Squabble: an efficient, scalable controversy classifier

Shiri Dori-Hacohen, Elinor Brondwine, and Jeremy Gollehon
AuCoDe
{firstname}@controversies.info

Abstract

We introduce Squabble, an efficient, scalable API for classifying controversial text such as news articles. Squabble is designed and implemented in python for commercial purposes with industry best practices, which can be followed by other researchers aiming to commercialize their innovations. We demonstrate multiple orders of magnitude speedup compared to prior work, while retaining effectiveness.

1 Introduction & Prior Work

The last few years have seen growing interest in computational analysis of controversies (cf. [GDFMGM17, MZDC14]). Recent work demonstrated a clear link between controversies and disinformation, demonstrating that polarizing topics are more vulnerable to fake news, and proposing controversy as a feature in prediction and classification of mis- and disinformation [VQSZ19], highlighting the importance of classifying controversy. Others explored the connection between controversy and sentiment, highlighting that they are related but not interchangeable constructs [KLF18, MZDC14]. Recent work generated unsupervised explanations of *what* makes a topic controversial [KA19]. Social good and commercial applications of detecting controversy include crisis management, public relations, and defense. Consider, for example, that recent mass shootings and terrorist attacks were preceded by activity on social media referencing highly controversial matters[1].

Controversy Language Models. Jang et al. described a framework for detecting controversy probabilistically [JFDHA16] and introduced a novel approach based on *controversy language models (CLM)*. CLM evaluates whether a document better matches a controversy vs. a non-controversy (or background) language model, relying on the following comparison: $\log P(D|C) - \log P(D|NC) > \alpha$. Here, $P(D|C)$ and $P(D|NC)$ represent the probabilities that a document was generated from a controversial or a non-controversial unigram language model, respectively. CLM can be estimated by constructing a collection of controversial documents; here, we refer to one of construction approaches, the Wikipedia Controversy Feature (WCF) [JFDHA16], which uses the top K Wikipedia articles that have high controversy scores. Once trained, the classifier no longer relies on Wikipedia for its success; rather, the language model trained from Wikipedia can be applied to any text, whether or not it discusses topics covered in Wikipedia.

2 The Squabble API

As is common in research labs, the original CLM code in Java was built with much attention paid to effectiveness, but little paid to efficiency and organization. As an early-stage startup, we wanted to provide our research team the ability to iterate their models quickly on large data sets without waiting days for answers - a situation that slowed our team down significantly. In order to bring CLM out of the research lab into a commercially viable product that could handle large volumes of text from different genres, we built a production-ready system in Python, dubbed "Squabble". We will describe two main elements relevant for efficiency: the technical infrastructure and a redesign of the CLM approach.

Technical infrastructure. We created a system that can ingest, filter, and store raw data from a wide variety of sources such as news and social media. As an initial testbed and to stress-test our infrastructure, we used a 10 year history of the Twitter Gardenhose collection - a random 1% sample of all tweets from 2008-2018 [OBRS10]. We created a multi-threaded Python

In: A. Aker, D. Albakour, A. Barrón-Cedeño, S. Dori-Hacohen, M. Martinez, J. Stray, S. Tippmann (eds.): Proceedings of the NewsIR'19 Workshop at SIGIR, Paris, France, 25-July-2019, published at http://ceur-ws.org

[1]https://www.npr.org/2019/03/15/703911997/the-role-social-media-plays-in-mass-shootings

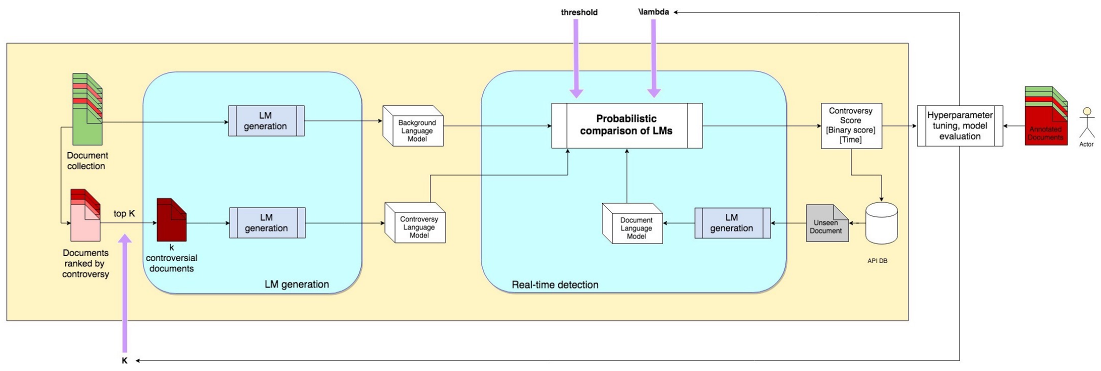
threshold
\lambda
Document collection
top K
Documents ranked by controversy
k controversial documents
LM generation
LM generation
LM generation
Background Language Model
Controversy Language Model
Probabilistic comparison of LMs
Document Language Model
LM generation
Unseen Document
Real-time detection
Controversy Score [Binary score] [Time]
API DB
Hyperparameter tuning, model evaluation
Annotated Documents
Actor
K

Figure 1: The Squabble API architecture. See Figure 2 for a zoomed in version of the "LM Generation" process.

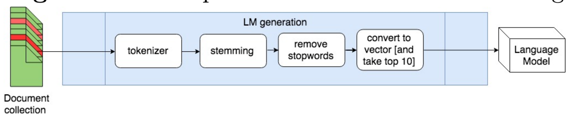

Figure 2: Procedure for generating Language Models.

program storing data in a PostgreSQL database hosted on Amazon Web Services RDS system. We added a component that extracts both the tweet text as well as any externally linked article text when a link was included in the tweet. We kept the filtering stage simple yet flexible, accepting as input a text file with a list of keywords or hashtags of interest. Tweets containing any of the keywords or tags are included in the database.

CLM revisited. We reimplemented and refined the controversy detection algorithm described in Section 1, with system architecture is presented in Figure 1. We used established Python packages such as NLTK [LB02] and Scikit-learn [PVG$^+$11], and created a research testbed in order to evaluate the Squabble API. We describe evaluation details in Section 4.

Squabble accepts data with a text stream via SQL queries or CSV files. In pilot efforts, prospective customers sent data via large CSV files which we were able to ingest and run through Squabble rapidly. Prior to creating Squabble, such pilots would take days to run, slowing development down. Like our data processing code, the Squabble code can likewise scale via multi-threading. In addition, Squabble can be applied in a wide variety of verticals, such as finance, defense, and public relations. We constructed Squabble explicitly to allow for that possibility. As an early-stage startup, this also gives us flexibility to pivot easily should the need arise, without expensive retooling of the technology.

3 Efficiency improvements

Prior to commencing this project, we set internal success metrics for efficiency (see Table 1) that we estimated would allow us to successfully process customer requests and internal research at scale for the foreseeable future. Processing a massive data set into a structured database, on a budget, turned out to be the biggest core technological hurdle in the infrastructure portion of our system. PostgreSQL's concurrency behaviour couldn't handle the amount of data being sent. Once the underlying issue was resolved, data storage speed immediately increased by multiple orders of magnitude, not only meeting our success metric but also exceeding our most ambitious projections for speed, as seen in Table 1. Since this code was structured for scalability, we can add multi-threading or multi-processing with no additional effort. As seen in Table 1, we easily met our success metrics and achieved orders of magnitude speedup.

Table 1: Before, After & Success Metrics for Squabble. Infrastructure speeds reported on a per core basis on a server. Controversy scoring reported on a dual core laptop.

	Infrastructure	Controversy Scoring
Before	100 tweets/sec/core	7 requests/sec
Success metric	1,000 tweets/sec/core	500 requests/sec
After	100,000 tweets/sec/core	700 requests/sec
Speedup	1000x	100x

Table 2: Dataset from prior work [DHA15] with key statistics

	# Docs (%)	Terms: Mean	Terms: Std
Controversial	78 (25.75%)	828.43	1159.78
Non-controversial	225 (74.25%)	367.1	564.11
All	303 (100%)	485.86	787.28

4 Evaluation

We leverage the dataset introduced by Dori-Hacohen and Allan [DHA15] that consists of judgments for 303 webpages[2] from the ClueWeb09 collection[3], which is presented in Table 2. The evaluation set is imbalanced, in the sense that it contains more non-controversial (225 documents) that controversial documents (78 documents). Therefore, we focused on balanced accuracy as our metric of choice against several baselines such as sentiment and Naive Bayes, and we also report other metrics for completeness' sake (see Table 3). Our results with the WCF approach were in-line with prior

[2]`http://ciir.cs.umass.edu/downloads`
[3]`http://lemurproject.org/clueweb09/`

Table 3: Classification scores for Squabble compared to several baselines. Squabble outperforms in all metrics evaluated (other than recall, which a trivial baseline accomplishes by definition).

	B. Acc.	R	Acc.	P	F1
Squabble score	0.876	0.955	0.835	0.600	0.737
Sentiment	0.476	0.909	0.253	0.233	0.370
Random	0.545	0.727	0.451	0.267	0.390
MultinomialNB	0.816	0.864	0.791	0.543	0.667
All controversial	0.500	1.000	0.242	0.242	0.389
None controversial	0.500	0.000	0.758	NaN	NaN

work [JFDHA16], demonstrating reproducibility from the original paper[4], and also show that effectiveness was not sacrificed for the sake of efficiency.

5 Conclusion

We presented Squabble, a robust, commercially-viable API for controversy classification, which is efficient and scalable. Squabble can be applied in a vertical-agnostic manner. By reimplementing the controversy language model [JFDHA16] in python using industry best practices, we increased its efficiency by orders of magnitude, without sacrificing effectiveness. Efficiency and scalability position Squabble to be used in commercial settings for a wide variety of applications. Its modularity and research testbed allow for extensibility and improvement in the future, as more effective methods of classifying controversy are discovered.

Limitations & Future Work. The dataset from prior work is somewhat limited [DHA15]; length between controversial and non-controversial documents is inconsistent (see Table 2). It is possible that CLM's effectiveness improves by biasing longer documents. Foley showed that AUC on this dataset has been maximized compared to its inter-annotator agreement [Fol18]. In future work, we hope to create new ground truth data sets for controversy, and encourage other researchers to do the same. Future work will scale our architecture to run concurrently on multiple servers and speed up controversy scoring further. More work is needed to understand the connection between controversy, mis- and dis- information.

Acknowledgements. This material is based upon work supported by the National Science Foundation under Grant No. 1819477. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[DHA15] Shiri Dori-Hacohen and James Allan. Automated controversy detection on the web. In *European Conference on Information Retrieval*, pages 423–434. Springer, 2015.

[Fol18] John Foley. Explainable agreement through simulation for tasks with subjective labels. *arXiv preprint arXiv:1806.05004*, 2018.

[GDFMGM17] Kiran Garimella, Gianmarco De Francisci Morales, Aristides Gionis, and Michael Mathioudakis. Reducing controversy by connecting opposing views. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 81–90. ACM, 2017.

[JFDHA16] Myungha Jang, John Foley, Shiri Dori-Hacohen, and James Allan. Probabilistic approaches to controversy detection. In *Proceedings of the 25th ACM international on conference on information and knowledge management*, pages 2069–2072. ACM, 2016.

[KA19] Youngwoo Kim and James Allan. Unsupervised explainable controversy detection from online news. In *European Conference on Information Retrieval*, pages 836–843. Springer, 2019.

[KLF18] Kateryna Kaplun, Christopher Leberknight, and Anna Feldman. Controversy and sentiment: An exploratory study. In *Proceedings of the 10th Hellenic Conference on Artificial Intelligence*, page 37. ACM, 2018.

[LB02] Edward Loper and Steven Bird. NLTK: the natural language toolkit. *arXiv preprint cs/0205028*, 2002.

[MZDC14] Yelena Mejova, Amy X Zhang, Nicholas Diakopoulos, and Carlos Castillo. Controversy and sentiment in online news. *arXiv preprint arXiv:1409.8152*, 2014.

[OBRS10] Brendan O'Connor, Ramnath Balasubramanyan, Bryan R Routledge, and Noah A Smith. From tweets to polls: Linking text sentiment to public opinion time series. In *Fourth International AAAI Conference on Weblogs and Social Media*, 2010.

[PVG$^+$11] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.

[VQSZ19] Michela Del Vicario, Walter Quattrociocchi, Antonio Scala, and Fabiana Zollo. Polarization and fake news: Early warning of potential misinformation targets. *ACM Trans. Web*, 13(2):10:1–10:22, March 2019.

[4]Details omitted for space considerations.