arXiv:2001.01819v1 [cs.CL] 7 Jan 2020

RECAST: Interactive Auditing of Automatic Toxicity Detection Models

Austin P Wright
Georgia Institute of Technology
apwright@gatech.edu

Muhammed Ahmed
Mailchimp
muhammed.ahmed@mailchimp.com

Omar Shaikh
Georgia Institute of Technology
oshaikh@gatech.edu

Stephane Pinel
Mailchimp
stephane.pinel@mailchimp.com

Haekyu Park
Georgia Institute of Technology
haekyu@gatech.edu

Diyi Yang
Georgia Institute of Technology
diyi.yang@cc.gatech.edu

Will Epperson
Georgia Institute of Technology
willepp@gatech.edu

Duen Horng (Polo) Chau
Georgia Institute of Technology
polo@gatech.edu

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

Copyright held by the owner/author(s).
CHI'20,, April 25–30, 2020, Honolulu, HI, USA
ACM 978-1-4503-6819-3/20/04.
https://doi.org/10.1145/3334480.XXXXXXX

Abstract

As toxic language becomes nearly pervasive online, there has been increasing interest in leveraging the advancements in natural language processing (NLP), from very large transformer models to automatically detecting and removing toxic comments. Despite the fairness concerns, lack of adversarial robustness, and limited prediction explainability for deep learning systems, there is currently little work for auditing these systems and understanding how they work for both developers and users. We present our ongoing work, RECAST, an interactive tool for examining toxicity detection models by visualizing explanations for predictions and providing alternative wordings for detected toxic speech.

Author Keywords

Machine Learning Fairness; Interactive Visualization; Algorithmic Bias; Natural Language Processing

CCS Concepts

•Human-centered computing → Visualization application domains;

Introduction

There is a growing desire to moderate and remove toxic language from social media and public forums, a result of increasing online interactions [7]. Due to the high volume

of interaction, manual moderation is often not tractable, leading to the development of automatic toxicity detection models such as the Google Perspective API [2].

Although there has been little work on understanding the function and impact of toxicity detection systems specifically, various interactive tools have been built for understanding the internal mechanisms of natural language processing (NLP) systems. Some tools such as SANVis and exBert allow for interactive exploration of attention mechanisms in transformer models [12, 8]. Other tools, like Errudite, allow users to view hypotheses with respect to the true error distribution on the entire dataset [20]. Some techniques attempt to adversarially perturb language input to a model [21, 16], with some methods using a human-in-the-loop design [9]. Currently deployed models also contain inherent biases towards certain subgroups; some research aims to expose gender [22] and racial biases [10, 3] in NLP models and their underlying datasets [15]. Some widely deployed NLP models, specifically BERT, also tend to overlook simple linguistic structures like negation, reducing their effectiveness [6]. This existing body of research highlights major potential flaws in the underlying language models for these systems. Specifically, the models can be:

1. Limited in their understanding of some basic linguistic structures (like negation).
2. Susceptible to adversarial examples that can fool the model, allowing bad actors to introduce toxic language into the discourse.
3. Vulnerable to social biases present in the training dataset leading to a high risk for biased toxicity heuristics against marginalized groups.

These flaws pose critical challenges for both developers and end users. Developers of toxicity detection systems may find it harder to understand what linguistic features their models have learned; without a good understanding, it would be challenging for them to improve model performance, or to fix potential erroneous model outputs. Similarly, users of online forums that use black-box NLP models might question how their language is being examined.

We address these challenges by developing an interactive tool called **RECAST**, which allows for the interactive interrogation of toxicity detection models through counterfactual alternative wording and attention visualization. RECAST currently supports analysis of a fine-tuned BERT model on the Jigsaw Toxicity dataset [1], and it may be extended to other attention-based models or alternative datasets. Our ongoing work presents the following contributions:

1. **RECAST**, an interactive system allowing users to dynamically interrogate the classification of text toxicity by visualizing its sources in a piece of text and examining alternative wordings. Despite the prevalent use of online moderation tools, users lack the ability to interrogate what constitutes toxicity according to these models — a problem our RECAST aims to tackle.
2. **Preliminary Experimental Results**: using RECAST we performed a qualitative analysis of noteworthy use-cases for interrogating toxicity models, including highlighting linguistic features that are detected (and missed) by toxicity models.
3. **Preliminary Examples of Bias**: using RECAST, we show examples of biased classification of toxic language using current state of the art classification models. Furthermore, we propose functionality to flag possibly biased misclassifications that can be found when interrogating these models.

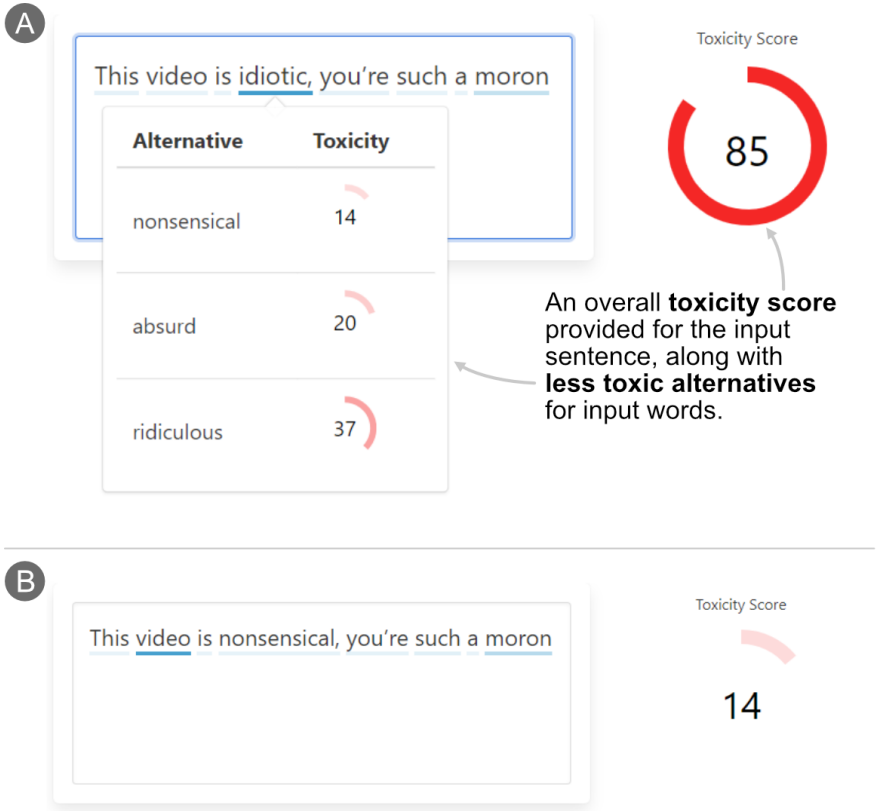

Figure 1: A: RECAST consists of a textbox and a radial progress bar. A color change on the radial progress, along with a score, indicate the toxicity of a sentence. Toxicity ranges from white (non-toxic) to red (very toxic). Users can hover over options to preview toxicity scores for replacing the selected word in the sentence. **B**: upon replacing the word (in the case of this figure, replacing "idiotic" with "nonsensical"), the main radial progress bar reflects the reduced toxicity score. However the small attention on the other pejorative word "moron" compared to "video" in the alternative version shows the idiosyncrasies of the model and underlying dataset.

RECAST

RECAST is an interactive system (Figure 1) that allows users to input text data and view the toxicity of the overall sentence, along with which words most contribute to output score. RECAST outputs a score between 0 and 100, from least toxic to most toxic. Users can edit the textbox field and watch the toxicity score dynamically update. In general, users will enter a phrase or sentence into the main textbox and view their initial toxicity score. After hovering over words responsible for the output, users select alternative words that reduce their toxicity score.

System Design and Implementation

Dataset The dataset we used for training the backend model was sourced by Google's Jigsaw; a similar dataset is used by Google's Perspective API [1]. The Perspective API is used as a content moderation tool; therefore, its underlying dataset was a suitable proxy for RECAST's goal. We also chose to use this dataset because it was pre-cleaned, openly available, and already used as a benchmark for detecting hate-related speech online. The dataset consists of a set of sentences along with multiple labels that characterise the form of toxicity. We chose to only use the toxicity label in the dataset for modeling purposes, as the other labels (identity hate, insult, etc.) were subsets of toxicity. Because our objective is to highlight and reduce overall toxicity, we chose to ignore these subsets in RECAST. A noteworthy limitation of this dataset is its focus on explicit hate speech, or speech that directly insults through the use of particular hateful keywords (like the words "idiotic" and "moron," as seen in Figure 1). Implicit hate speech, however, tends to focus stereotypes, avoiding explicit phrases (e.g. "you're smart for a girl"). Future work on collecting implicit hate speech could extend RECAST to support such examples.

Transformer Architecture Currently, our work utilizes DistillBERT, a lighter version of the full BERT model that retains 97% of the original model's understanding capacity with substantially fewer parameters [14]. Furthermore, transformer models utilize self-attention, which can be used as a proxy for explanation [18]. Transformer models also utilize context through their use of self-attention, which makes these models far more effective than using a simple dictionary of hateful words. Although there are a wide range of models we could have used, such as those proposed in the original 2017 Jigsaw Kaggle competition [1], we decided to utilize transformer models due to their recent prevalence in most NLP tasks, as RECAST aims to be generally useful for current models.

Backend Server and Frontend Client Our backend uses PyTorch [13], a popular deep learning library, and the transformers package [19]. Despite the current dependence on transformer based models, we designed RECAST to be model agnostic, so the backend's model can easily be swapped given that it provides some form of explanation for its output prediction (i.e. in the case of a transformer, we use attention). Our frontend was written with modern web development technologies, using Svelte and D3 for various visualizations. We wrote RECAST to be mobile-ready: because predictions occur on the backend, the client itself requires very little compute power.

Attention Visualization
Since we use attention to explain our model's choices, we needed to visualize the attention on the words themselves. Self-attention is pairwise and occurs across multiple attention-heads in transformers. For RECAST, we needed to reduce the attention to a single representative head and display this attention on a single word instead of a pair. To do this, we only analyzed the average of the attention heads on the last layer, as later layers tend to focus on higher level semantics [5]. We also dropped special tokens that indicate the start ([CLS]) and end of a sentence ([SEP]). Finally we considered various visualisation concepts that can be used to show the relative importance of words such as highlighting and text opacity [17]. We utilized an underline on every word, where the opacity of each underline would be controlled by attention placed on each word. We found that using an underline instead of adjusting the opacity of the word itself helped with visibility.

For example, our model labels the sentence "I hate you" with an overall toxicity score of 70, mostly deriving from the word "I" (the word "hate" alone is non-toxic; addition of a subject that actively hates makes this phrase toxic). Thus, our underlining looks like: "I hate you". Another example, which we later revisit, can be seen in Figure 1.

Alternative Wording
Alternative wording provides users with options to swap or delete words in a sentence that are responsible for high toxicity scores. We experimented with a set of methods to find viable alternatives for toxic words in a sentence. All the provided methods use a human-in-the-loop design, allowing users to select choices that provide a toxicity tradeoff. Figure 1 highlights such a tradeoff in RECAST.

Our current method for alternative wording starts by selecting all highly attended words and suggesting substitutions with words that reduce toxicity the most. Candidate substitutions are selected and ranked by using the k-nearest words from Word2Vec embeddings [11]. We discuss an extension to this method in our ongoing work section.

Preliminary Results

Here, we detail two usage scenarios from the perspective of two different users: a user wishing to see how their lan-

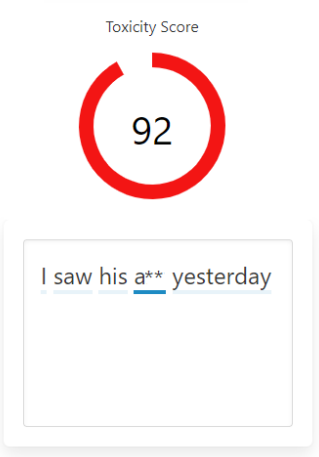

Figure 2: A case where a developer might see biases in their model with respect to dialects. The language shown above is non-toxic in the AAE dialect, yet RECAST shows a fairly high toxcitiy score.

guage is understood by AI, and a developer who wishes to see biases or shortcomings in their model.

Usage Scenarios

User Facing Consider a user who is confused about why his toxic comment is being flagged on a popular social media website. Perhaps the user wants to explore alternatives that are less toxic in nature, or see exactly what about his comment raised the automoderator's concerns. The user can copy or retype their sentence into the textbox (shown in Figure 1) while viewing the toxicity score. As they type, they can see the underlined attention on each word.

The user notes that the model is paying the most attention to the insults in their input. As they hover over the offending phrase, RECAST suggests alternative wordings. In the case of Figure 1, RECAST suggests deleting the offending word, or substituting it with a less offensive alternative. Because of this, the user comes away with a stronger understanding of what the automoderator is looking for. After selecting an alternative, the user submits their comment again, which results in an acceptance.

In this situation, a user gets immediate input from what might initially appear to be a black box, which results in an increased level of transparency in online communities.

Developer Facing Developers can use RECAST as an end-to-end summary of the model's evaluation stage, similar to an integration test. By running various inputs on their model, developers can see what underlying biases and shortcomings the model has inadvertently learned. In this scenario, a developer may be looking for underlying biases in their training data regarding African American English (AAE) dialects, similar to what Sap et al. explores [15]. Figure 2 highlights the attention and toxicity scores from a sample input that would generally be considered non-toxic within the AAE dialect. However, the attention visualization shows the model focusing on the expletive without regard to the dialect.

In this scenario, the developer comes away with a newfound understanding of the model: it tends to focus on more obvious scenarios. Subtle characteristics (like dialects) affect toxicity, which the developer's model tends to ignore.

Another interesting example involves gender biases. A developer could query sentences that differ in gender to probe for biases and check the corresponding toxicity scores. An example is shown in Figure 3.

Ongoing Work and Conclusion

Sentence Semantic Similarity

Although word embeddings provide words that appear near our original input, we found that the substitutions in our current method occasionally make little grammatical sense. Initial experimentation indicated that using synonyms (from a database like WordNet) instead of k-nearest word embeddings helped preserve our sentence semantics. To this end, we also plan to limit substitutions to specific parts of speech, namely nouns, adjectives, and verbs. Occasionally, the meaning of an alternate sentence deviates too far from the original. To avoid this, we also plan to add a secondary constraint besides toxicity: embeddings from the Universal Sentence Encoder [4]. Our planned method will attempt to optimize for increased similarity between the Universal Sentence Encoder's embeddings of the changed and original sentence, while decreasing toxicity score. We also plan to quantitatively evaluate the performance of our algorithms.

User Feedback and Model Retraining

Because developers or users may note shortcomings in the model, we aim to include a feature that allows flagging

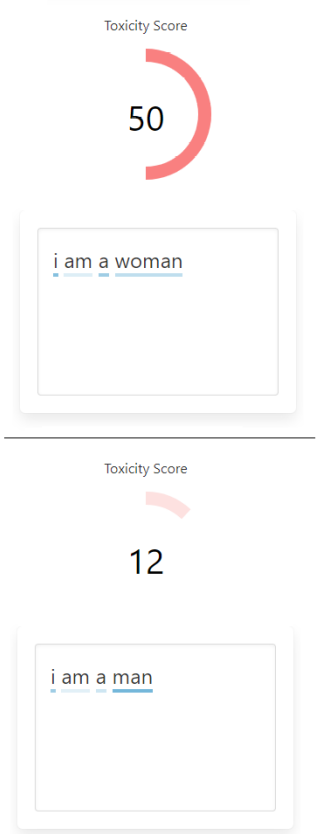

Figure 3: Another case where a developer might see biases in their model with respect to gender. The language shown above has a lower toxicity score when the gender is changed from woman to man.

of such examples instead of simply ignoring them. These examples can be used to provide researchers with data for retraining. Developers can then create a train-test cadence, where their model retrains at a set interval from flagged examples.

Open Source + User Studies

We also aim to repackage RECAST into an open-source browser extension and analyze usage behaviour. Specifically, we aim to explore the following presence-related questions: in what situations do users accept a less toxic alternative for their input, and are such situations dependent on the forum or community a user visits?

Ethical Concerns

In some cases, RECAST brings up counter-examples that are as offensive as the original sentence, but have a lower toxicity score. Given this side-effect, users can potentially view RECAST as an tool to generate adversarial attacks. While this risk of dual-use must be noted, we believe RECAST is justified due to its ability to help detect shortcomings in existing models, and thereby help improve them.

Planned Evaluation

Besides evaluating RECASTâĂŹs features, we plan to recruit NLP and practitioners to evaluate the efficacy of the system as a whole in auditing and improving toxicity detection models. Furthermore we plan to test the degree to which nontechnical end users will be able to use RECAST to understand the linguistic features learned by these models, and evaluate how they may change their own language in response.

Conclusion

RECAST takes steps towards increased transparency for black-box NLP models that are responsible for moderating large swaths of the internet. By enabling users to interact with text input, view alternative wordings for toxic sentences, and identify potential biases, RECAST provides insights into these models. Our future steps involve adding a feedback and retraining feature, releasing and studying usage for an open-source browser extension, and building upon our alternative wording algorithm.

Acknowledgements

This work was supported in part by NSF grants IIS-1563816, CNS-1704701, NASA NSTRF, gifts from Intel (ISTC-ARSA), NVIDIA, Google, Symantec, Yahoo! Labs, eBay, Amazon.

REFERENCES

[1] 2017. Jigsaw Toxicity Dataset. https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge. (2017).

[2] 2017. Perspective API. https://www.perspectiveapi.com/. (2017).

[3] Tolga Bolukbasi, Kai-Wei Chang, James Zou, Venkatesh Saligrama, and Adam Kalai. 2016. Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings. (2016).

[4] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018. Universal Sentence Encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Brussels, Belgium, 169–174. DOI: http://dx.doi.org/10.18653/v1/D18-2029

[5] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What Does BERT Look

at? An Analysis of BERTâĂŹs Attention. *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP* (2019). DOI:http://dx.doi.org/10.18653/v1/w19-4828

[6] Allyson Ettinger. 2019. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. (2019).

[7] Paula Fortuna and Sérgio Nunes. 2018. A survey on automatic detection of hate speech in text. *ACM Computing Surveys (CSUR)* 51, 4 (2018), 85.

[8] Benjamin Hoover, Hendrik Strobelt, and Sebastian Gehrmann. 2019. exBERT: A Visual Analysis Tool to Explore Learned Representations in Transformers Models. (2019).

[9] Brandon Laughlin, Christopher Collins, Karthik Sankaranarayanan, and Khalil El-Khatib. 2019. A Visual Analytics Framework for Adversarial Text Generation. (2019).

[10] Thomas Manzini, Lim Yao Chong, Alan W Black, and Yulia Tsvetkov. 2019. Black is to Criminal as Caucasian is to Police: Detecting and Removing Multiclass Bias in Word Embeddings. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 615–621. DOI: http://dx.doi.org/10.18653/v1/N19-1062

[11] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2 (NIPSâĂŹ13)*. Curran Associates Inc., Red Hook, NY, USA, 3111âĂŞ3119.

[12] Cheonbok Park, Inyoup Na, Yongjang Jo, Sungbok Shin, Jaehyo Yoo, Bum Chul Kwon, Jian Zhao, Hyungjong Noh, Yeonsoo Lee, and Jaegul Choo. 2019. SANVis: Visual Analytics for Understanding Self-Attention Networks. (2019).

[13] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.). Curran Associates, Inc., 8024–8035.

[14] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. (2019).

[15] Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A. Smith. 2019. The Risk of Racial Bias in Hate Speech Detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, 1668–1678. DOI: http://dx.doi.org/10.18653/v1/P19-1163

[16] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. 2019. How can we fool LIME and SHAP? Adversarial Attacks on Post hoc Explanation Methods. (2019).

[17] Hernan Valdivieso, Denis Parra, Andres Carvallo, Gabriel Rada, Katrien Verbert, and Tobias Schreck. 2019. Analyzing the Design Space for Visualizing Neural Attention in Text Classification. (2019). https://tinyurl.com/yzt866vb

[18] Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not Explanation. (2019).

[19] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *ArXiv* abs/1910.03771 (2019).

[20] Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel Weld. 2019. Errudite: Scalable, Reproducible, and Testable Error Analysis. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, 747–763. DOI:http://dx.doi.org/10.18653/v1/P19-1073

[21] Wei Emma Zhang, Quan Z. Sheng, Ahoud Alhazmi, and Chenliang Li. 2019. Adversarial Attacks on Deep Learning Models in Natural Language Processing: A Survey. (2019).

[22] Jieyu Zhao, Tianlu Wang, Mark Yatskar, Ryan Cotterell, Vicente Ordonez, and Kai-Wei Chang. 2019. Gender Bias in Contextualized Word Embeddings. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 629–634. DOI: http://dx.doi.org/10.18653/v1/N19-1064