Lessons for Supporting Data Science from the Everyday Automation Experience of Spell-Checkers

Kevin Crowston
Syracuse University School of Information Studies
Syracuse, NY 13244, USA
crowston@syr.edu

Workshop proceedings *Automation Experience across Domains*
In conjunction with CHI'20, April 26th, 2020, Honolulu, HI, USA
Website: http://everyday-automation.tech-experience.at

Abstract
We apply two theoretical frameworks to analyze spell-checkers as a form of automation and apply the lessons learned to analyze opportunities to support data science. The analysis distinguishes between automation of analysis to suggest actions and automation of implementation of actions. Having the automation work in the same space as users (e.g., editing the same document) supports stigmergic coordination between the two, but attention is needed to ensure that the contributions can be combined and have a recognizable form that indicates their purpose.

Author Keywords
automation, spell-checking

CCS Concepts
•**Social and professional topics** $\rightarrow$ **Automation;** •**Human-centered computing** $\rightarrow$ *Interaction design theory, concepts and paradigms;* •**Applied computing** $\rightarrow$ *Word processors;*

Introduction
A form of automation (i.e., the capability of a system to perform some tasks without human involvement) experienced by many people daily is the spell-checker, which has evolved from a stand-alone application providing suggested corrections [3, 6] to an integral component of word proces-

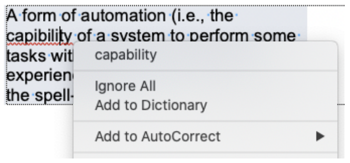
A form of automation (i.e., the capibility of a system to perform some tasks wit experien the spell-

Figure 1: A spelling mistake identified by the Microsoft Word spell-checker and a proposed replacement

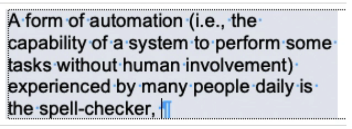
A form of automation (i.e., the capability of a system to perform some tasks without human involvement) experienced by many people daily is the spell-checker, ¶

Figure 2: A spelling mistake automatically corrected by the Microsoft Word spell-checker*

* Note: animation works in Adobe Reader but not in some other PDF readers.

sors or even a ubiquitous component of a user interface framework [4]. As a user types, automated spell-checkers flag unknown words as likely errors, offer suggested replacements (see Fig. 1) or even make replacements without human involvement (see Fig. 2). In this position statement, we analyze the nature of automation provided by spell-checkers to derive lessons for ubiquitous automation in other settings, specifically, data science.

Theory

We apply two frameworks for our analysis. First, we apply a simple framework developed in Ref [1]. This framework decomposes information processing tasks into four steps: 1) information acquisition; 2) information analysis; 3) decision and action selection; and 4) action implementation. By considering if each step can be partly or fully automated (meaning that the particular step can be done by a system without human intervention), the framework identifies four levels of automation:

0. No automation
1. Decision support: steps 1 and 2 are automated but in step 3, the system recommends possible actions from which the human chooses one to implement
2. Blended decision making: all steps are automated but only for a subset of decisions
3. Complete automation

Second, the workshop call identifies four key aspects of ubiquitous automated systems: intelligibility, interventions, interplay and integrity. In this position statement, we focus on the first two: how can a human tell what the system is doing and intervene if desired? To analyze these issues, we apply theorizing about stigmergic coordination, meaning coordination through a shared work product rather than through separate communication. Ref [2] identifies three socio-technical affordances needed to support stigmergic coordination, namely visibility and combinability of work of recognizable genres. Visibility means that work done by one contributor is visible to others. Combinability means that different contributions can be made to fit together, as has been observed to be important for open source software development [5]. Genre means that the contributed work has socially-recognized regularities of form and purpose that enable others to know how they should work with it. The analysis in Ref [2] focuses on supporting coordination between members of a work team but these features may also support coordination between a system and a user.

Results

Applying the first framework, spell-checking systems initially were decision support systems (level 1), flagging unrecognized words and giving a list of possible replacements when requested. Currently, many support blended decision making (level 2), automatically fixing (or at least changing) some detected errors while deferring other to the user. However, given the variability of typing errors, it seems unlikely that spell-checking will ever be completely automated.

Considering next questions of intelligibility, a spell-checker's suggestions in current systems are visible because the system is integrated with the work it is meant to support so that the intervention happens in the same space as the work. In other words, the interaction between the system and the user is stigmergically coordinated. The users' typing in a document triggers the actions of the spell-checker and the spell-checker offers suggestions to the user or takes actions independently in the same interface, thus making the actions visible. Interestingly, spell-checkers don't show certainty of their suggestions, though it might be implicit in the ordering of suggestions. For spell-checking, the other two affordances needed for stigmergic coordination, combinability and genre of contributions, are non-issues, as words are easily combined and have a clear form and purpose.

Finally, considering opportunities for intervention, a user can intervene in the work of the spell-checker by interacting with it in the document. Most spell-checkers can be customized by correcting the corrections made or adding to the dictionary. However, further tuning is not possible, e.g., being able to tune how confident the system should be of a correction before it is automatically implemented.

Discussion

We next consider how the observations about spell-checking might be transferred to a more complex task. We will consider in particular the task of data analysis, i.e., writing a data-science-analysis script. A spell-checker for a data analysis could be exactly the same as for word processing, e.g., correcting a misspelled function or variable name or incorrect arguments. More interestingly, an automated system could check the data analysis at a higher level. A system could assess data quality, e.g., spotting outliers or problems with missing data, suggesting transformations to correct skew or more ambitiously, noticing bias in the data. It could create additional data columns, e.g., breaking up complex data into components or finding related datasets and joining them. Finally, a system could suggest additional actions for an analysis, e.g., suggesting useful visualizations or modelling approaches given what it knows about the data or diagnostics for a user-selected analysis. If the assumptions of a test are violated, it could suggest an alternative, e.g., a non-parametric test instead of a parametric one.

Our analysis of spell-checkers suggests some design implications for such a system. First, there are different levels of functionality: at the lowest level of automation, the system would simply flag issues and suggest possibilities to the user while at a higher level, it would automatically execute some actions (e.g., automatically checking test assumptions). And as before, completely automated analysis seems unlikely.

Second, intelligibility would be increased by having the system work in the same space as the users to support stigmergic coordination, e.g., in the same notebook if the analyst is using a notebook. Spell-checking words would work the same way as in word processor, while interventions in the process could be done by creating a note on notebook cell with suggested changes or creating additional cells, e.g., the cells to run and interpret diagnostics for an analysis or to create a visualization. The system could communicate intent or certainty by adding comments to the code. Finally, if the system intervenes by providing code to run, the user could edit the code if not appropriate.

Third, the work on stigmergic coordination suggests two additional affordances needed to support stigmergic coordination, in addition to visibility. The first is combinability, meaning that the work done by different contributors can be easily fitted together. In the case of data science, a notebook provides a mechanisms for combinability, as different contributors can add different cells. To make cells function smoothly together does require some additional work, e.g., identifying which variables hold the necessary data.

The second factor is genre, meaning socially recognized regularities of form and purpose. For a user to be able to use suggestions made by an automated system, they need to be able to recognize what those contributions do and how to use them. Applied to data science analyses, the theory suggests that there is a need for the user to be able to recognize the purpose of a suggested analysis. Such recognition could be explicitly supported, e.g., by commenting in the code.

Conclusion

The analysis offers two general takeaways for future design. First, automation can happen at different levels and in different ways. We distinguish in particular between automation of analysis to suggest actions and automation of implementation of actions. Second, having the system work in the same space as the users supports stigmergic coordination between the two. However, additional affordances, namely combinability and genre are necessary to support this mode of coordination.

REFERENCES

[1] Kevin Crowston and Francesco Bolici. 2019. Impacts of machine learning on work. In *Hawai'i International Conference on System Sciences (HICSS–52)*. http://hdl.handle.net/10125/60031

[2] Kevin Crowston, Jeff S. Saltz, Amira Rezgui, Yatish Hegde, and Sangseok You. 2019. Socio-Technical Affordances for Stigmergic Coordination Implemented in MIDST, a Tool for Data-Science Teams. *Proc. ACM Hum.-Comput. Interact.* 3, CSCW, Article Article 117 (Nov. 2019), 25 pages. DOI: http://dx.doi.org/10.1145/3359219

[3] Fred J. Damerau. 1964. A technique for computer detection and correction of spelling errors. *Commun. ACM* 7, 3 (1964), 171–176. DOI: http://dx.doi.org/10.1145/363958.363994

[4] Ivor Durham, David A. Lamb, and James B. Saxe. 1983. Spelling correction in user interfaces. *Commun. ACM* 26, 10 (1983), 764–773. DOI: http://dx.doi.org/10.1145/358413.358426

[5] James Howison and Kevin Crowston. 2014. Collaboration through superposition: How the IT artifact as an object of collaboration affords technical interdependence without organizational interdependence. *MIS Quarterly* 38 (3/2104 2014), 29–50. DOI: http://dx.doi.org/10.25300/MISQ/2014/38.1.02

[6] James L. Peterson. 1980. Computer programs for detecting and correcting spelling errors. *Commun. ACM* 23, 12 (1980), 676–687. DOI: http://dx.doi.org/10.1145/359038.359041