Visualizing Authorship and Contribution of Collaborative Writing in e-Learning Environments

Johnny Torres
ESPOL Polytechnic University
Department of Electrical and Computer Engineering (FIEC)
Guayaquil, Ecuador
jomatorr@espol.edu.ec

Sixto García
ESPOL Polytechnic University
Department of Electrical and Computer Engineering (FIEC)
Guayaquil, Ecuador
sgarcia@espol.edu.ec

Enrique Peláez
ESPOL Polytechnic University
Department of Electrical and Computer Engineering (FIEC)
Guayaquil, Ecuador
epelaez@espol.edu.ec

ABSTRACT

Nowadays, several productivity platforms provide effective capabilities to edit collaboratively the content of a document. In educational settings, e-Learning approaches have taken advantage of this functionality to encourage students to join others to complete projects that include the writing of text documents. Although collaborative writing may foster interaction among students, the existing analytical metrics on these platforms are limited and can slow down the process of review by instructors in trying to determine the level of contribution of each student in the document. In this paper, we describe an analytic framework to measure and visualize the contribution in collaborative writing.

CCS CONCEPTS

• **Applied computing** → **Education**; **Collaborative learning**; • **Human-centered computing** → *Collaborative and social computing systems and tools.*

KEYWORDS

Collaborative Writing; Natural Language Processing; Human-Computer Interaction

ACM Reference Format:
Johnny Torres, Sixto García, and Enrique Peláez. 2019. Visualizing Authorship and Contribution of Collaborative Writing in e-Learning Environments. In *24th International Conference on Intelligent User Interfaces (IUI '19), March 17–20, 2019, Marina del Rey, CA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3301275.3302328

1 INTRODUCTION

Collaboration can enhance the learning process inside as well as outside the classroom. It enables students to develop skills that could not be acquired working alone, such as, critical thinking and peer discussion. In educational models, an active learning methodology motivates the analysis of the courses' content by encouraging students to engage in activities, such as reading, discussion, problem-solving, and collaborative writing. This methodology often leads students to complete their assignments more effectively and with higher quality [4, 25].

Google Docs, Office 365 or Wikipedia are among the most popular collaboration platforms. These platforms allow students (and users in general) to edit documents collaboratively, provide versioning features (history), and some allow collaboration in real time. Although versioning is important for editors, it would be useful for reviewers to have metrics on the contribution of each editor.

Prior studies have proposed methodologies to measure the users' contribution in collaborative writing [2, 8, 18, 21], but fail to identify fine-grained contributions or its type.

In this paper, we propose a semantic analysis pipeline and visualization to measure an editors' contribution in documents written collaboratively on educational environments. We seek to provide a measurement of the overall contribution of students in collaborative writing, using both quantitative and qualitative metrics. Although the task of extracting qualitative metrics can be considered as a cognitive process, characterized by a high level of complexity [20], we leverage on recent advances on natural language understanding based on deep learning models [24].

The contributions in this paper are two-fold. First, we propose and evaluate a deep learning pipeline to measure qualitative contributions of students in collaborative documents. Second, we propose a visualization that shows the quality of the contributions on collaborative writing.

The structure of this paper is as follows: In section 2, a review of previous work on collaborative writing is presented. The proposed methodology is described in section 3. Section 5, describes the evaluation, as well as, the datasets used. A discussion of the findings and limitations is presented in section 6, then the conclusions and future work is outlined in section 7.

2 RELATED WORK

Several approaches have been proposed to measure contribution in collaborative writing. Most previous research has focused on Wikipedia or Git[1] due to their open source nature, whereas few studies use Google Docs or Office 365. All these systems keep track of each revision (e.g., edits or commit actions) in documents and provide a baseline method to measure contributions based on words or lines count.

In the context of Wikipedia, several approaches have been proposed to measure users' contributions. Viégas et al. [21] attributes

[1]https://git-scm.com/

IUI '19, March 17–20, 2019, Marina del Rey, CA, USA

ACM ISBN 978-1-4503-6272-6/19/03…$15.00
https://doi.org/10.1145/3301275.3302328

the content of a sentence to the user who made the last change. However, it does not recognize correctly the author of the content reintroduced after its deletion. Not tracking small changes (e.g., adding words, fixing grammar, or formatting) is another drawback of working at the level of sentences. Thus, this method does not correctly measure the overall contribution of users.

Ding et al. [7] developed a visualization of enterprise wikis at large scale, measuring users' contributions based on the number of edits done by each user to different pages. Hess et al. [11] also proposed a method to measure the extent of a user's partial contribution in each version of the document. It compares the current version to the previous one, and the overall contribution is the sum of all partial contributions. Based on a similar approach, Sabel [18] calculated the contributions of users to each version of the document and then used the results as the users' rating in a reputation schema.

In the context of software development with source code management systems such as Apache Subversion [16] or Git, the contribution measurement has been analyzed regarding code ownership. Version control systems often measure the quality of source code at line-level tracking [2]. This coarse-grained level tracking is also used as a basic unit to identify contributors. Line-level tracking allows identification of the user who made the last change in a specific line of code in a file, but it loses the information about the original creator. This functionality is appropriate for collaborative software development environments because it allows detection of the user responsible for introducing defects or making changes in the code. However, this mechanism is not suitable for tracking the original contributors of the content or detecting changes at a more fine-grained level such as words or characters.

Most recent approaches, analyze the history of edits and focus on determining the authorship at sentences [1] or words [8] level. After establishing the authorship, these methods calculate the overall contribution of each author based on the sum of the number of words or sentences [23] but fall short in explaining the type of contributions.

3 METHODOLOGY

The proposed methodology relies on an end-to-end pipeline that can be used by instructors to evaluate collaborative writing, as proposed in [8, 22]. Figure 1 shows an overview of the system, which performs the following actions: (a) all different document formats are transformed to an intermediate representation, (b) an authorship detection algorithm determines what each editor quantitatively did in the document, (c) a syntactic and semantic analysis is performed on the documents' content for qualitative contribution analysis, (d) a ranking method scores the contribution of all editors based on both quantitative and qualitative metrics, and (e) an interactive visualization of the document shows the editors' content contributions, as well as, the ranking metrics.

3.1 Intermediate representation

The initial step transforms the documents created in one of the collaborative applications (e.g., MS Word, Google Docs, Wikipedia) to an intermediate representation in XML format, similar to Wikipedia internal representation to handle versioning of the articles. The parent level is the document and its global properties, such as title, description, source application, creation date/time. The child level represents each revision of a document and contains the following properties: author, modification date/time, text content. This representation allows to standardize the format of documents created by different applications, and simplify preprocessing for the next tasks in our pipeline.

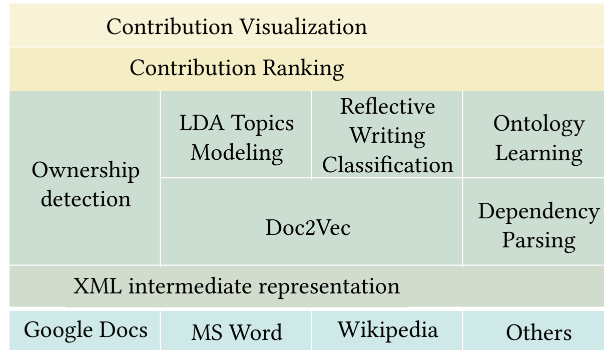

Figure 1: System architecture overview.

3.2 Words Ownership

Flöck and Rodchenko [10] presented a tree model approach to establish content authorship, and therefore measure the users' contribution in a document. In their initial version, the model only considered paragraphs and sentences. This limitation produced a precision of approximately 60%, which is unsuitable for usage in production. In an improved algorithm, Flöck and Acosta [8] proposed a fine-grained level representation of the document by considering paragraphs, sentences, and tokens. The authors map the documents' representation into a k-partite graph model, which is more efficient and reported a 30% increase in precision. We extended the k-partite graph-based model to establish the authorship of a document's words.

The original implementation of the authorship algorithm uses the Wikipedia pages internal structure, but we generalized it to our intermediate representation. In this k-partite graph-based model, the parent nodes represent the revisions, the nodes in the next level represent paragraphs, followed by sentences, and finally tokens in the lower level. The hierarchical links denote a *containment* relationship, e.g., if $paragraph_i \in P_k$, $sentence_i \in S_k$, and $(p, s) \in revision_k$, therefore paragraph $p_i \subset sentence_i$. Links between nodes of different revisions K denote the origin of that specific node, e.g., $sentence_i \in S_k$ points to a $word_i \in sentences_j \in S_{k-1}$, that means the work was created in previous version. An important feature is the detection of significant changes to the document in a small period, which the algorithm labels as vandalism or plagiarism patterns. This feature allows us to detect students who are probably copying and pasting large chunks of information from external sources into the document. Finally, the algorithm labels the leaf nodes with its respective author.

3.3 Syntactic Analysis

For the syntactic analysis, we applied the part-of-speech (POS) tagging [6]. We leverage recent advances in POS tagging through the use of bidirectional long short-term memory (bi-LSTM). This model provides state-of-the-art performance in multilingual documents, and it is less sensitive to the corpus size [17], which fits our scenario.

Then, we incorporate a named entity recognition (NER) tagging, which avoids relying on domain-specific knowledge and handcrafted features. Furthermore, we wanted the NER model to learn from small corpora. To this end, we use a neural network architecture based on bi-LSTMs and conditional random fields (CRF) [12].

Finally, we use a dependency parser based on neural networks [5]. This greedy, transition-based dependency parser learns how to classify words using a small number of dense features to represent the dependency arc between words. As a result of the dependency parsing, we can establish the relationship between entities and words and allows us to construct a baseline ontology for the document by combining the tags in our syntactic analysis.

3.4 Topic modeling

In the semantic analysis, we associate authorship of the main ideas with a high level of contributions to the document by applying topics' modeling. We hypothesize that topics represent the main ideas in a document. To extract topics, we apply *lda2vec* [15], a hybrid model consisting of Dirichlet Topic Models [3] (LDA) and Word Embeddings [14] (word2vec). This model brings together the semantic richness of *word2vec* and the interpretability of LDA.

Traditional LDA parameterized "topics" as categorical distribution over sparse word space. The *lda2vec* model replaces this limited multivariate distribution on an embedding space. The model performs clustering of a set of words semantically related to topics using embedding space. To improve the performance, the model uses pre-trained word embedding from a domain-general corpus (Spanish Wikipedia) instead of learning word vectors and topics jointly.

3.5 Reflective classification

Continuing the semantic analysis, we are interested in identifying the reflective writing style (i.e., students' response related to experiences, opinions, events), which denotes a more profound contribution compared to other writing styles. Reflective writing is one of the most common ways to evaluate student learning and analysis of skills. However, being a different style, reflective writing may require additional effort when used. When used appropriately, reflective writing denotes a deep understanding or mastery of the critical concepts of a particular subject by students. One approach to determining reflective writing is to define a set of patterns generally employed in reflective writing [19]. These patterns can be detected as meta expressions [author] - [reflection], such as: "I think", "I have considered", "our recommendation".

Before describing the classification of reflective sentences, we describe the documents' content representation. Due to the sparsity in text documents, the techniques commonly used to extract semantic features, like bag-of-words (BoW), fail to generalize for unseen data [14]. Therefore, we use a distributed representation of the documents' content [13].

BoW modeling has two disadvantages representing text as a fixed-length feature vector. First, it loses the ordering of the words in a sentence. Second, it ignores the semantics of the words, e.g., words like "strong" and "powerful" have a similar meaning; hence their vectors are expected to be close in the hyperplane.

Le and Mikolov [13] proposed a model for learning a fixed-length feature representation of variable-length text, such as sentences or paragraphs. Using an unsupervised algorithm, the model represents the sentences or paragraphs by dense vectors, which can predict the words in the document. The distributed representation captures semantic structures that are used to classify sentences as using a reflective style or not.

3.6 Contribution Ranking

The model calculates the score of each metric independently, and it is the percentage of the editor's contribution to the total. The quantitative metrics is the number of words; however, the qualitative metrics include the number of topics, reflective sentences, and ontologies. The overall contribution ranking is based on the weighted average (by α) of the quantitative and qualitative metrics, and it is calculated as follows:

$$R_i = \alpha * \sum_i \frac{w_i}{W} + \left[(1-\alpha) * \left[\sum_m \sum_i \frac{q_m i}{Q} \right] \right]$$

Where, the first term w_i are the words contributed by student i, and the second term is the average of the qualitative metrics for student i.

4 VISUALIZATION

The visualization renders the document's content in an interactive web page showing the list of authors and summarizing their contribution metrics. The instructors can explore the words written by each student by selecting in the specific student, in the right box that contains the list of authors. Alternatively, instructors can select a specific word to highlight all the content a specific student has created, or select it in the list of authors.

The list of authors summarizes the contribution in one column and shows the overall score. After the user selects an author from the list, the system shows a detailed visualization about qualitative metrics, showing the three aspects of the qualitative score: topics, ontologies, and reflective writing, contributed to the document. The detailed visualization is a D3 sunburst visualization[2], where the inner circle represents the percentage of words contributed. The next outer circle represents the number of topics, reflective writing, and ontologies. Outer circles show the details of each of the qualitative metrics, i.e., the words associated with them.

This visualization extends the work introduced by Flöck et al. [9], which is a visualization for Wikipedia articles, similar to Wang [22], in which the authors propose a visualization of the authorship of the words in the document.

This tool has a client-server architecture, where the client is the web interface that connects to a web service in the server-side. Although the web interface relied on the Wikipedia articles rendering,

[2]https://bl.ocks.org/kerryrodden/7090426

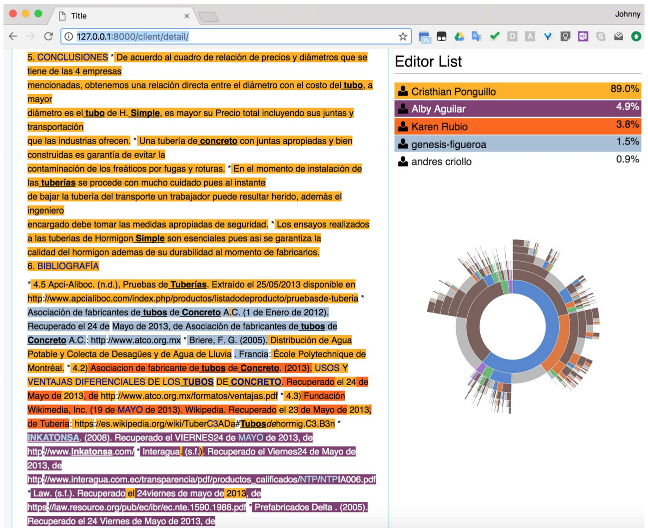

Figure 2: Screenshot of the interactive visualization for collaborative writing.

and it works as a user-script for browser extension tampermonkey[3], we have decoupled the interface from Wikipedia and made it a standalone interface so it can be integrated into any web application, as shown in figure 2.

In the server-side, the web service connects to the data source to get the complete history of the document. Initially, our data source is Google Docs, but the architecture can be extended to use different data sources, including Office 365, Wikipedia, and similar platforms with versioning support and open format policies. Then, it feeds the document history to our analytic pipeline to determine: authorship, syntactic, and semantic information. Finally, it provides the document's content along with the rendering and analytic information to the web interface in the client-side.

5 EVALUATION

The dataset used in our experiments consists of 39 Google documents with a total of 763 revisions, made by 92 students. The links of these documents were submitted to the academic system of ESPOL University by students as part of their assignments or group projects. Using the links of the documents, we collected their revision history from Google Drive storage utilizing the API[4].

We established the ground truth of the students' contributions in the documents through manual evaluations. Since we measure the relative quality of the contributions, we do not use the grading of the documents given by the professors. Instead, two research assistants analyzed each document following strict guidelines to measure the quality of the students' contribution.

Independently, each assistant measured the number of topics, reflective sentences, and ontologies that appear in each of the documents in our corpus. We calculated the agreement percentage and the kappa score to account for the quality of the annotation. The table 1 shows a simple agreement percentage (*po* column) for each of the quality metrics. The kappa score (kappa column) is a more robust measure to compare labelings by two human annotators by considering the likelihood of random labels assignment (*pe* column). The formula is defined as: $\kappa = (p_o - p_e)/(1 - p_e)$.

The kappa score is between -1 and 1 and scores higher than .8 are considered a good agreement, such as the case for topics annotation. However, agreement on reflective sentences and ontologies show the level of difficulty labeling those categories.

Table 1: Annotation agreement

metric	N	po	pe	kappa
topics	195	0.94	0.22	0.92
reflective sentences	78	0.81	0.26	0.74
ontologies	390	0.64	0.10	0.60

A third human annotator resolved the disagreements before evaluating again the scores generated by our analytic pipeline. Figure 3 shows how our model performs when compared to human evaluator. We split the dataset in 80% for training and 20% for testing. The evaluation was conducted using a 10-folds cross-validation method, and the results reported on the test set. The results shows the potential to establish a robust baseline qualitative metrics for instructors in the evaluation of collaborative writing in educational environments.

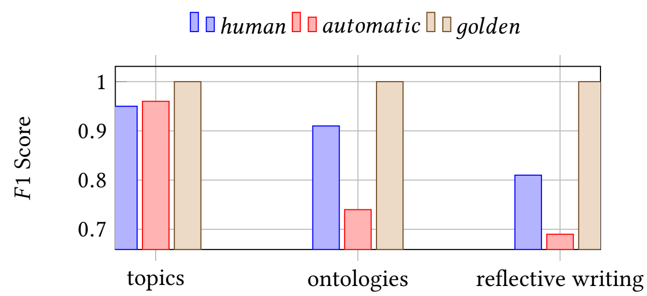

Figure 3: Performance of the model and a human evaluator with respect to a golden test set.

6 DISCUSSION

This section describes some limitations regarding the aspects in the proposed machine learning pipeline to analyze documents and visualize the contribution of authors.

6.1 Documents Analytics

In our experiments, we used documents related to a specific course that required the collaborative writing of text documents. The type of writing may be related to a specific course, some courses may require mainly technical writing, while other courses require mostly descriptive writing (which may include reflective writing). Therefore, for a generalized implementation, the system must be able to work with different types of documents and the variety of tasks of each course.

In addition, to complete their homework, students can use one of several productivity platforms for collaborative writing. Therefore,

[3]https://tampermonkey.net
[4]https://developers.google.com/drive/v2/reference/

the system should be able to read documents from heterogeneous platforms, which may have a different versioning behavior, such as those that allow real-time collaboration.

6.2 User Interface evaluation

For our visualization tool, the end-user interface requires a more thorough evaluation. The evaluation must analyze the impact on the end users regarding aspects such as integration with collaborative writing platforms, how easy it is to use them and their response capacity (ie, how fast the system can analyze a document and show the metrics).

6.3 A/B testing

The user evaluation may include an A/B testing, with respect the inner visualization in the figure 2. It could be essential to know the end-users' opinion on having two different types of visualizations regarding the extent and type of contribution.

7 CONCLUSIONS

Fully understanding the documents, especially the quality associated with them, could generate many benefits in educational environments. Mainly, for instructors, it could alleviate the workload by giving them a quick baseline of the work done by each student in collaborative writing. We improved the previous work to provide the instructors with a quantitative and qualitative visualization of metrics using an interactive web tool.

Future research directions could address some of the limitations discussed in this paper, in relation to the flow of document analysis, as well as the visualization tool. In the analysis of documents, it might be useful to consider the structure of the document in addition to the semantic analytics, verifying that they comply with specific sections requested by the professors. For the user interface, it can be interesting to implement and evaluate a seismic visualization to summarize the complete history of changes in the document considering the type of contribution.

REFERENCES

[1] Ofer Arazy and Eleni Stroulia. 2009. A Utility for Estimating the Relative Contributions of Wiki Authors.. In *ICWSM*.

[2] Robert Baggen, José Pedro Correia, Katrin Schill, and Joost Visser. 2012. Standardized code quality benchmarking for improving software maintainability. *Software Quality Journal* 20, 2 (2012), 287–307.

[3] David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3, Jan (2003), 993–1022.

[4] Charles C Bonwell and James A Eison. 1991. *Active Learning: Creating Excitement in the Classroom. 1991 ASHE-ERIC Higher Education Reports.* ERIC.

[5] Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 740–750.

[6] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12, Aug (2011), 2493–2537.

[7] Xianghua Ding, Catalina Danis, Thomas Erickson, and Wendy A Kellogg. 2007. Visualizing an enterprise wiki. In *CHI'07 extended abstracts on Human factors in computing systems*. ACM, 2189–2194.

[8] Fabian Flöck and Maribel Acosta. 2014. WikiWho: Precise and efficient attribution of authorship of revisioned content. In *Proceedings of the 23rd international conference on World wide web*. ACM, 843–854.

[9] Fabian Flöck, David Laniado, Felix Stadthaus, and Maribel Acosta. 2015. Towards Better Visual Tools for Exploring Wikipedia Article Development–The Use Case of "Gamergate Controversy". In *Ninth International AAAI Conference on Web and Social Media*.

[10] Fabian Flöck and Andriy Rodchenko. 2012. Whose article is it anyway?–Detecting authorship distribution in Wikipedia articles over time with WIKIGINI. *Online proceedings of the Wikipedia Academy* (2012).

[11] Michael Hess, Brian Kerr, and Lev Rickards. 2006. *Wiki user statistics for regulating behaviour*. Technical Report. Citeseer.

[12] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360* (2016).

[13] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. 1188–1196.

[14] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[15] Christopher E Moody. 2016. Mixing Dirichlet Topic Models and Word Embeddings to Make lda2vec. *arXiv preprint arXiv:1605.02019* (2016).

[16] C Michael Pilato, Ben Collins-Sussman, and Brian W Fitzpatrick. 2008. *Version control with subversion*. " O'Reilly Media, Inc.".

[17] Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. *arXiv preprint arXiv:1604.05529* (2016).

[18] Mikalai Sabel. 2007. Structuring wiki revision history. In *Proceedings of the 2007 international symposium on Wikis*. ACM, 125–130.

[19] Simon Buckingham Shum, Ágnes Sándor, Rosalie Goldsmith, Xiaolong Wang, Randall Bass, and Mindy McWilliams. 2016. Reflecting on reflective writing analytics: Assessment challenges and iterative evaluation of a prototype tool. In *Proceedings of the sixth international conference on learning analytics & knowledge*. ACM, 213–222.

[20] Miguel-Angel Sicilia, Nikolaos Th Korfiatis, Marios Poulos, and George Bokos. 2006. Evaluating authoritative sources using social networks: an insight from Wikipedia. *Online Information Review* 30, 3 (2006), 252–262.

[21] Fernanda B Viégas, Martin Wattenberg, and Kushal Dave. 2004. Studying cooperation and conflict between authors with history flow visualizations. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 575–582.

[22] Dakuo Wang. 2016. How People Write Together Now: Exploring and Supporting Today's Computer-Supported Collaborative Writing. In *Proceedings of the 19th ACM Conference on Computer Supported Cooperative Work and Social Computing Companion*. ACM, 175–179.

[23] Dakuo Wang, Judith S Olson, Jingwen Zhang, Trung Nguyen, and Gary M Olson. 2015. DocuViz: visualizing collaborative writing. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 1865–1874.

[24] Xiang Zhang and Yann LeCun. 2015. Text understanding from scratch. *arXiv preprint arXiv:1502.01710* (2015).

[25] Wenyi Zhou, Elizabeth Simpson, and Denise Pinette Domizi. 2012. Google Docs in an Out-of-Class Collaborative Writing Activity. *International Journal of Teaching and Learning in Higher Education* 24, 3 (2012), 359–375.