, Kowlesar R, Aarts J, *et al*. Unintended consequences of reducing QT-alert overload in a computerized physician order entry system. *Eur J Clin Pharmacol* 2009; 65 (9): 919–25.
17. Chused AE, Kuperman GJ, Stetson PD. Alert override reasons: a failure to communicate. *AMIA Annu Symp Proc* 2008 Nov 6; 111–5.
18. Ahn EK, Cho SY, Shin D, *et al*. Differences of reasons for alert overrides on contraindicated co-prescriptions by admitting department. *Healthc Inform Res* 2014; 20 (4): 280–7.
19. Seidling HM, Paterno MD, Haefeli WE, *et al*. Coded entry versus free-text and alert overrides: what you get depends on how you ask. *Int J Med Inform* 2010; 79 (11): 792–6.
20. Nanji KC, Slight SP, Seger DL, *et al*. Overrides of medication-related clinical decision support alerts in outpatients. *J Am Med Inform Assoc* 2014; 21 (3): 487–91.
21. Kouloumpis E, Wilson T, Moore J. Twitter sentiment analysis: the good the bad and the OMG! In: Fifth International AAAI Conference on Weblogs and Social Media; 2011; Barcelona, Spain.
22. Bollen J, Mao H, Pepe A. Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena. In: Fifth International AAAI Conference on Weblogs and Social Media; 2011; Barcelona, Catalonia, Spain: Association for the Advancement of Artificial Intelligence.
23. Thet TT, Na J-C, Khoo CSG. Aspect-based sentiment analysis of movie reviews on discussion boards. *J Inf Sci* 2010; 36 (6): 823–48.
24. Wang H, Can D, Kazemzadeh A, *et al*. A system for real-time Twitter sentiment analysis of 2012 U.S. presidential election cycle. In: *ACL 2012 System Demonstrations*; 2012; Jeju Island, Korea: Association for Computational Linguistics.
25. Bird S, Loper E, Klein E. *Natural Language Processing with Python*. O'Reilly Media, Inc. 2009. http://nltk.org/book. Accessed November 12, 2018.
26. Python Software Foundation. Python 3.6.4 documentation. 2018. https://docs.python.org/3/. Accessed January 29, 2018.
27. R Core Team. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing; 2017.
28. Robin X, Turck N, Hainard A, *et al*. pROC: an open-source package for R and S+ to analyze and compare ROC curves. *BMC Bioinformatics* 2011; 12: 77.
29. Wickham H. *ggplot2: Elegant Graphics for Data Analysis*. New York, NY: Springer-Verlag; 2009.