Use What You Choose: Applying Computational Methods to Genre Studies in Technical Communication

Brian Larson
Georgia Institute of Technology
781 Marietta Street NW Atlanta,
Georgia 30318
blarson@gatech.edu

William Hart-Davidson
Michigan State University
235 Bessey Hall
East Lansing, MI 48824
hartdav2@msu.edu

Kenneth C. Walker
University of Texas, San Antonio
One UTSA Circle
San Antonio, TX 78249
kenneth.walker@utsa.edu

Douglas Walls
North Carolina State University
221 Tomkins Hall
Raleigh, NC 27695
dwalls@ncsu.edu

Ryan Omizo
University of Rhode Island
90 Lower College Road
Kingston, RI 02881
rmomizo@uri.edu

ABSTRACT

This paper reports on the results of an intensive application development workshop held in the summer of 2015 during which a group of thirteen researchers came together to explore the use of machine-learning algorithms in technical communication. To do this we analyzed Amazon.com consumer electronic product customer reviews to reevaluate a central concept in North American Genre Theory: stable genre structures arise from recurring social actions ([1][2][3][4][5]). We discovered evidence of genre hybridity in the signals of instructional genres embedded into customer reviews. Our paper discusses the creation of a prototype web application, "Use What You Choose" (UWYC), which sorts the natural language text of Amazon reviews into two categories: instructionally-weighed reviews (e.g., reviews that contain operational information about products) and non-instructionally-weighed reviews (those that evaluate the quality of the product). Our results contribute to rhetorical genre theory and offer ideas on applying genre theory to inform application design for users of information services.

CCS Concepts

• Human-Centered Computing□Collaborative and social computing theory, concepts and paradigms□Collaborative content creation

SIGDOC'16, September 21 – 23, 2016, Arlington, VA, USA.

ACM 978-1-4503-4495-1/16/09…$15.00
DOI: http://dx.doi.org/10.1145/2987592.2987603

Keywords

Genre Theory; Technical Communication; Natural Language Processing; Web Application Design, User Experience

1. INTRODUCTION

1.1 Testing Genre Theory with an Exploratory, Proof of Concept Development Project

The project we report on here was an exploratory one, designed to test an idea about the way genres of written discourse form. Specifically, we sought to tease out "genre hybridity," the notion that some genres will exhibit formal features associated with more than one written genre. To do this, we worked in three teams to build and test a machine-learning classifier designed to detect and extract *instructional* texts from customer reviews in the [6] SNAP (Amazon.com) text corpus (see also [7]). We measured the outcome against a human-coded sample from the same corpus, using qualitative text analysis and reliability measurement techniques. Our goal was to build a mockup for a useful web service – one that could either stand alone or be incorporated into a site like Amazon.com – that could sort the natural language text of reviews into two categories: instructionally-weighted reviews (e.g., reviews that contain operational information about products) and non-instructionally-weighted reviews (those that evaluate the quality of the product).

Doing so, we sought to do a real-world evaluation of rhetorical genre theory. If the classifier was successful, we would see the theory of genre-hybridity put to use in a proof-of-concept manner that would lend credibility to theory and, more broadly, to the prospects of building other text analysis services with a basis in rhetorical theories.

Our paper begins with a very brief framing section that positions our project in relation to a few key ideas from technical communication and genre theory, in particular. The paper then describes the work of our three scrum development teams (UX, Research, and Development). All three teams followed a mixed methods approach [8] to fabricate the UWYC prototype. The

Research team developed a binary categorical coding scheme, performing a conceptual content analysis [9] to classify each sentence in the corpus for the presence of instructional content. The UX team developed two personae and accompanying user stories for the project, shared these with the other teams for feedback, and then moved on to design and implement a simple input view and results view for the web application. The Development Team converted the human-coded samples into a machine-readable training set that would inform a support vector machine classifier [10] [11]. This classifier annotates sentences from test data to classify reviews as "instructional" or "non-instructional." The development team proceeded to train and test the classifier to improve its performance.

In the results and implications section, we discuss two key outcomes of our project: (1) the viability of using machine learning to parse natural language texts for higher order rhetorical concerns; and (2) the effectiveness of our scrum-based team model for conducting exploratory research. Finally, we look ahead to the future of this project and the potential for empirical validation of a central concept in rhetorical genre theory: genre hybridity.

1.2 Searching for Genre Hybridity in Open Systems

Selber [12] notes that in open systems of instructional content that "encourage users to become authors and editors of instruction sets" a significant amount of variation and, accordingly, communicative richness and complexity arises due to the relatively "organic nature of the open web" (107-8). As a result, innovations in these systems depend, according to Selber, "less on inventing novel capabilities and more on constructing philosophies and practices that are sympathetic to the communicative nature of open instruction sets" (110-11). Miller [2] & Schryer [1] provide the foundational claim upon which Selber can build his own argument: recurring social situations create the need for what, over time, become stable genre structures. We could accordingly expect in an open system such as Amazon.com that has many people writing with only a few enforced structural components that we would see features of review texts arising that serve users' immediate use-related purposes.

Skalicky [13] studied a sample of Amazon product reviews rated as "most helpful" by users of the service and found that those containing "experience" information – accounts of customers using the product – were rated higher than those that did not. By contrast, reviews that appeared to be "soft sell" or overly persuasive were not rated highly. Our team read these results as an indicator of genre hybridity in line with the prediction stated earlier. We reasoned that experience information – and even instructional information about how to use or get the best results from a product – would be seen as valuable by readers of reviews because it could help customers see beyond the initial moment of adoption to learn about what their own use experience might be like.

As rhetoric scholars, we saw an opportunity to take advantage of the relative stability of instructional information – itself a well known genre – and to harvest it from another genre - customer reviews. The result could be a proof of concept to demonstrate possibilities presented by genre hybridity if we could reliably distinguish between instructional and persuasive elements of the customer reviews.

As a service, Use What You Choose (UWYC), sought to draw on the collective knowledge of customers and product owners as reflected in Amazon product reviews. The service gathers online customer reviews, subtracts the persuasive content, and harvests instructional information about the use experience of product owners. We came to think of it like a crowdsourced version of *Consumer Reports*. For would-be customers, the service could provide information about what it is like to own and use a product. For those using Amazon's product reviews as they currently are presented, instructional information may be difficult to locate because it may or may not be present in many or most of reviews and, in any case, it is intermingled with other kinds of information.

2. METHOD

2.1 Refining a Coding Scheme, Working With the Corpus: The Research Team

During each of three sprints, the research team (RT) worked toward a practical coding scheme while responding to evolving requirements from the Development Team (DT) and User Experience Team (UXT). The RT developed a binary categorical coding scheme, performing a conceptual content analysis [10] to classify each sentence in the corpus for the presence of instructional content. During development, we did not attempt to assess the guide's reliability or validity according to standard measures ([15] [16]), choosing instead to rely on the efficacy of the resulting machine-learning model as a test of the coding scheme's validity.

Our account first follows the work of the RT & UXT through the three sprints. We then circle back to explain the work of the DT before turning to results and implications.

2.2 Sprint 1: User Stories & Coding Categories

The RT consisted of six members, who began examining the data and considering options for developing a coding guide. Because the DT had tokenized the data into sentences and presented them to the RT in the form of an Excel spreadsheet with one sentence per row, it seemed practical to use the sentences as units of analysis. Meanwhile, in order to index our coding scheme to user experiences, the UXT team provided the following user stories shortly after the beginning of the sprint:

> User Story 1, version 1: As an online shopper, I want to hear how others have experienced the product I have or am about to purchase in order to understand 1) what tips or advice others may have for effective use, 2) what alerts others may offer to unwanted outcomes.
>
> User Story 2: As a scholar of rhetoric & technical communication, I am interested in harvesting the useful instructional information about technology use to better understand how knowledge about technology is created and shared.

With these user stories as a guide, the RT broke into three teams of two, each of which took approximately 150 units from the Electronics 1 file. After an inductive process of reading and coding the sentences silently, each pair talked together about what typical patterns they were seeing, and then the whole team came together to discuss their findings. As a consequence of the first round of coding, the RT developed the first version of its coding scheme. We agreed to seek units where:

Coding Scheme, version 1:

> Author describes an action relating to the use of the product that may or may not be taken and in that unit or an adjacent unit the author either (a) describes the consequence of taking or not taking that action or (b) describes a problem that the action remedies. This should exclude descriptions relating to other products but not previous versions of this product.

After discussing issues of context, audience, and purpose for these products, we recognized that using the sentence as our unit of analysis was problematic, because in many cases, the consequences of an action described in one unit were described in the previous or following unit. To satisfy the first user story, we realized we needed to connect "tips" and "advice" to consequences. Consequently, our coding scheme called on the coder to consider adjacent units. The resulting scheme seemed to capture the information necessary to respond to the user stories that the UXT articulated.

2.3 Sprint 2: System Requirements & A Coding Guide

By early in the second sprint, the UXT had revised User Story 1 to reflect the emergent features in the reviews. The more general categories of "tips or advice" were now more readily distinguished as "hints" and "hacks":

> User Story 1, version 2: As an online shopper, I want to hear how others have experienced the product I have or am about to purchase in order to understand 1) what ~~tips or advice~~ hints and/or hacks others may have for effective use, 2) what alerts others may offer to unwanted outcomes.

The UXT also offered the following statement of system requirements pertaining to User Story 1: Given an Amazon URL or product name:

1. Analyze all the sentences in customer-supplied reviews of the product
2. Find reviews that offer helpful instructional information
3. Present a sortable list of results that includes
    a. An excerpt of hint/hack/alert information as a preview
    b. Icons to indicate that a review contains hint/hack/alert
    c. Helpfulness score (from Amazon.com)
4. Allow the end-user to select and expand the review to read the full version

These changes and the resulting requirements did not alter the objectives of the RT. The goal of the RT at this point was to provide to the DT a large enough corpus of coded units to permit the DT to train a machine-learning classifier to identify those reviews that included instructional information. The RT thus formalized the coding scheme to permit it to identify units with a binary designator:

Coding Scheme, version 2

1. Mark as "1" any unit where
    a. the author describes an action relating to the use of the product that may or may not be taken AND
    b. in that unit or an adjacent unit the author
        i. describes the consequence of taking or not taking that action OR
        ii. describes a problem that the action remedies.
    c. This should exclude descriptions relating to other products but not previous versions of this product.
2. Mark anything else as "0".

This formulation of the coding scheme was easier for coders to interpret and apply than version 1 because of the bracketed conjunctional and disjunctional conditions. It also functioned to clarify that the codes were mutually exclusive and collectively exhaustive: every unit should be coded in exactly one state – "0" or "1." During this sprint, the RT continued coding using the same process as in the first sprint, but this time, their purpose was to reach a higher degree of confidence in the coding scheme and to identify examples of the coded units to use in combination with the coding scheme as a coding guide.

2.4 Sprint 3: Prototype Views & Reliable Raters

At this point, the UXT reaffirmed the user stories and statement of requirements from the second sprint. By this time, the coding scheme took the following form:

1. Mark as "1" any unit where
    a. the author describes or implies an action in that unit that the author took relating to the use of the product ~~that may or may not be taken~~ AND
    b. the action is described or implied in such a way that it could mediate the interaction of a reader of the review with the product AND
    c. in that unit or an adjacent unit the author
        i. describes the consequence of taking or not taking that action OR
        ii. describes a problem that the action remedies.
    ~~d. This should exclude descriptions relating to other products but not previous versions of this product.~~
2. Mark anything else as "0".

The revision to (1)(a) arose from the fact that a unit sometimes assumed that an action had been taken without actually asserting it; see the discussion of units 44 and 71 below for examples. The addition of 1(b) was meant to address the user stories, which are focused on utility of the information in the review for the reader of the review. In other words, how would the reader of the review make use of the review to mediate her own actions with regard to the product? RT members talked extensively about comments relating to "I returned the product," or "I went back to using my old product," etc. But these kinds of comments could not help the person who bought this product to use this product. The former section (1)(c) was made one of two guidelines for applying the coding scheme:

- "1" does not include descriptions relating to alternative products but not previous versions of this product or accessories for this product.
- Contacting tech support does not satisfy 1(a) or 1(b).

The coding guide now included this coding scheme, guidelines, and example units the RT selected to demonstrate application of the coding scheme. We present some of these examples below because they help to show some of the complexity that the DT later had to deal with when it came time to decide how much of the text surrounding a particular "hit" it was useful to show to make sure readers were getting useful information and not seeing misleading or confusing excerpts.

2.4.1 Example Coding Units: Hacks

Units 74, 166, & 174 below describe hacks: the use of a different indoor antenna rather than the product antenna, and the use of the product at night vs. the daytime both near and far away from the factory-provided loop antenna:

> *74. The indoor AM antenna was connected to my Technics stereo receiver and works much better than the loop antenna that shipped with the unit or when using a length straight insulated wire.*

> 166. *The TERK-1000 is tunable and enhances the AM signal during the daytime, but nightime is another story.I know AM signals attenuate and degrade at night, but I expected this antenna, with its ability to tune for a best signal and incorporating the latest technology, to also enhance the AM stations at night. Forget it.*

> 174. *The Terk made little or no difference during the day but at night when the stations I want to listen to reduce power the signal strength will increase several counts when I place the Terk next to the factory loop.*

2.4.2 Example Coding Units: Alerts

Unit 43 below describes a risk. Unit 44 implies the action of using the security features, because the author would have needed to use them in order to determine that they need to be more user-friendly.

> *43. Remember, that if you live in an urban environment, a unit like this exposes you to possible identity theft.*

> *44. The security features need to be more user-friendly. People are just not using them and they are getting hurt.*

2.4.3 Example Coding Units: Potential Problems for the Classifier

These samples represent sentences that fit the overall structures we were looking for but are otherwise difficult to understand without additional context. Taken out of context, they may be of very little use or, in some cases, misleading to readers. Unit 63 describes an action, the writer's user experience of the product, but this is not an action in response to any specific problem. Unit 64 implies that the user completed a firmware upgrade. Unit 65 refers to an action, but not using this product, rather opting to use a different product.

> *63. I was disappointed after using the Netgear MR814v2 for more than a week to determine it causes my RCA cable modem (supplied by Comcast) to unexpectedly restart, losing my Internet connection, and to degrade the connection speed when it does work.*

> *64. The Netgear firmware upgrade did not resolve the problem.*

> *65. After going back to my non-wireless router, the Netgear MR614, all is fine and my cable modem no longer restarts unexpectedly.*

2.4.4 Example Coding Units: Sentences that Provide Context

The final examples are sentences that might otherwise be discarded for not fitting strictly within the guidelines of instructional text. But we saw these as the kinds of valuable context-providing statements that we wanted the classifier to group with those statements. Unit 70 provides context for unit 71, which implies that the writer investigated other equipment (the network card).

> *70. I can't say much about this router, it gets the job done and gets it done well.*

> *71. I had some trouble but it wasn't the router's fault it was the network card I had (broke).*

With the coding guide stable, the RT turned its attention to training members of the other teams in order to test the coding guide and, most importantly, to generate the training set needed for the classifier. The resulting discussions from the training sessions identified further questions, but the trainers from the RT generally concluded that the new coders were applying the codes successfully and that the coding scheme was stable.

The leader of the RT created "homework" for 11 of the 13 total team members that consisted of an MS Excel file with approximately 1,000 units per team member to code overnight and upload to a shared-access directory. On the following morning, the RT leader concatenated all those files – consisting of nearly 7,000 total coded units – and provided them to the DT.

2.5 User-Centered Influences On Research and Development

Of course, user experience design is interested in more than just "making things look pretty." In rapid prototyping, UX becomes a way for making sure that the use value of the tool in relation to a particular user remains a focus for the entire development team. Given the compressed timeframe of tool development in this project, various types of UX testing such as blueprinting, journey mapping, and other forms of user-testing research were not an option. Our UX team found itself articulating user needs and values through the process of scrum-based development as well as adding important direction in terms of the project itself and possible uses for the research and development teams' work.

Leveraging scrum-based development, the UXT sought to iteratively narrow the focus of the application to suit the emerging possibilities of the data set while keeping users' needs in mind. In our early discussions, the UX team went to work designing possible user stories for the data set. As we fleshed out the initial user stories, we determined that the ability to locate pieces of genres or, perhaps, subgenres within the primary genre of product reviews could be of value. Rather than focus on traditional consumer review moves (e.g comments on the quality of build materials, perceived value of product, descriptions of retailer interaction and support, etc.), we focused on "how to" moves embedded within more typical review language.

While the RT developed disciplinary-based labels and a coding guide to reliably identify post-purchase use information describing how to use the product for maximum effect or how to overcome design flaws in certain contexts, the UX team focused on making this information usable and useful for potential readers. These users would be the same type of people who might be using Amazon.com to begin with: folks looking to buy consumer electronics. We also kept a second user story in mind that was more like those of us in the room: scholars interested in locating subgenres within a given text corpus.

The design team determined that, on the whole, "helpful advice" would be the focus of the Use What You Choose service and that the kind of information users would find valuable was obtainable based on the early results of the RT.

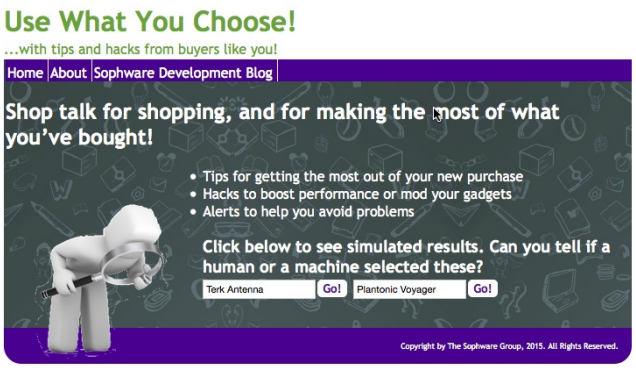

Figure 1. UWYC Input Screen – HTML version

The UX team proceeded to design mock ups, wire frames, and, finally, an HTML & CSS-based front end that included a sample input screen and a sample results screen. At the conclusion of our three day workshop, we had both the front and back-end resources for the service working, though due to limited time, our front-end model used a satic or "canned" set of results for demonstration purposes. The results shown, however, did use results derived from I/O with the back end service.

2.6 Developing the Classifier

The DT consisted of three team members. For the task of rapid prototyping, the DT's primary responsibilities included devising methods to read and clean the annotated data set of training and testing sentences compiled by the Research Team, creating a binary support vector machine (SVM) classifier for analysis, and producing output that could be read by the entire UWYC team for qualitative verification of the app's capabilities.

2.7 Text Processing

At its core, the development of a machine learning text classifier involves creating textual models characterized by maximizing highly distinguishable features and minimizing less informative features such as punctuation, function words, and/or typographic cases. The first step in this model building is what is often called text normalization or text cleaning [16]. In this section, we will describe the protocols uses to clean coded sentences culled by the RT and the rationale behind these protocols.

The first step in the text processing protocol is putting natural language sentences organized by the RT into a Python programming environment, which, in this case, is a simple matter of using Python to open an .csv file. This input procedure yields a series of raw texts with the original punctuation and whitespace intact. The next step is to reduce the variability of this raw text by removing non-essential textual features. Consequently, punctuation marks such as commas and periods are deleted and all words are converted into lowercase. The manner at which computers read strings makes this conversion necessary. The words "Headphones" and "headphones" may convey the same semantic content to a human reader and can be collapsed as two instances of a same word or concept. However, a computer reads these words as separate items because of the discrepancy in spelling. When asked to mark the significant features of a text, the weight assigned to "headphones" would then be *divided* between two example ("headphones": 1, "Headphones": 1) rather than totaling 2 ("headphones": 2). This type of *noise* – information that convolutes interpretation – can render a word such as "headphone" as less significant to the overall message of the text than it really is.

The subsequent text processing steps endeavor to further reduce noise from the natural language text by removing stopwords from a pre-designed list. For the most part, the stopword list contains function words such as articles, prepositions, pronouns, connectives, and verbs of existence, which, while important for grammatical structure, carry little semantic information. The use of stopword lists is a common practice in many classification and information retrieval tasks [18]. The assumption here is that topic discovery or retrieval tasks depend on the more infrequent words in a corpus, not syntactic placeholders. For example, an Amazon review about headphones is typified more by the occurrence of the word "headphone" than "the" or "a" – words that likely will be more frequent in a review. By removing articles such as "the" or "a" we heighten the weight of "headphone" in the corpus by eliminating competing word counts. However, we should state that the use of a stopword list is not innocent. The words that we decide to retain or subtract will influence future analytical steps, and if treated as a default step, may confound research design. For example, conditionals such as "if" and negations such as "no" or "not" are automatically stripped. However, in assessing the instructional and non-instructional content of an Amazon review, conditionals and negations can undergird moves definitive of the genre such as advising readers "not" to perform a certain task with the production or explaining how a production could be used given certain conditions. Consequently, we did not filter if conditionals and negations in our text processing step.

Upon completion of the text processing protocols, each sentence is vectorized into a term document matrix based on their term frequency-inverse document frequency weights. In this step, each sentence provided by the RT is converted into a bag of words representation. Each word type in the corpus is accounted for in an array. This array functions as a sort of master vocabulary for the corpus. In a parallel step, the raw sentences coded by the RT are treated as an individual document matrix of terms. The frequencies and absences of each term in the document matrix is noted in each document matrix. Moreover, each term is assigned a weight based on term frequency-inverse document frequency weighting. This weighting process represents another attempt to minimize the significance of commonly occurring words and to maximize the significance of less common words. In this calculation, terms that appear with high frequency within a document and across the corpus as a whole receive lower weights. Meanwhile, those terms that occur with high frequency within a document and low frequency across the corpus are given higher weights. These higher weights serve as one of the distinguishing features of a document.

2.8 SVM Machine Learning

With the coded sentences cleansed of noise and converted into term-document matrices, they are now amenable to machine learning. Term weights for each document (essentially the counts of present and absent terms per document given the corpus vocabulary) function as features that the machine learning algorithm will use to assign a document to the "Instructional" and "Non-instructional" classes. The UWYC app uses a support vector machine (SVM) algorithm [10] as its classification method.

In order to "teach" the machine learning classifier to differentiate between Instructional and Non-instructional sentences, the coded data provided by the RT was divided during the development sprints into training and test sets. The training set comprised of .80 of the total corpus and was exposed to the SVM algorithm to establish the decision-making mechanisms for classification. The testing set comprised of .20 of the total coded data and was used to determine the accuracy of the UWYC classifier.

We should note that the data coded by the RT included an unbalanced distribution of classes. Of the 7,088 coded sentences from the Amazon review corpus, only 709 were classed as 1 or Instructional. 6,372 sentences were classed as 0 or Non-instructional. This placed a premium on the 1 or Instructional sentences, and limited the size of the training and test set. Consequently, the training set for 1 sentences included 567 sentences; the test set included the remaining 142 sentences unseen by the classifier. This quantity is far less than ideal. In general, we would like a larger volume of training and testing sentences to work with; however, given the time constraints of the workshop we were constrained to this provisional level of validation. The following confusion matrix report illustrates the results of classifier testing:

Table 1. SVM Classifier Results

Class	Precision	Recall	F1-score
0	.81	.74	.77
1	.76	.83	.79
Avg/total	.79	.79	.78

The small size of the testing set should limit the enthusiasm for the classifier; however, the balanced results presented in Table 1 do provide encouraging support for the viability of a machine learning classification program educated by qualitative coding methods from the field of technical communication and professional writing. The UWYC prototype classifier's precision and recall are balanced for both 0 and 1 coding decisions, suggesting that it is equally good at identifying Instructional and Non-instructional content. This was not a given because sentences with Non-Instructional content are by definition open the more permutations and linguistic variability than those sentences that explicitly offer instructional messages about a product in the review.

For the final output of the UWYC app, reviews are aggregated. Each review is divided into its constituent sentences. Each sentence is then classed as either a 0 or 1. The percentage of 1 sentences per review is finally returned to the user so that he/she can more efficiently search out reviews that feature instructional content about the chosen product.

3. RESULTS

3.1 Promise, if not Proof, of Concept

The UWYC project showed a number of promising results that suggest that the key idea – taking advantage of predicted genre hybridity in an open system of product reviews - could be valuable.

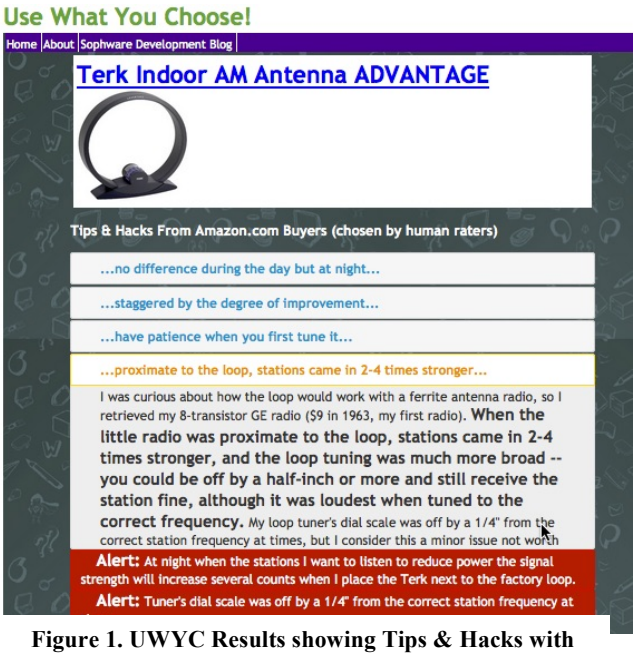

Figure 1. UWYC Results showing Tips & Hacks with one tip expanded and others collapsed

3.2 The Application…worked!

The UWYC back-end service prototype outputs its classification results in .csv files. One column features the original natural language review. A secondary column indicates the percentage of Instructional sentences found in the review (1.0 - 0.0). In this section, we reflect on a sampling of classified reviews from the cellphones and accessories category of the Amazon review corpus.

Table 2. Sample Output: Two Sentences with Instructional Content

Compact and fits snugly in the ear without discomfort. Set up was an ease.	1.0
A necessity for all clumpsy (SIC) people who drop everything. The screw on the back of the case needs to be tightened (SIC) properly, but besides that. It is definitely (SIC) worth the money to protect your pda.	1.0

Both examples above were found to be completely comprised of Instructional sentence types. In the first example, two sentences are devoted to reporting consequences of using the cellphone accessory. In the second example, the writer describes how the accessory solves a problem for "people who drop everything" and

that the accessory protects a pda. Moreover the writer indicates a hack to the product that improves its performance.

On the other end of the spectrum are those reviews that were judged by the classifier to have no Instructional content based on the RT's code book.

Table 3. Sample Output: Two Sentences without Instructional Content

I can't believe that my phone stays charged for six days. What a great deal. I only turned on bluetooth when I needed it, otherwise, I believe it would have lasted only a few days.	0.0
AND ONCE AGAIN, THIS WAS A REALLY GREAT VALUE. ARRIVED QUICKLY. VERY SATISFIED WITH THIS PRODUCT. I WILL DEFINITELY BUY IT AGAIN. ALOHA FROM HAWAII.	0.0

The first review above focuses on personal experiences with the product, but does not offer advice to readers or explain how the accessory solves a problem. The second review emphasizes the delivery and price point of the item and suggests future actions, although these actions do not account for the specific operations of the product.

Because we focused on extracting the clearest signal from Amazon review data through our text processing and code book, the discovery of 1.0 and 0.0 reviews offer less interesting cases than those reviews that feature both Instructional and Non-instructional content. After all, the classifier is tuned to these extremes. Reviews that combine Instructional and Non-instructional content introduce more variability into the analytical pipeline. The sample in Table 4 illustrates this point:

Table 4. Sample Output: One sentence with Instructional and Non-instructional content

Quick Verdict: Looks awkward on your face, it is awkward to wear, and too awkward to setup. Skip this for a Jawbone headset if you truly need one.Full Review: Although this Jabra headset works as advertised, I think that it is much more trouble than it is worth. I got this when Jabra was the leading brand, and I I had numerous issues with it. For one, it is designed with injection-molded earpieces that can fit on the left or right ear. This is great it theory, but in order to switch ears, you have to rotate the earpiece, which causes the earpiece (blue in the picture) to become loose over time. The earpieces are difficult to clean. The cheap paint chipped off of this headset very quickly. I found the sync operation too difficult for practical use (wait for light to blink rapidly to set into discovery mode, then sync) because if you leave it active and ready to make calls at all times the battery life is very short lived. I still use this occasionally when I have iSight/FaceTime set up on my iMac or MacBook Pro and I want to wander around the room while maintaining a conversation. The sound itself is very clear, but there is always a short delay and it is generally strange to listen to a conversation that was meant for stereo (as on video conference) with just mono sound. In the current market, this headset is way too large and does not have the latest Bluetooth 2.1 EDR+ technology for more robust and long distance connections.	.45

The above review mixes descriptions of the purchase, evaluations, recommendations, and instructional statements as defined by the RT. The writer leads with a brief narrative regarding the motivation behind the order. This narrative is followed by a description of the problems presented by the product, which then transitions into possible user hacks of the product. In this case, the writer is using what we have termed Non-Instructional content as scaffolding devices for the transmission of Instructional content. Thus, while the review itself may feature an equal proportion of Instructional and Non-instructional content, the strength of the Instructional content may be greater for the nuanced use of Non-instructional statements. This suggests that the advantage of the UWYC machine learning classifier is in the way that it can automatically track the compositing of rhetorical moves.

3.3 The Process Was Valuable

Prior to the workshop, only a handful of our team members knew one another. None of us had worked together before. And coming into the three-day experience, only the workshop leaders had experience designing and implementing software systems. Our implementation work in this project was, as with the work of the Research and UX design teams, exploratory in nature. That is, we had learning as our primary goal. Specifically, we set out to learn if a phenomenon like genre hybridity could be found in product reviews as hypothesized by Selber [1] and suggested, albeit faintly, by Skalicky [14]. Working together, we learned that it could be found and, based on the results of the RT, that it could be reliably found by humans. We also learned from the DT that it was at least plausible that the signals for instructional text are distinct enough from that of the persuasive components of reviews that we could train a machine-learning algorithm to identify these in unseen texts. Finally, we learned from the UXT that finding the bits of instructional texts in product reviews and presenting them in a distinct view could be a useful service for consumers in its own right.

Upon reflection, we liken the work of our three teams as a kind of elaborate, hands-on thought experiment using scrum methods. As both DT and RT members grappled with what they saw in front of them, each group worked to reconcile conclusions with others on the team. For the RT, this took on the shape of qualitative inquiry, wherein each rater compared results with a single peer to reach agreement, and with pairs persuading the larger RT that their assessments were correct. We can contrast this type of knowledge making with the weighting of unit characteristics that the DT members tracked as the machine-learning algorithms engaged test data. Each team provided their results as input to the UXT, who worked to understand how an end user – a consumer or another

academic researcher – might encounter the information in a scenario of use. All told, it served as a fascinating way to engage genre conjectures derived from genre theory.

3.4 Beyond the Workshop

The RT plans to extend its work by coding more units in the corpus, eventually resulting in a training set two or three times as large as the initial training set. This should permit the DT to train a more effective machine-learning model. However, the RT also seeks to develop a process for coding sentences that satisfies some standard of epistemic validity so that the coded units might be more useful for theoretical research within the disciplines of rhetoric and technical communication. This will involve having two coders code each unit. For each pair of coders, Cohen's Kappa [15] will be calculated. The mean Kappa for all pairs can be used to assess reliability overall. Pairwise Kappas allow assessment of whether particular pairs struggled or succeeded based on a common understanding of the coding guide. Finally, the RT will recruit two "naïve" coders, too – researchers not present during the two-day workshop but who could be asked to code a set of sentences with the coding guide to see if the guide works outside the original group. This permits an assessment of the coding guide's reproducibility [20].

The DT and the UXT will work, in the meantime, to pair the back-end and front-end prototype systems to further test the way information is presented to end users. For the time being, we will use only those texts available in the SNAP corpus in order to allow us to run validation testing on a stable set of texts, eventually comparing the reliability results from the RT with the machine classifier. This should give us not only an indication of the viability of the idea for a "live" service – one that could point to the full Amazon review system or something similar – but it may also provide convincing evidence of the validity of genre hybridity as a feature of open systems.

4. ACKNOWLEDGMENTS

The Authors would like to thank all of the UWYC team members: Heather Alexander, Casey Boyle ,Steffen Guenzel, Sally Henschel, David Kaufer, Suzanne Lane, Timothy Laquintano, Christine Stephenson. We also thank the Rhetoric Society of America Summer Institutes and our 2015 hosts, the University of Wisconsin – Madison, for the opportunity to bring this group of scholars together. Finally, we thank our anonymous reviewers for their helpful feedback.

5. REFERENCES

[1] Freedman, A., & Medway, P. (1994). Locating genre studies: Antecedents and prospects. *Genre and the new rhetoric*, 1-20.

[2] Miller, C. R. (1984). Genre as social action. *Quarterly Journal of Speech,* 70(1984), 151-167.

[3] Schryer, Catherine F. "Records as genre." *Written communication* 10, no. 2 (1993): 200-234.

[4] Miller, C.R. (1994). Rhetorical community: The cultural basis of genre. *Genre and the new rhetoric*, *6778*.

[5] Miller, Carolyn R., and Dawn Shepherd. "Blogging as social action: A genre analysis of the weblog." *Into the blogosphere: Rhetoric, community, and culture of weblogs* 18, no. 1 (2004): 1-24.

[6] Leskovec, J., & Sosic, R. (2014). Snap. py: SNAP for Python, a general purpose network analysis and graph mining tool in Python.

[7] J. McAuley and J. Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. RecSys, 2013.

[8] Mirel, B., Barton, E., & Ackerman, M. (2008). Researching telemedicine: Capturing complex clinical interactions with a simple interface design. *Technical communication quarterly*, *17*(3), 358-378.

[9] Boettger, R. K., & Palmer, L. (2010). Quantitative content analysis: Its use in technical communication. *Professional Communication, IEEE Transactions on*, *53*(4), 346-357.

[10] Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*(3), 273-297.

[11] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. *The J. Mach Learn. Res.*, *12*, 2825-2830.

[12] Selber, S. A. (2010). A rhetoric of electronic instruction sets. Technical Communication Quarterly, 19(2), 95-117.

[13] Skalicky, S. (2013). Was this analysis helpful? A genre analysis of the Amazon. com discourse community and its "most helpful" product reviews. *Discourse, Context & Media*, *2*(2), 84-93.

[14] MacNealy, M. S. (1998). *Strategies for Empirical Research in Writing*. Longman.

[15] Carletta, J. (1996). Assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, 22(2), 249–254.

[16] Beveridge, Aaron. "Looking in the Dustbin Data Janitorial Work, Statistical Reasoning, and Information Rhetorics." Computers and Composition Online Fall 2015-Spring 2016 (2015): n.p. casit.bgsu.edu/cconline/fall1… (Accessed 4/18/2016).

[17] Rajaraman, A., & Ullman, J. D. (2012). *Mining of massive datasets* (Vol. 1). Cambridge: Cambridge University Press.

[18] W. J. Potter and D. Levine-Donnerstein, "Rethinking validity and reliability in content analysis," *Journal of Applied Communication Research*, vol. 27, no. 3, pp. 258–284, Aug. 1999.