2017 IEEE 25th International Conference on Program Comprehension (ICPC)

Analyzing User Comments on YouTube Coding Tutorial Videos

Elizabeth Poché, Nishant Jha, Grant Williams, Jazmine Staten, Miles Vesper, Anas Mahmoud
Division of Computer Science and Engineering, Louisiana State University
Baton Rouge, LA, 70803
{epoche8, njha1, gwill18, jstate1, mvespe1, amahmo4}@lsu.edu

***Abstract*—Video coding tutorials enable expert and novice programmers to visually observe real developers write, debug, and execute code. Previous research in this domain has focused on helping programmers find relevant content in coding tutorial videos as well as understanding the motivation and needs of content creators. In this paper, we focus on the link connecting programmers creating coding videos with their audience. More specifically, we analyze user comments on YouTube coding tutorial videos. Our main objective is to help content creators to effectively understand the needs and concerns of their viewers, thus respond faster to these concerns and deliver higher-quality content. A dataset of 6000 comments sampled from 12 YouTube coding videos is used to conduct our analysis. Important user questions and concerns are then automatically classified and summarized. The results show that Support Vector Machines can detect useful viewers' comments on coding videos with an average accuracy of 77%. The results also show that SumBasic, an extractive frequency-based summarization technique with redundancy control, can sufficiently capture the main concerns present in viewers' comments.**

I. Introduction

Since its launch in 2005, YouTube has quickly become the largest video sharing platform on the Internet. As of the year 2016, YouTube has reported more than a billion users with over 300 hours of video uploaded every minute[1]. YouTube relies on freely-contributed user-generated content that covers a broad variety of subjects. Such content is made freely available to the consumers. Similar to other social media platforms, the core of YouTube's philosophy is to support and promote collaboration between content creators (YouTubers) and content consumers (viewers). YouTube enables community interaction through several features such as *thumps-up* and *thumps-down*, video sharing, user text comments, and video replies.

In recent years, YouTube has become a main source for programmers to learn new programming skills [1], [2]. Coding tutorial videos typically take the form of a *screencast*—a developer uploads a copy of his/her computer screen as code is typed along with an audio narration to walk viewers through the process. A coding tutorial video can range from a fundamental HELLO WORLD programming lesson to more advanced topics, describing personal development experiences, practices, and unique programming strategies [3]. Through these tutorials, viewers can visually follow source code changes and learn about the environment where the program is compiled, debugged, and executed. This provides a unique and more interactive medium for sharing knowledge in comparison to traditional text-based resources, such as manuals, programming books, and online platforms (e.g., *Stack Overflow*).

Previous research in this domain has focused on either viewers [4] or content creators [3]. MacLeod et al. [3] conducted a large-scale study to understand how and why software developers create video tutorials. In a more recent work, Ponzanelli et al. [4] proposed CodeTube, an approach which helps developers find coding content on YouTube more effectively. In our work, we focus on the link connecting content creators to their audience. More specifically, we analyze viewer's feedback expressed through user comments on coding videos. A comment is a textual description of a viewer experience with the video [5]. Extensive analysis of YouTube comments in other domains has revealed that they can be a viable source of descriptive annotations of videos, carrying *socially significant* knowledge about users, videos, and community interests in general [6]. YouTube comments have been analyzed to detect cyberbullying [7], drug abuse [8], and infer people's political affiliations.

Our main assumption in this paper is that extracting, classifying, and summarizing information in viewers' comments on coding tutorial videos will help content creators to be more engaged with their audience and plan more effectively for future videos. Ultimately, boost their visibility, number of views, and number of subscribers [9], [10]. In particular, in this paper we document the following contributions:

- Data collection: We collect and manually classify 6,000 YouTube comments sampled from 12 different coding tutorial videos. These videos cover a mixture of simple and advanced topics from a broad range of programming languages.
- Comments classification: We investigate the performance of different text classifiers to identify effective classification configurations that can automatically distinguish viewers' feedback that requires action from the content creator from other miscellaneous comments (e.g., spam).
- Comment summarization: We investigate the performance of different text summarization techniques in capturing the main topics raised in the content concern comments on coding videos.

[1] https://www.youtube.com/yt/press/statistics.html

978-1-5386-0535-6/17 $31.00 © 2017 IEEE
DOI 10.1109/ICPC.2017.26

The remainder of this paper is organized as follows. Section II discusses YouTube as an educational tool, motivates our research, and lists our research questions. Section III describes our data collection protocol and discusses the information value of user comments on coding tutorial videos. Section IV investigates the performance of multiple text classification techniques in capturing useful comments. Section V describes and evaluates the performance of different comment summarization strategies in generating meaningful summaries of useful viewers' comments. Section VI discusses the threats to the study's validity. Section VII reviews related work. Finally, Section VIII concludes the paper and discusses prospects for future work.

II. Background, Motivation, and Approach

YouTube has been recently utilized as an effective educational tool across a variety of subjects, such as literature [11], [12], health sciences [13], arts and humanities [11], language learning [14], and STEM education [15]. Using YouTube in the classroom has been found to motivate and increase student participation, critical awareness, and enhance their deep learning skills [16], [17].

In computer science education, YouTube is mainly used in programming classes. Novice and expert programmers use coding videos to visually learn new programming languages, understand new programming structures and techniques, and expose themselves to other programmers' styles of coding and debugging [2]. Carlisle [1] examined the effect of using YouTube for teaching Java at a college level. The author reported that students who watched YouTube videos were more prepared for class and performed better on the test. Furthermore, the results showed that YouTube videos provided a source of outreach for the university by drawing the attention of younger demographics.

Similar to other major social media platforms (e.g., Facebook and Twitter), at its core, YouTube encourages a participatory culture; YouTube viewers can share, rate, reply to, and comment on videos. Comments represent a direct communication channel between content creators and their audience. Through comments, viewers can express their feelings toward certain content, participate in active discussions with other viewers, and direct questions to content creators. From a content creator's perspective, comments provide a valuable source of feedback to understand their viewers' needs and alter their content accordingly.

Research has shown that the level of interaction between content creators and their viewers is a main dimension of *popularity* on YouTube [18]. More specifically, YouTubers who consistently read and respond to their viewers are more likely to retain subscribers and gain more views [9], [10]. However, as videos become more popular, they tend to receive more comments. Larger number of comments causes potentially valuable user feedback to be buried within comments that do not provide any useful feedback about the content of the video [19]. This makes it almost impossible for content creators and viewers to go through these comments manually to respond to user concerns, or engage in useful discussions. Furthermore, similar to other media platforms, spammers take advantage of the openness and popularity of YouTube to spread unsolicited messages to legitimate users. These messages are used to launch phishing, advertising, or malware attacks on large demographics of users or simply to advertise for other channels on the website [20], [21].

Motivated by these observations, we introduce a study aimed at separating and presenting useful user feedback on YouTube coding tutorial videos. Our presumption is that, gathering, classifying, and then summarizing comments in an informative way helps content creators respond more effectively to their viewers' concerns [22]. To guide our research, we formulate the following research questions:

- ***RQ_1: How informative are comments on coding tutorial videos?*** YouTube is a public service. Viewers typically come from different backgrounds with a broad spectrum of opinions. Therefore, the assumption that all comments carry useful information is unrealistic. While some users may post comments that are beneficial to both the content creators and viewers, others may post comments that are entirely irrelevant or do not contain any useful information. Therefore, the first objective of our analysis is to determine how informative are the comments on YouTube coding videos. In other words, do such comments have enough information value that can be useful for the content creators?
- ***RQ_2: Can informative comments be automatically identified and classified?*** Filtering massive amounts of comments manually can be a tedious and error-prone task. Therefore, for any solution to be practical, an automated approach is needed to effectively filter through data and separate informative from uninformative comments with a decent level of accuracy.
- ***RQ_3: How to summarize and present informative comments?*** YouTube comments can be lexically (e.g., abbreviations and colloquial terminology) and semantically (i.e., lack of proper grammars) restricted. Furthermore, several comments might raise similar concerns. Presenting such large, and maybe redundant, amounts of raw comments to the content creator can cause confusion. This emphasizes the need for automated methods to summarize informative comments to enable a more effective data exploration process.

III. Qualitative Analysis

In this section, we answer our first research question regarding the potential value of comments on YouTube coding tutorial videos. In particular, we describe our data collection protocol along with the main findings of our manual qualitative analysis.

A. YouTube Comments Collection

To conduct our analysis, we sampled 6000 comments from 12 coding tutorial videos selected from multiple coding

TABLE I
YOUTUBE VIDEOS USED IN OUR ANALYSIS

#	DATE	ID	TOPIC	LENGTH (MINS)	VIEWS	COMMENTS	USEFUL	MISC.
1	Nov 10, 2010	bWPMSSsVdPk	HTML	12:16	1,115,800	1,170	112	388
2	Jan 27, 2011	Ggh_y-33Eso	HTML	14:33	368,715	1,089	141	359
3	Apr 6, 2011	ABRP_5RYhqU	C++	8:25	691,212	1,276	106	394
4	Apr 6, 2011	jTS7JTud1qQ	C++	9:12	491,014	1,346	156	344
5	Apr 21, 2013	Z149x12sXsw	Android	25:57	728,067	2,360	164	336
6	Aug 3, 2013	S3t-5UtvDN0	C++	33:31	878,825	1,254	252	248
7	Oct 21, 2013	3JluqTojuME	CSS	14:07	1,145,092	1,409	229	271
8	Jun 3, 2014	WPvGqX-TXP0	Java	34:29	1,767,831	3,761	109	391
9	Aug 29, 2014	yPu6qV5byu4	MySQL	41:09	620,717	1,730	115	385
10	Nov 10, 2014	N4mEzFDjqtA	Python	43:15	1,576,235	3,063	159	341
11	Nov 20, 2014	Rub-JsjMhWY	C++	1:10:32	1,474,219	5,645	128	372
12	Jul 5, 2015	lisiwUZJXqQ	C#	1:24:58	391,330	1,476	156	344

channels[2]. The data was collected on September 6, 2016. Table I describes these videos, including each video's unique identifier (ID) which appears at end of each YouTube video's URL, the main topic of the video, the length of the video in minutes, and the number of views and comments. To sample our dataset, we collected all available comments from each video, a total of 41,773 comments. These comments were retrieved from YouTube using the latest YouTube Data API[3]. This API extracts comments and their metadata, including the author's name, the number of likes, and the number of replies. Results are returned in JSON format. 500 comments were then randomly sampled from each video. The *C# Random()* function was used to sample the data. This function uses a time-dependent default seed value from the *C# System* library.

B. Qualitative Analysis

To answer our first research question, we manually analyzed the sampled data to identify comments that carry some sort of useful feedback to content creators and viewers. To simplify the classification process, we categorized the comments into two general categories, including content concerns (informative) and other miscellaneous comments. These categories can be described as follows:

- **Content Concerns:** These comments include questions or concerns about certain parts of the video content that needs further explanation (e.g., *"at 11:14 do i have to use 'this' at all?"*). This category also includes comments that point out errors within the video (e.g., *"At 27:10 there's a small mistake. The first parameter is the starting index, the second parameter is the number of chars"*). Content concerns can also include requests for certain future content (e.g., *"Can you make a Arcpy video?"*). This category also includes comments which are suggestions to improve the quality of the tutorial by giving advice to the programmer (e.g., *"You are talking way too fast"*) or suggesting a change in the media settings (e.g., *"The current font is too small and the background is too dark cant see it"*).
- **Miscellaneous:** This category includes all other comments that do not provide any technical information about the content of the video. For instance, some comments include praise (e.g., *"Very well explained, many thanks!"*), insults (e.g., *"This tutorial is trash."*), or spam (e.g., *"Learn Web Development Essentials and Become a Web Developer From Scratch in this Complete HTML & CSS Beginner's Course learn more here!"*). Such comments are not beneficial to the content creators or the viewers.

The sampled comments in our dataset were manually examined and classified by a team of five annotators. Our team of annotators included two PhD students in software engineering, two undergraduate computer science students (senior level), and a professor in software engineering. The team members have an average of 4 years of experience in programming. A tool was created to aid in the manual classification process. Each annotator classified each comment and saved the results to a separate database. The tool then merged the different classifications and majority voting was used to classify each comment. No time constraint was enforced to avoid fatigue factors (i.e., the participant becomes bored or tired and starts randomly classifying the comments). It took all five annotators an average of 3 weeks to fully classify the data (6000 comments).

The results of the manual classification process is shown in Table I. Around 30% of the comments were found to be content-related, or useful, meaning that the majority of comments are basically miscellaneous. These findings answer ***RQ1*** and motivates ***RQ2*** and ***RQ3***. More specifically, given that only one third of comments contain potentially useful information, how can such comments be automatically identified and effectively presented to the content creator?

IV. COMMENTS CLASSIFICATION

The second phase of our analysis is concerned with automatically classifying our ground-truth dataset into informative (technical concerns) and miscellaneous comments. In

[2]http://seel.cse.lsu.edu/data/icpc17.zip
[3]https://developers.google.com/youtube/v3/

particular, the main research question at this phase is ***RQ₂: Can informative content-relevant comments be automatically identified and classified?*** This question can be further broken down into two sub-questions, first, what classifiers are most effective in the context of YouTube comments, and second, what classification features generate the most accurate results?

A. Classifiers

To answer the first part of ***RQ₂***, we investigate the performance of two text classification algorithms, including Naive Bayes (NB) and Support Vector Machines (SVM). Both algorithms have been extensively used in related literature to classify YouTube comments [5], [19], [20], [23], [24]. In detail, NB and SVM can be described as follows:

- **Naive Bayes (NB):** NB is an efficient linear probabilistic classifier that is based on Bayes' theorem [25]. NB assumes the conditional independence of the attributes of the data. In other words, classification features are independent of each other given the class. In the context of text classification, NB adopts the *Bag-of-Words* (BOW) approach. More specifically, the features of the model are the individual words of the text. The data is typically represented by a 2-dimensional *word x document* matrix. In the Bernoulli NB model, an entry in the matrix is a binary value that indicates whether the document contains a word or not (i.e., {0,1}). The Multinomial NB, on the other hand, uses normalized frequencies of the words in the text to construct the *word x document* matrix [26].
- **Support Vector Machines (SVM):** SVM is a supervised machine learning algorithm that is used for classification and regression analysis in multidimensional data spaces [27]. SVM attempts to find optimal hyperplanes for linearly separable patterns in the data and then maximizes the margins around these hyperplanes. Technically, support vectors are the critical instances of the training set that would change the position of the dividing hyperplane if removed. SVM classifies the data by mapping input vectors into an N-dimensional space, and deciding on which side of the defined hyperplane the data instance lies. SVMs have been empirically shown to be effective in domains where the data is sparse and highly dimensional [28].

SVM and NB have been found to work well with short text. Short-text is a relatively recent Natural Language Processing (NLP) type of text that has been motivated by the explosive growth of micro-blogs on social media (e.g., Tweets and YouTube and Facebook comments) and the urgent need for effective methods to analyze such large amounts of limited textual data. The main characteristics of short-text include data sparsity, noisy content, and colloquial terminologies. In what follows, we investigate the performance of these two classifiers in detecting useful feedback present in coding videos' comments.

B. Text processing

In the context of text classification, researchers typically use combinations of text pre-processing strategies to remove potential noise and help the classifier to make more accurate predictions [28]. These strategies include text reduction techniques such as stemming (**ST**) and stop-word (**SW**) removal. Stemming reduces words to their morphological roots. This leads to a reduction in the number of features (words) since only one base form of the word is considered. Stop-word removal, on the other hand, is concerned with removing English words that are considered too generic (e.g., *the*, *in*, *will*). We also remove words that appear in one comment only since they are highly unlikely to carry any generalizable information [29]. Table II shows the different pre-processing steps in order.

Furthermore, in our analysis, we use Multinomial NB, which uses the normalized frequency (tf) of words in their documents [26]. $tf_{t,c}$ can be defined as the number of times a term t occurs in a comment c [30]. This value is often adjusted as $ln(1+tf_{t,c})$ to ensure that words that appear too frequently get penalized as they receive less weight than less frequent words. Multinomial NB is known to be a robust text classifier, consistently outperforming the binary feature model (Multivariate Bernoulli) in highly diverse, real-world corpora [26].

C. Implementation and Evaluation

To implement NB and SVM, we use Weka [31], a data mining suite that implements a wide variety of machine learning and classification techniques. We also use Weka's built-in stemmer (`IteratedLovinsStemmer` [32]) and stop-word list to pre-process the comments in our dataset.

SVM is invoked through Weka's SMO, which implements John Platt's sequential minimal optimization algorithm for training a support vector classifier [33]. Choosing a proper kernel function can significantly affect SVM's generalization and predictive capabilities [34]. The kernel function of SVM transforms the nonlinear input space into a high dimensional feature space. Therefore, the solution of the problem can be represented as being a linear regression problem [35], [36]. In our analysis, the best results were obtained using the universal kernel. Universal kernels have been found to be more effective than other kernels when the data is noisy [34], [37].

To train our classifiers, we use 10-fold cross validation. This method of evaluation creates 10 partitions of the dataset such that each partition has 90% of the instances as a training set and 10% as an evaluation set. The evaluation sets are chosen such that their union is the entire dataset. The benefit of this technique is that it uses all the data for building the model, and the results often exhibit significantly less variance than those of simpler techniques such as the holdout method (e.g., 70% training set, 30% testing set) [38].

Recall, precision, and F-measure are used to evaluate the performance of the different classification techniques used in our analysis. Recall is a measure of coverage. It represents the ratio of correctly classified instances under a specific label to the number of instances in the data space that actually belong

TABLE II
TEXT PRE-PROCESSING STEPS APPLIED TO THE COMMENTS

Original Comment	"I don't really understand the importance of an abstract class. Isn't it just a more limited version of a normal class?"
Stop-Word Removal	"don't understand importance abstract class Isn't limited version normal class"
Stemming	"don understand import abstract cl isn lim ver norm cl"
vectorToStringFilter	<abstract, cl, don, import, isn, lim, norm, understand, ver>
TF Weights	<0.69, 1.09, 0.69, 0.69, 0.69, 0.69, 0.69, 0.69, 0.69>

TABLE III
PERFORMANCE OF DIFFERENT CLASSIFICATION CONFIGURATIONS

CLASSIFIER	P	R	F
NB	0.63	0.71	0.67
NB + STEMMING	0.62	0.76	0.68
NB + STEMMING + STOP-WORD	0.61	0.73	0.67
SVM	0.79	0.75	0.77
SVM + STEMMING	0.77	0.73	0.75
SVM + STEMMING + STOP-WORD	0.75	0.65	0.70

to that label. Precision, on the other hand, is a measure of accuracy. It represents the ratio of correctly classified instances under a specific label to the total number of classified instances under that label. Formally, if A is the set of data instances in the data space that belong to the label λ, and B is the set of data instances that were assigned by the classifier to that label, then recall (***R***) can be calculated as $R_\lambda = |A \cap B|/|A|$ and precision (***P***) can be calculated as $P_\lambda = |A \cap B|/|B|$. We also use $F = 2PR/(P + R)$ to measure the harmonic mean of recall and precision.

D. Results and Discussion

The results of our classification process are shown in Table III. The results show that SVM ($F = 0.77$) was able to outperform NB ($F = 0.68$). This difference in performance can be explained based on the characteristics of our data space. More specifically, even though YouTube comments are limited in size, the feature space (number of words) is typically very large [28]. This can be attributed to the fact that people on social media use informal language (e.g., slang words, acronyms, abbreviations) in their comments [39], [40]. This drastically increases the number of features that the classifier needs to process and leads the vector representation of comments to be very sparse (i.e., only very few entries with non-zero weights). While machine learning algorithms tend to over-learn when the dimensionality is high, SVM has an over-fitting avoidance tendency—an inherent behavior of margin maximization which does not depend on the number of features [41]. Therefore, it has the potential to scale up to high-dimensional data spaces with sparse instances [28]. NB tends to be less accurate when dealing with sparse text spaces due the unrealistic conditional independence assumption among the classification features.

Our results also show that removing English stop-words impacts the performance SVM negatively ($F = 0.70$). This can be explained based on the observation that some of the eliminated stop-words represent valuable features to our classification problem. For example, the comment, *"why did you leave space every time you started a new one eg htmlbody Etc"* was incorrectly classified as miscellaneous after the removal of the stop-words <*a, did, on, why, and you*>. Similarly, the comment, *"is it true that this video covers 95% of what we need to know about MySQL"* was incorrectly classified as miscellaneous after the removal of the stop-words <*is, about, it, of, that, to, we, what*>. In general, comments related to the video content tend to take the form of questions. Removing words, such as *why* or *what*, from the comment's text changes the question to a statement, thus leading to misclassification.

The slight decline of SVM performance after applying stemming ($F = 0.75$) can be attributed to instances of over-stemming. For example, in the two comments: *"You are truly incredible"* and *"I think you could increase the quality of your instructional videos by speaking in a less monotonous way"* both words *incredible* and *increase* were stemmed to *incr*, thus corrupting the space of features by considering two different words as one and causing the second comment to be incorrectly classified as miscellaneous.

V. COMMENTS SUMMARIZATION

The third phase of our analysis is focused on generating succinct summaries of the content concern comments. A summary can be described as a short and compact description that encompasses the main theme of a collection of comments related to a similar topic [42], [43]. The main objective is to assimilate the perspectives of a large number of user comments to focus the content creator's attention on common concerns. In particular, the research question at this phase of our analysis is, ***RQ$_3$: How to summarize informative comments on YouTube coding videos?***

A. Summarization Behavior: A Pilot Study

The summarization task in our analysis can be described as a multi-document summarization problem, where each comment is considered as a separate document. In general, multi-document summarization techniques can be either extractive or abstractive. Extractive methods select specific documents, or

keywords, already present within the data as representatives of the entire collection. Abstractive methods, on the other hand, attempt to generate a summary with a proper English narrative from the documents [42]. Generating abstractive summaries can be a very challenging task. It involves heavy reasoning and lexical parsing to paraphrase novel sentences around information extracted from the corpus [44]. This problem becomes more challenging when dealing with the lexically and semantically limited YouTube comments. In contrast, extractive summaries have the advantage of being full sentences, thus carry more meaningful information [45], [43]. However, as only a limited number of representative comments are selected, important concerns about certain topics might be missed.

To get a sense of how developers would identify the main concerns in a list of comments, we conducted a pilot study using two participants. Our first study participant has 7 years of experience in HTML programming and the second has 5 years of experience in C++ programming. Each participant was assigned two videos in their area of expertise and asked to watch the videos, read the list of comments we provided, and select the top 10 comments that captured the viewers' concerns raised in the comments. No time constraint was enforced.

Our pilot study participants were interviewed after the experiment. Both of them implied that they initially identified the main viewers' concerns on the video after going through the list of comments for 3-4 times. Once these topics were identified (stood out due to their frequent appearance), they selected sample comments that they thought represented these topics, basically, included terms from the main topics identified. Our participants also implied that, while it was not easy to pick these representative comments, especially due to the very high degree of diversity in the language used by viewers, they would still prefer to see full comment summaries over keyword summaries. For example, the first video in Table I (`bWPMSSsVdPk`) provides a very basic tutorial of HTML language. Examining the list of useful comments on this video shows that they revolve around three main concerns, including embedding images (`<img></img>`) in HTML pages, the `DOCTYPE` tag, and questions about the browser used to parse the HTML code. Table IV shows some of the comments related to the `image` and `DOCTYPE` topics, responding to any of these comments would address majority of the concerns raised in other comments about these topics. However, only displaying tags such as <`img`, `DOCTYPE`, `browser`> might not be as informative to describe what these concerns actually are.

B. Automated Summarization

Given our observations during the pilot study, the main challenge at this point is to determine the best strategies for generating summaries, or the comments that address majority of the common concerns raised by the viewers. Formally, an extractive summarization process can be described as follows: given a topic keyword or phrase T and the desired length for the summary K, generate a set of representative comments S with a cardinality of K such that $\forall s_i \in S, T \in s_i$ and $\forall s_i, \forall s_j \in S, s_i \nsim s_j$. The condition $s_i \nsim s_j$ is enforced to ensure that selected comments provide sufficiently different information (i.e., not redundant) [46].

In the literature, several extractive comment summarization techniques have been proposed [43], [46], [47], [48]. Majority of these techniques rely on the frequencies of words as an indication of their perceived importance [47]. In other words, the likelihood of words appearing in a human generated summary is positively correlated with their frequency [46]. In fact, this behavior was observed in our pilot study where our participants indicated that the frequency of certain terms was the main factor for identifying the general user concerns and selecting their summary comments.

Based on these observations, in our analysis, we investigate the performance of a number of term frequency based summarization techniques that have been shown to work well in the context of social media data. These techniques include:

- **Term Frequency (TF)**: Term, or word, frequency is the most basic method for determining the importance of a word. Formally, a word value to its document is computed as the frequency of the word (f_i) divided by the number of words in the document N. In our analysis, the importance of a comment of length n is calculated as the average of the weights of its individual words $\sum_{i=1}^{n} f_i/N$. Note that unlike classical TF, we compute TF as the proportion of times a word occurs in the entire set of comments rather than individual comment. This change is necessary to capture concerns that are frequent over the entire collection of comments [46].
- **Hybrid TF.IDF**: Introduced by Inouye and Kalita [46], the hybrid TF.IDF approach is similar to the TF approach. However, term frequency (TF) is accompanied with a measure of the term scarcity across all the comments, known as inverse document frequency (IDF). IDF penalizes words that are too frequent in the text. An underlying assumption is that such words do not carry distinctive value as they appear very frequently. Formally, TF.IDF can be computed as:

$$TF.IDF = f(w, D) \times \log \frac{|D|}{|d_i : w \in d_i \wedge d_i \in D|} \quad (1)$$

  where *f(w,d)* is the term frequency of the word *w* in the entire collection, $|D|$ is the total number of comments in the collection, and $|d_i : w \in d_i \wedge d_i \in D|$ is the number of comments in *D* that contains the word w. The importance of a comment can then be calculated as the average TF.IDF score of its individual words.
  To control for redundancy, or the chances of two very similar comments getting selected, before adding a top comment to the summary, the algorithm makes sure that the comment does not have a textual similarity above a certain threshold with the comments already in the summary. Similarity is calculated using cosine similarity between the vector representations of the comments.
- **SumBasic**: Introduced by Nenkova and Vanderwende [47], SumBasic uses the average term

TABLE IV
EXAMPLE OF DIFFERENT COMMENTS RELATED TO SIMILAR TOPICS

Comments related to the topic "image"	Comments related to the topic "DOCTYPE"
I dont get the image i made when i put Img srcimgbmp	you dont need doctype html
how did u get the bitmap image thing i dont have it im using win10	not important to start with the DOCTYPE html tag
btw my image is not showing on the webpage	What about DocType

frequency (TF) of comments' words to determine their value. However, the weight of individual words is updated after the selection of a comment to minimize redundancy. This approach can be described as follows:

1) The probability of a word w_i in the input corpus of size N words is calculated as $\rho(w_i) = n/N$, where n is the frequency of the word in the entire corpus.
2) The weight of a comment S_j is calculated as the average probability of its words, given by $\sum_{i=1}^{|S_j|} \rho(w_i)/|S_j|$
3) The best scoring comment is selected. For each word in the selected comment, its probability is reduced by $\rho(w_i)_{new} = \rho(w_i) \times \rho(w_i)$. This step is necessary to control for redundancy, or minimize the chances of selecting comments describing the same topic using high frequency words.
4) Repeat from 2 until the required length of the summary is matched.

C. Evaluation

We recruited 12 programmers (judges) to participate in our experiment, including 7 graduate students in computer science, 2 undergraduate computer science students, and 3 industry professionals. An interview was conducted prior to the experiment to determine the programming experience of our study participants. 5 of our judges reported experience in C++, 6 judges reported experience in Java, and 5 judges reported experience HTML and CSS programming. Based on our judges area of expertise, 6 videos from Table I were selected to conduct our analysis. Each participant was then assigned 2-3 videos that matched their area of expertise. The main task was to watch the video, read the provided list of content concern comments of the video, and identify 10 representative comments that capture the common concerns raised by the viewers. The comments were randomized to avoid any bias where users select comments from the top of the list. No time constraint was enforced, but most of our participants responded within a two week period. In summary, each video was summarized by 5 judges, generating a total of 30 summaries. Table V describes our experimental setup.

The different summarization techniques proposed earlier were then used to generate the summaries for the 6 videos included in our experiment. To enhance the accuracy of the summarization techniques, in addition the list of dictionary stop-words, we compiled a new list of English-like words or colloquial terms that appear frequently in our dataset but do not carry any useful information. In general, we identified four

TABLE V
EXPERIMENTAL SETUP (JUDGES: NUMBER OF JUDGES, EXP : JUDGES' AVERAGE YEARS OF EXPERIENCE IN THE TOPIC, VIDEOS: VIDEOS FROM TABLE I USED IN THE EXPERIMENT)

TOPIC	JUDGES	EXP (YRS)	VIDEOS
HTML	5	5.5	1, 2
C++	5	4	3, 4
Java	5	4.5	8
CSS	5	4.5	7

categories of these words, including: words that are missing apostrophes (e.g., *whats*, *dont*, *theyre*, *ill*), abbreviations (e.g., *plz*, *pls*, *whatevs*), acronyms (e.g., *lol*, *btw*, *omg*), and emotion words that do not contribute any technical value (e.g., *love*, *hate*, *like*). Stemming was also applied to minimize the redundancy of different variations of words (e.g., *show, showing, shown, and shows*). In what follows, we present and discuss our results.

D. Results and Discussion

To assess the performance of our summarization techniques, for each video, we calculate the average term overlap between our study participants' selected lists of comments (reference summaries) and the various automatically generated summaries. Formally, a recall of a summarization technique t is calculated as:

$$Recall_t = \frac{1}{|S|} \sum_{i=1}^{|S|} \frac{match(t, s_i)}{count(s_i)} \quad (2)$$

where S is the number of reference summaries, $match(t, s_i)$ is the number of terms that appear in the reference summary s_i and the automated summary generated by t, and $count(s_i)$ is the number of unique terms in the reference summary s_i. An automated summary that contains (recalled) a greater number of terms from the reference summary is considered more effective [49], [46], [47]. In our analysis, recall is measured over different length summaries (5, 10, 15, and 20 comments included in the summary).

The results are shown in Fig. 1. Randomly generated summaries (comments were selected randomly from each set using the `.NET Random` class) were used to compare the performance of our proposed methods. Our results show that all methods outperformed the random baseline. On average, SumBasic was the most successful in summarizing common concerns found in viewers' comments, followed by Hybrid

TF.IDF, and finally TF which achieved the lowest recall. Fig. 2 shows that best Hybrid TF.IDF's results were observed at the lowest threshold value (0.2), which means that for a comment to be included in the summary it should not be more than 0.2 similar to any other comment already in the summary. In general, our results suggest that a simple frequency-based approach with redundancy control can actually be sufficient to capture the common concerns in viewers comments.

We also observe that a major factor influencing the quality of the generated summaries is the irrelevant words, or words that do not provide any useful information to the content creator. This problem becomes dominant when dealing with social media data. More specifically, people tend to use a highly diverse set of vocabulary in their comments on platforms such as YouTube, Twitter, and Facebook. Such rapidly evolving vocabulary includes abbreviations, acronyms, emoticons, phonetic spellings, and other neologisms [39], [40]. Our expectation is that a continuous filtering of these words will significantly enhance the performance of our summarization techniques. To verify this claim, we perform another manual round of irrelevant word removal from the list of human generated summaries (e.g., the comment *"how you do font colors plz hlp"* is reduced to *"font colors"*). We then repeat our recall analysis. The results, shown in Fig. 3, show more enhancement over the recall rates, confirming the ability of the proposed techniques of generating informative summaries that actually capture the majority of users' concerns. However, given the rapid pace in which the language evolve online, creating and maintaining a generic up-to-date list of uninformative words can be a challenging task. Therefore, any future implementation of our approach should give the content creator the ability continuously to update such lists of words.

It is important to point out that other, more computationally expensive, techniques such as Latent Dirichlet Allocation (LDA) [50] and text clustering have been employed in the literature to generate comment summaries [43], [51]. However, such techniques typically require heavy calibration of multiple parameters in order to generate decent results. This limits the practicality of such techniques and their ability to produce meaningful summaries [42], [45]. In our analysis, we rely on computationally inexpensive techniques (frequency-based) that are relatively easier to implement and calibrate. This design decision was enforced to facilitate an easy transfer of our research findings to practice. The ability to produce the output while keeping the operating cost as low as possible has been found to be a major factor influencing tools adaptation in practice [52].

VI. Related Work

MacLeod et al. [3] conducted a large-scale study to understand how and why software developers create video tutorials on YouTube. The authors conducted a set of interviews with a number of content creators to understand their main motives and the challenges they face when producing code-focused videos. The authors reported that videos can be a very useful medium for communicating programming knowledge

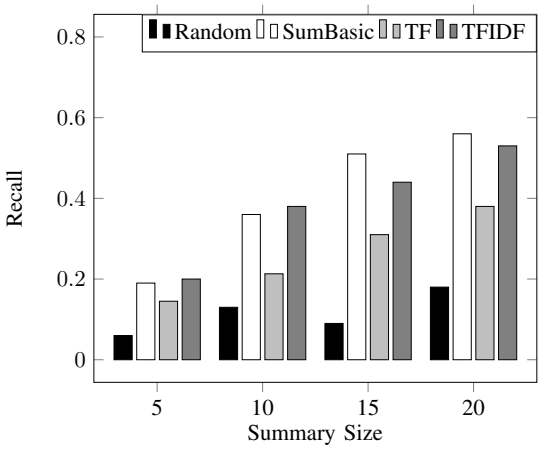

Fig. 1. The performance of the different summarization techniques measured at different length summaries (5, 10, 15, 20)

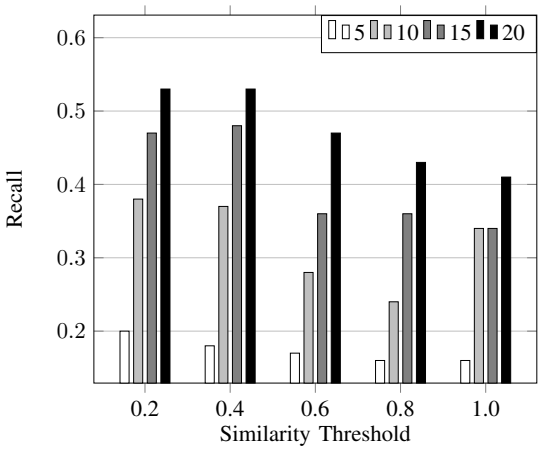

Fig. 2. The performance of Hybrid TF.IDF at different similarity thresholds measured at different length summaries (5, 10, 15, 20)

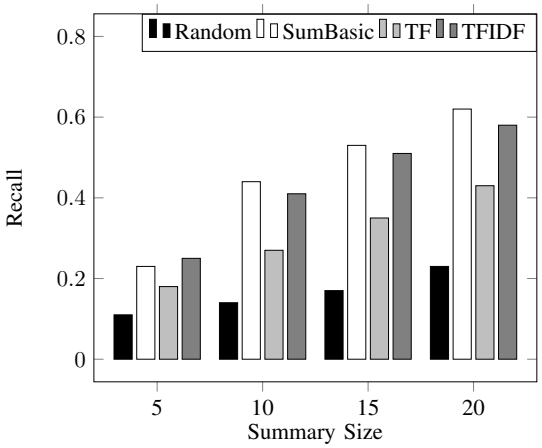

Fig. 3. The performance of the different summarization techniques measured at different length summaries (5, 10, 15, 20) after manually removing more irrelevant words

by complementing traditional code documentation methods. In addition, the authors found that content creators typically build their online reputation by sharing videos through their social media channels.

Ponzanelli et al. [4] presented CodeTube, an approach that facilitates the search for coding videos on the web by enabling developers to query the content of these videos in a more effective way. The proposed approach splits the video into multiple fragments and returns only the fragments most related to the search query. CodeTube also automatically complements the video fragments with relevant *Stack Overflow* discussions. Two case studies were used to show the value of CodeTube and its potential benefits to developers.

Parnin et al. [53] conducted a set of interviews with several software bloggers to understand their motivations for blogging and the main challenges they face. The authors reported that developers used blogging for documentation, technology discussion, and announcing progress, where personal branding, knowledge retention, and feedback were the main motivations for blogging. The authors also manually analyzed several blog posts to understand the community interactions and social structures in software blogs. The results showed that the most frequent types of comments blog authors received were questions and expressions of gratitude. The results also showed that management of blog comments was one of the main challenges that faced software bloggers.

Dinakar et al. [7] proposed an approach for detecting and modeling textual cyberbullying in YouTube comments. The authors applied a range of binary and multi-class classifiers over a corpus of 4,500 manually annotated YouTube comments to detect sensitive topics that are personal to individuals. The results showed that cyberbullying can be detected by building individual topic-sensitive classifiers that capture comments with topics related to physical appearance, sexuality, race, culture, and intelligence.

Momeni et al. [48] studied the properties and prevalence of useful comments on different social media platforms. The authors used several classification models to detect patterns of user-generated comments on different social media platforms including Flickr and YouTube. The results showed that comments that contain a higher number of references, a higher number of named entities, fewer self-references and lower sentiment polarity are more likely to be inferred as useful.

Khabiri et al. [43] proposed a topic-based summarization algorithm to summarize comments on YouTube videos. In particular, the authors used topic models to generate topics over each video's comments. The comments were then clustered by their topic assignments into groups of thematically-related subjects. Within each cluster, a small set of key informative comments is selected using a PageRank-style algorithm. The proposed approach was evaluated over a dataset of YouTube comments collected from 30 videos and manually annotated by human judges. The analysis showed that topic-based clustering can yield competitive results in comparison to traditional document summarization approaches.

Inouye and Kalita [46] compared the performance of different summarization algorithms in summarizing social media micro-blogs. These algorithms were evaluated against manually produced summaries and summaries produced using state-of-the-art summarization techniques. The results showed that Hybrid TF.IDF and SumBasic were able to generate the highest levels of agreement with the human generated summaries.

Siersdorfer et al. [5] investigated the relationship between the content and the ratings of YouTube comments. The authors presented an in-depth study of commenting and comment rating behavior on a sample of more than 10 million user comments on YouTube and Yahoo. The authors used data mining to detect acceptance of comments by the community and identify highly controversial and polarizing content that might spark discussions and offensive commenting behavior.

Thelwall et al. [6] analyzed a large sample of comments on YouTube videos in an attempt to identify patterns and generate benchmarks against which future YouTube research can be conducted. The authors reported that around 23.4% of comments in their dataset were replies to previous comments. Positive comments elicited fewer replies than negative comments. The authors also reported that audience interaction with YouTube videos can range from passive entertainment to active debating. In particular, videos about religion tend to generate the most intense discussions while videos belonging to the music, comedy, and *how to & style* categories generated the least discussions in their comments sections.

VII. Threats to Validity

The study presented in this paper has several limitations that might affect the validity of the results [54]. A potential threat to the proposed study's internal validity is the fact that human judgment is used to classify our sample of YouTube comments and prepare our ground-truth summaries. This might result in an internal validity threat as humans, especially students, tend to be subjective in their judgment. However, it is not uncommon in text classification to use humans to manually classify the data, especially in social media classification [43], [6]. Therefore, these threats are inevitable. However, they can be partially mitigated by following a systematic classification procedure using multiple judges, at different levels of programming experience, to classify the data.

Threats to external validity impacts the generalizability of the results [54]. A potential threat to our external validity stems from the dataset used in our experiment. In particular, our dataset is limited in size and was generated from a limited number of YouTube videos at a certain point of time. To mitigate this threat, we ensured that our dataset of comments was compiled from videos covering a broad range of programming languages (Java, C++, C#, Python, HTML, CSS, and MySQL). Furthermore, we used randomization to sample 500 comments from the set of comments collected for each video. While considering all the data instances in the analysis might enhance the external validity of our results, manually analyzing such large amounts of data can be a tedious and error-prone task which might jeopardize the interval validity of the study.

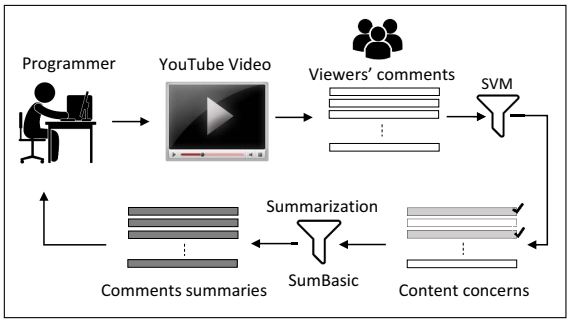

Fig. 4. A summary of the proposed approach

Other external threats might stem from the tools we used in our analysis. For instance, we used Weka as our machine learning and classification platform. Nonetheless, Weka has been extensively used in the literature and has been shown to generate robust results across a plethora of applications. Furthermore, using publicly available open source tool enables other researchers to replicate our results.

Construct validity is concerned whether the metrics used in the analysis capture the aspect of performance they are supposed to measure [54]. In our experiment, there were minimal threats to construct validity as the standard performance measures recall, precision, and F-Score, which are extensively employed in related research, were used to assess the performance of the different methods investigated in our analysis.

VIII. Conclusions and Future Work

This paper presents an approach (summarized in Fig. 4) for identifying, classifying, and presenting useful user comments on YouTube coding tutorial videos. Our analysis is conducted using 6000 comments sampled from 12 coding tutorial videos covering a broad range of programming languages. A manual qualitative analysis was conducted to determine the information value of sampled comments. Our results showed that around 30% of these comments contained useful information that can be beneficial to the content creator. These comments included concerns related to the content of the video, such as questions about a specific problem in the video or recommendations for future content.

The manually classified comments were then automatically classified using Naive Bayes and Support Vector Machines. These two classifiers have been extensively used in short-text classification tasks. The results showed that SVM was more effective than NB in capturing and categorizing content request comments. Finally, content concern comments were summarized using multiple automated summarization algorithms. A pilot study using two expert programmers was conducted to understand the human summarization behavior when dealing with YouTube comments. Our results showed that programmers prefer extractive summaries over keyword-based abstractive summaries. Based on these observations, frequency-based summarization algorithms with redundancy protection were used in our analysis. These algorithms, including TF, Hybrid TF.IDF and SumBasic, are known for their simplicity (implementation, calibration, and computation overhead) and decent performance in the context of social media data.

A human experiment using 12 programmers was conducted to assess the performance of the different summarization techniques. The results showed that the summarization algorithm SumBasic was the most successful in recalling the majority of common concerns in viewers comments. The results also showed that the performance of the algorithm could be enhanced by frequently eliminating lists of irrelevant words.

The long term goal of the proposed work is to help content creators respond faster to their viewers concerns, deliver higher quality content, and ultimately gain more subscribers. To achieve this goal, the line of work in this paper will be expanded along several directions as follows:

- Data collection: A main part of our future effort will be devoted to collecting larger datasets from a more devise set of programming languages. More data will enable us to conduct in depth analysis of viewers' commenting patterns on coding tutorial videos, thus draw more robust conclusions.
- Tool support: A working prototype that implements our findings in this paper will be developed. This prototype will enable us to run usability studies to assess the usefulness of our proposed approach for content creators.
- Human studies: Our future work will include conducting surveys and human experiments with actual YouTubers to get a more in-depth understanding of their needs and expectations as well as response patterns to their viewers.

Acknowledgment

We thank our study participants. This work was supported in part by the Louisiana Board of Regents Research Competitiveness Subprogram (LA BoR-RCS), contract Number: LEQSF(2015-18)-RD-A-07.

References

[1] M. Carlisle, "Using YouTube to enhance student class preparation in an introductory java course," in *Proceedings of the 41st ACM Technical Symposium on Computer Science Education*, 2010, pp. 470–474.

[2] M. Piteira and C. Costa, "Learning computer programming: Study of difficulties in learning programming," in *Proceedings of the 2013 International Conference on Information Systems and Design of Communication*, 2013, pp. 75–80.

[3] L. MacLeod, M. Storey, and A. Bergen, "Code, camera, action: How software developers document and share program knowledge using YouTube," in *International Conference on Program Comprehension*, 2015, pp. 104–114.

[4] L. Ponzanelli, G. Bavota, A. Mocci, M. Di Penta, R. Oliveto, B. Russo, S. Haiduc, and M. Lanza, "Codetube: Extracting relevant fragments from software development video tutorials," in *International Conference on Software Engineering*, 2016, pp. 645–648.

[5] S. Siersdorfer, S. Chelaru, W. Nejdl, and J. San Pedro, "How useful are your comments? Analyzing and predicting YouTube comments and comment ratings," in *international conference on World Wide Web*, 2010, pp. 891–900.

[6] M. Thelwall, P. Sud, and F. Vis, "Commenting on YouTube videos: From guatemalan rock to el big bang," *Journal of the American Society for Information Science and Technology*, vol. 63, no. 3, pp. 616–629, 2012.

[7] K. Dinakar, R. Reichart, and H. Lieberman, "Modeling the detection of textual cyberbullying," in *AAAI Conference on Web and Social Media, Workshop on the Social Mobile Web*, 2011.

[8] J. Walther, D. DeAndrea, J. Kim, and J. Anthony, "The influence of online comments on perceptions of antimarijuana public service announcements on YouTube," *Human Communication Research*, vol. 36, no. 4, pp. 469–492, 2010.

[9] C. Chau, "YouTube as a participatory culture," *New Directions for Youth Development*, vol. 2010, no. 128, pp. 65–74, 2010.

[10] D. Welbourne and W. Grant, "Science communication on YouTube: Factors that affect channel and video popularity," *Public Understanding of Science*, vol. 25, no. 6, pp. 706–718, 2016.

[11] C. Desmet, "Teaching shakespeare with YouTube," *English Journal*, pp. 65–70, 2009.

[12] A. Clifton and C. Mann, "Can YouTube enhance student nurse learning?" *Nurse Education Today*, vol. 31, no. 4, pp. 311–313, 2011.

[13] S. Burke and S. Snyder, "YouTube: An innovative learning resource for college health education courses," *International Electronic Journal of Health Education*, vol. 11, pp. 39–46, 2008.

[14] J. Brook, "The affordances of YouTube for language learning and teaching," *Hawaii Pacific University TESOL Working Paper Series*, vol. 9, no. 2, pp. 37–56, 2011.

[15] J. Basham and M. Marino, "Understanding STEM education and supporting students through universal design for learning," *Teaching Exceptional Children*, vol. 45, no. 4, pp. 8–15, 2013.

[16] P. Duffy, "Engaging the YouTube google-eyed generation: Strategies for using web 2.0 in teaching and learning," *The Electronic Journal of e-Learning*, vol. 6, no. 2, pp. 119–130, 2008.

[17] S. Burke, S. Snyder, and R. Rager, "An assessment of faculty usage of YouTube as a teaching resource," *Internet Journal of Allied Health Sciences and Practice*, vol. 7, no. 1, p. 8, 2009.

[18] G. Chatzopoulou, C. Sheng, and M. Faloutsos, "A first step towards understanding popularity in YouTube," in *INFOCOM IEEE Conference on Computer Communications Workshops*, 2010, pp. 1–6.

[19] A. Ammari, V. Dimitrova, and D. Despotakis, "Semantically enriched machine learning approach to filter YouTube comments for socially augmented user models," *UMAP*, pp. 71–85, 2011.

[20] A. Sureka, "Mining user comment activity for detecting forum spammers in YouTube," *arXiv preprint arXiv:1103.5044*, 2011.

[21] F. Benevenuto, T. Rodrigues, V. Almeida, J. Almeida, C. Zhang, and K. Ross, "Identifying video spammers in online social networks," in *Proceedings of the 4th International Workshop on Adversarial Information Retrieval on the Web*, 2008, pp. 45–52.

[22] M. Potthast, B. Stein, F. Loose, and S. Becker, "Information retrieval in the commentsphere," *ACM Transactions on Intelligent Systems and Technology*, vol. 3, no. 4, p. 68, 2012.

[23] A. Serbanoiu and T. Rebedea, "Relevance-based ranking of video comments on YouTube," in *2013 19th International Conference on Control Systems and Computer Science*. IEEE, 2013, pp. 225–231.

[24] W. Yee, A. Yates, S. Liu, and O. Frieder, "Are web user comments useful for search," *Proc. LSDS-IR*, pp. 63–70, 2009.

[25] P. Langley, W. Iba, and K. Thompson, "An analysis of bayesian classifiers," in *Aaai*, vol. 90, 1992, pp. 223–228.

[26] A. McCallum and K. Nigam, "A comparison of event models for Naive Bayes text classification," in *AAAI-98 Workshop on Learning for Text Categorization*, 1998, pp. 41–48.

[27] C. Burges, "A tutorial on Support Vector Machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[28] T. Joachims, "Text categorization with Support Vector Machines: Learning with many relevant features," in *European Conference on Machine Learning*, 1998, pp. 137–142.

[29] J. Grimmer and B. Stewart, "Text as data: The promise and pitfalls of automatic content analysis methods for political texts," *Political Analysis*, vol. 21, no. 3, pp. 267–297, 2013.

[30] S. Robertson, "Understanding Inverse Document Frequency: On theoretical arguments for IDF," *Journal of Documentation*, vol. 60, no. 5, pp. 503–520, 2004.

[31] "Weka 3: Data mining software in java," http://www.cs.waikato.ac.nz/ ml/weka/, accessed: August, 15th.

[32] J. Lovins, "Development of a stemming algorithm," *Mechanical Translation and Computational Linguistics*, vol. 11, pp. 22–31, 1968.

[33] J. Platt, "Fast training of Support Vector Machines using sequential minimal optimization," in *Advances in Kernel Methods - Support Vector Learning*, B. Schoelkopf, C. Burges, and A. Smola, Eds., 1999, pp. 185–208.

[34] I. Steinwart, "On the influence of the kernel on the consistency of Support Vector Machines," *Journal of Machine Learning Research*, vol. 2, pp. 67–93, 2001.

[35] K. Lee, D. Palsetia, R. Narayanan, M. Patwary, A. Agrawal, and A. Choudhary, "Twitter trending topic classification," in *International Conference on Data Mining Workshops*, 2011, pp. 251–258.

[36] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida, "Detecting spammers on Twitter," in *r. In Collaboration, Electronic messaging, Anti-Abuse and Spam Conference*, 2010, p. 12.

[37] C. Micchelli, Y. Xu, and H. Zhang, "Universal kernels," *Journal of Machine Learning Research*, vol. 7, pp. 2651–2667, 2006.

[38] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *International Joint Conference on Artificial Intelligence*, 1995, pp. 1137–1145.

[39] L. Squires, "Enregistering internet language," *Language in Society*, no. 39, pp. 457–492, 2010.

[40] V. Kulkarni, R. Al-Rfou, B. Perozzi, and S. Skiena, "Statistically significant detection of linguistic change," in *International Conference on World Wide Web*, 2015, pp. 625–635.

[41] P. Brusilovsky, A. Kobsa, and W. Nejdl, Eds., *The Adaptive Web: Methods and Strategies of Web Personalization*, 2007, pp. 335–336.

[42] C. Llewellyn, C. Grover, and J. Oberlander, "Summarizing newspaper comments," in *International Conference on Weblogs and Social Media*, 2014, pp. 599–602.

[43] E. Khabiri, J. Caverlee, and C. Hsu, "Summarizing user-contributed comments," in *International AAAI Conference on Weblogs and Social Media*, 2011.

[44] U. Hahn and I. Mani, "The challenges of automatic summarization," *Computer*, vol. 33, no. 11, pp. 29–36, 2000.

[45] E. Barker, M. Paramita, A. Funk, E. Kurtic, A. Aker, J. Foster, M. Hepple, and R. Gaizauskas, "What's the issue here?: Task-based evaluation of reader comment summarization systems," in *International Conference on Language Resources and Evaluation*, 2016, pp. 23–28.

[46] D. Inouye and J. Kalita, "Comparing Twitter summarization algorithms for multiple post summaries," in *International Conference on Social Computing (SocialCom) and International Conference on Privacy, Security, Risk and Trust (PASSAT)*, 2011, pp. 298–306.

[47] A. Nenkova and L. Vanderwende, "The impact of frequency on summarization," Microsoft Research, Tech. Rep., 2005.

[48] E. Momeni, C. Cardie, and M. Ott, "Properties, prediction, and prevalence of useful user-generated comments for descriptive annotation of social media objects," in *AAAI Conference on Weblogs and Social Media*, 2013.

[49] C. Lin, "Rouge: A package for automatic evaluation of summaries," in *Workshop on Text Summarization Branches Out*, 2004, pp. 74–81.

[50] D. Blei, A. Ng, and M. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[51] Z. Ma, A. Sun, Q. Yuan, and G. Cong, "Topic-driven reader comments summarization," in *International Conference on Information and Knowledge Management*, 2012, pp. 265–274.

[52] D. Lo, N. Nagappan, and T. Zimmermann, "How practitioners perceive the relevance of software engineering research," in *Joint Meeting on Foundations of Software Engineering*, 2015, pp. 415–425.

[53] C. Parnin, C. Treude, and M.-A. Storey, "Blogging developer knowledge: Motivations, challenges, and future directions," in *International Conference on Program Comprehension*, 2013, pp. 211–214.

[54] A. Dean and D. Voss, "Design and analysis of experiments," 1999.