CRYPTEXT: Database and Interactive Toolkit of Human-Written Text Perturbations in the Wild

Thai Le* Yiran Ye Yifan Hu** Dongwon Lee

University of Mississippi* The Pennsylvania State University Yahoo Inc.**

thaile@olemiss.edu* {yby5204,dongwon}@psu.edu yifanhu@yahooinc.com**

***Abstract*—User-generated textual contents on the Internet are often noisy, erroneous, and not in correct grammar. In fact, some online users choose to express their opinions online through carefully *perturbed* texts, especially in controversial topics (e.g., politics, vaccine mandate) or abusive contexts (e.g., cyberbullying, hate-speech). However, to the best of our knowledge, there is no framework that explores these online "human-written" perturbations (as opposed to algorithm-generated perturbations). Therefore, we introduce an interactive system called CRYPTEXT. CRYPTEXT is a data-intensive application that provides the users with a database and several tools to extract and interact with human-written perturbations. Specifically, CRYPTEXT helps look up, perturb, and normalize (i.e., de-perturb) texts. CRYPTEXT also provides an interactive interface to monitor and analyze text perturbations online. The demo is available at: https://lethaiq.github.io/anthro.**

***Index Terms*—perturbations, database, machine learning**

I. INTRODUCTION

On popular social platforms such as Twitter, Reddit or Facebook, users sometimes slightly change the spelling of a text, without changing its original semantic meaning, for various reasons. For instance, the following sentences contain real perturbed texts (underlined): *"The democRATs responsible for their attempted race war," "A fake tree burned and RepubLIEcans are calling for,"* or *"Thinking about suic1de."* These perturbations are often users' deliberate attempts to emphasize their opinions implicitly or explicitly (e.g., democRATs and RepubLIEcans), or to censure offensive or sensitive wording to avoid platforms' censorship (e.g., suic1de). Moreover, we also found that religion or nationality related perturbations (e.g., "muslim"→"mus-lim" and "chinese"→"chi-nese") often appear in the context of cyberbullying, racism, or hatespeech on platforms.

These human-written perturbations are a useful resource for both researchers and practitioners as they provide inductively-derived (i.e., observable) attacks by humans, as opposed to deductively-derived (i.e., hypothesized) adversarial attacks by machine learning (ML) algorithms. Although such human-written perturbations occur frequently online, however, there has been no tool that helps discover, study, and analyze these human-written perturbations. Therefore, in this work, we propose CRYPTEXT, a system that detects and extracts human-written perturbations from social platforms. The contributions of CRYPTEXT can be summarized as follows:

- CRYPTEXT is uniquely equipped with a dictionary of over 2M human-written tokens that are categorized into over 400K unique phonetic sounds. CRYPTEXT is constantly learning new perturbations from social platforms of Reddit and Twitter, providing users with up-to-date perturbations. CRYPTEXT also enables the users to discover perturbations from input texts and normalize or de-perturb them (i.e., correct perturbed texts).
- CRYPTEXT also facilitates the evaluation of the robustness of textual ML models on noisy user-generated contents by perturbing inputs with realistic human-written (and not machine-generated) perturbations. CRYPTEXT also helps analyze the usage patterns of human-written perturbations and their associated sentiments online.
- CRYPTEXT is also beneficial for a variety of users and audiences including NLP researchers, linguists and social scientists. CRYPTEXT will be open-sourced.

II. LITERATURE REVIEW

A. Definition of Text Perturbations

Previous research on the use of text perturbations focused only on *machine-generated* adversarial texts. Previous literature often proposed algorithms to *minimally* manipulate an input text x to generate an adversarial text x' that can fool a target ML model $f(\cdot)$ such that $f(x')\neq f(x)$ and x' still preserves the semantic meaning of x as much as possible. Then, we can define *text perturbations* of x (which are different from adversarial texts), regardless of whether they are generated by machine or written by human, as a collection of manipulated texts $P_x=\{x'_1, x'_2, ..x'_N\}$ yet ***without*** the adversarial condition (i.e., $f(x') \neq f(x)$). From now on, we will use the notation $x' \in P_x$ to denote a perturbation (and *not* an adversarial example) of x. Moreover, we specifically focus on character-level perturbations since they are intuitive to humans.

B. Machine-Generated Perturbations

There are several automatic character-level text manipulation algorithms proposed in the adversarial NLP literature. These manipulation strategies include swapping, deleting a character in a word (e.g., "democrats"→"demorcats", TextBugger [1]), replacing a character by its most probable misspell (e.g., "republicans"→"rwpublicans", TextBugger [1]), replacing a character by another visually similar digit or symbol (e.g., "democrats"→"dem0cr@ts" TextBugger [1]),

Fig. 1. *Look Up* function displays human-written perturbations for the input token "amazon" in a form of an interactive 3D spherical word-cloud.

an *accent* (e.g., "democrats"→"dėmocrāts", VIPER [2]), or a *homoglyph* (DeepWordBug [3]). Although these strategies are shown to be very effective in flipping textual ML models, they are hypothesized based on some known vulnerabilities of a target textual model $f(\cdot)$, and *not inductively* derived from actual real-life observations. Thus, their applications are limited only to adversarial NLP settings.

C. Human-Written Perturbations

From analyzing several public text corpus, we observe that machine-generated and human-written perturbations express very different rules and strategies. For example, humans often add another layer of meaning on words by emphasizing a part of it (e.g., "democRATs", "repubLIEcans"). Moreover, hyphenation is also used to perturb a word (e.g., "mus-lim", "vac-cine"). Different from machines, humans perturb words by using emoticons, repeating characters (e.g., "porn"→"porrrrn"), and using phonetically similar characters (e.g., "depression"→"depresxion"). Human's manipulation strategies can also be very creative and require some implicit context to understand (e.g., "democRATS") beyond using a simple set of rules. Often, human-written perturbations also do not manifest any fixed rules. Hence, it is very challenging for an algorithm to systematically generate all human-written perturbations. As these strategies become more prevalent online, conventional ML models that often assume clean English input texts will face difficulties with noisy texts in real-life, especially models that deal with controversial topics (e.g., politics, vaccine mandate), or abusive contents (e.g., cyberbullying, hate-speech).

III. Demonstration of CrypText System

As the first step toward understanding human-written perturbations online, we design and propose the CrypText that discovers, utilizes and monitors human-written perturbations online. CrypText has four main functions: (1) looking up perturbations, (2) normalizing perturbations, (3) manipulating texts using human-written perturbations, and (4) visualizing and analyzing online contents through the lens of text perturbations. All functions are provided via a graphical-user-interface (GUI) on top of a database (DB) system and a set of public APIs. We elaborate each function and its use cases below.

Key	Value
TH000	{the, thee}
DI630	{dirty, dirrrty}
RE4425	{republicans, repubLIEcans, republic@@ns}

TABLE I
Example of extracted hash-map $\mathcal{H}_1$ from a corpus of two sentences *"the dirrty republicans"*, *"thee dirty repubLIEcans"*, *"the dirty republic@@ns"*

A. Human-Written Token Database

CrypText relies on the database of human-written tokens of both correctly-spelled English words and their potential perturbations. This database stores all of the raw case-sensitive tokens found from several public datasets. Most of them are originally curated for detecting abusive materials online (e.g., rumors [4], hatespeech [5], cyberbullying [6]) and often contain many perturbations. To curate the database, we first tokenize each sentence in the corpus, and encode their sounds using a customized version of the Soundex algorithm [7].

Soundex is a fuzzy matching algorithm that was originally developed to match last names of similar phonetic sounds, and it can also be found on search engines such as Lucene [8]. Given a token, Soundex first fixes the first character, then matches the remaining characters one-by-one following a pre-defined rule set (e.g., {b, f, p, v}→"1"). However, the original Soundex algorithm cannot recognize some visually-similar character manipulations that are frequently found in human-written perturbation texts (e.g., "l"→"1", "a"→"@", "S"→"5"). Moreover, fixing the first character can result in many mismatches (e.g., both "losbian" and "lesbian" share the same code "L215" yet have distinct sounds). To overcome these, we thus customize the Soundex algorithm to encode visually-similar characters the same. We also introduce the *phonetic level* parameter $\mathbf{k}$ and fix the first $\mathbf{k}+1(\mathbf{k}\geq 0)$ characters as part of the resulted encodings (e.g., "losbian"→"LO245", "lesbian"→"LE245" with $\mathbf{k}\leftarrow 1$). All the extracted data using the modified Soundex algorithxtm are then stored as several hash-maps $\mathcal{H}_{\mathbf{k}}$ with different values of $\mathbf{k}\leq 2$ in the database. Each key and value of $\mathcal{H}_{\mathbf{k}}$ is a unique Soundex encoding and a set of all tokens sharing the same encoding at the phonetic level $\mathbf{k}$. Table I shows an example of $\mathcal{H}_1$ extracted from a set of three sentences.

B. Look Up: Discovering Text Perturbations

Functionality. CrypText allows users to query to the DB a set of perturbations P_x for a specific token x in the form of an interactive 3D *word-cloud* (Figure. 1). CrypText also displays a list of returned perturbations in a tabular format with their respective edit distances with the query. To do this, CrypText relies on what we called the *SMS* property to characterize text perturbations. Specifically, we define a perturbation as a token that has phonetically similar Sound, is perceived with the same Meaning as its original word, yet has different Spelling. For example, we consider "demokRATs" as a perturbation of "democrats" because they spell differently yet sound the same, and human readers can perceive both with the same meaning, especially when placed in similar contexts

(e.g., *"Biden belongs to the democrats"* and *"Biden belongs to the demokRATs"*). However, there are no automatic mechanisms that can measure the semantic similarity between two words of different spellings, especially when misspelled tokens are often encoded as a out-of-vocabulary (OOV). To overcome this, we utilize their Levenshtein edit-distance $\mathbf{d}$ in addition to their similarity in SOUNDEX encodings as a proxy for semantic similarity. Intuitively, two tokens share the same semantic meaning if they phonetically sound the same and are separated by a sufficiently small Levenshtein distance. Given the example in Table I, a search using the query $x \leftarrow$"republicans" with $\mathbf{k}{=}1, \mathbf{d}{=}1$ results in $P_x \leftarrow$\{republicans, repubLIEcans\}. Figure 1 shows an example output of the *Look Up* function. We manually set $\mathbf{k}{=}1, \mathbf{d}{=}3$ by default. Advanced users will be able to adjust these parameters through a provided API.

Use Cases. One notable use case of *Look Up* is *keyword enrichment*, which provides additional queries to search for sensitive topics online. Specifically, it enables users to search for contents that otherwise could have been censured or unreachable with correctly-spelled keywords. For example, only 67% of the tweets found by Twitter's search API from Nov. 2021 using keyword "democrats" has negative sentiment, while that number is much higher of 87% if a search query also includes the perturbations of "democrats" from the *Look Up* function. We also observe the same results for "republicans" (66% v.s. 84%) and "vaccine" (46% v.s. 61%). We also found that most tweets that contain the perturbations of "vaccine" focus more on either pushing back vaccine mandate or raising concerns on its safety and effectiveness, compared with the ones that are not perturbed.

C. Normalization: Detecting and De-Perturbing Texts

Functionality. This function detects and corrects text perturbations from an input x by leveraging the SMS property described in the *Look Up* function. Specifically, for each i-th token x_i in x, CRYPTEXT retrieves a set of potential matching English words for x_i given hyper-parameter $\mathbf{k}, \mathbf{d}$. A token w^* is a valid perturbation if there exists an English word that shares the same sound encoding as x_i at phonetic level $\mathbf{k}$ (i.e., belong to the same hash-map $\mathcal{H}_\mathbf{k}$), within an upper-bound edit-distance $\mathbf{d}$. However, there can be several candidate English words that associate with the token x_i. In this case, CRYPTEXT further ranks them by approximating how they fit into their surrounding local context of x_i in x. Specifically, we utilize a large pre-trained masked language model G to calculate a *coherency* score. Intuitively, this score is calculated by utilizing the pre-trained power of a complex language model such as BERT [9] to calculate how likely w^* appears in the immediate context of x_i. Then, CRYPTEXT outputs the most probable *de-perturbed* candidate for each $x_i \in x$ on its GUI (Figure. 2). Advanced users can also retrieve all candidates w^* and their coherency scores via a provided API.

Use Cases. There are two notable uses cases, namely (1) de-noising inputs of textual ML models and (2) usage as an additional predictive signal for ML pipelines. Given that the majority of ML models are often trained only on clean English corpus, CRYPTEXT can be used to correct all possible human-written perturbations in the training corpus. Moreover, the presence of perturbations within a sentence can also inform potential adversarial behaviors from its writer, especially those offensive or controversial perturbations (e.g., cyberbullying, political discourse, vaccine mandates, spreading disinformation), as part of a ML pipeline.

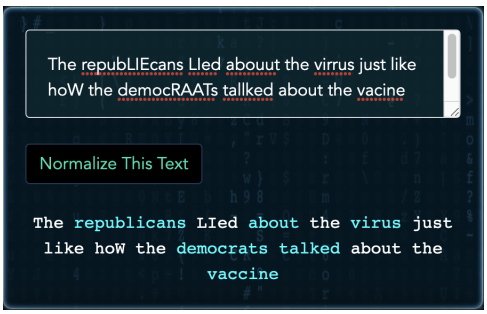

Fig. 2. *Normalization* function displays the normalized version of the input with corrected tokens being highlighted. The GUI also shows a popup describing a token before and after normalization.

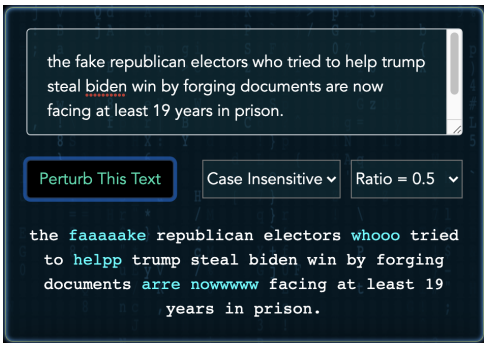

Fig. 3. CRYPTEXT perturbs an tweet with highlighted perturbations.

D. Perturbation: Text Manipulation using Human-Written Perturbations

Functionality. This function helps manipulate an input text using either case-sensitive or case-insensitive human-written perturbations at a user-specified manipulation ratio $\mathbf{r}$ (e.g., 15%, 25%, 50%). To do this, given an input text x, CRYPTEXT first randomly samples a subset of tokens in x according to the selected r value. Then, it iteratively replaces each of the selected tokens by a human-written perturbation that is randomly selected from the *Look Up* function's output (Figure. 3).

Use Case. Compared with other text perturbation methods in the literature (e.g., TextBugger [1], VIPER [2], DeepWordBug [3]), perturbations utilized by CRYPTEXT are guaranteed to be observable in human-written texts. This makes CRYPTEXT a more realistic tool to evaluate the robustness of textual ML models on noisy user-generated texts. This implication is applicable not only to NLP classification models, especially those that are built for abusive content moderation, but also on models developed for other tasks such as natural language inference and Q&A. In fact, many of the industrial APIs are

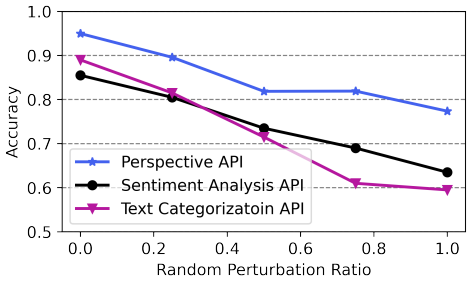

Fig. 4. Accuracy of Google's NLP APIs on texts perturbed by CRYPTEXT.

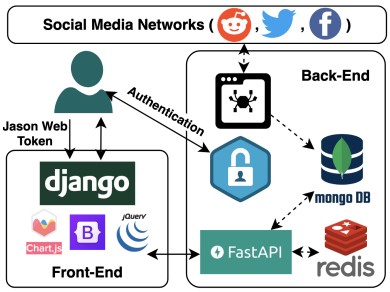

Fig. 5. An implementation of CRYPTEXT.

not well-equipped to deal with noisy human-written texts. Figure 4 shows that the *Perspective* toxic detection, sentiment analysis, and text categorization API from Google Cloud all suffer from texts perturbed by CRYPTEXT. Especially, the popular *Perspective* API's accuracy drops nearly 10% when only 25% of input texts are perturbed. CRYPTEXT also dedicates an *ML benchmark* page that frequently updates our evaluation of publicly available NLP APIs and models on noisy human-written texts.

E. Social Listening: Monitoring Human-Written Perturbations Online

Functionality. This function enables users to monitor the use of human-written perturbations online, especially on social platforms such as Reddit (via *PushShift API*[1]). Specifically, given a list of English words, CRYPTEXT first searches on the social platforms all the contents using their perturbations as queries. Then, it aggregates and displays the usage patterns of each individual perturbation in both frequency and sentiment through interactive timeline charts. We refer the readers to the attached video for the full visualization of this function.
Use Case. As described in the use case of *Look Up* function (Sec. III-B), using perturbations as queries help capture a more comprehensive view on a specific topic, especially one that is sensitive or divided. *Social Listening* helps achieve this by capturing contents that are often contrast or opposite to conventional opinions, which otherwise could be censured or unreachable on non-perturbed texts. Likewise, gatekeepers of social platforms also can utilize this function for better content moderation, especially in detecting and removing abusive texts on web (e.g., cyberbullying, racism, calling for violence and terrorism), many of which are often intentionally written with misspellings to evade automatic detection.

F. An Architectural Design

Figure 5 illustrates the architecture of the CRYPTEXT implementation. CRYPTEXT interacts with users via a GUI (i.e, website) and several function APIs. The back-end of CRYPTEXT is programmed with the *Django* and *FastAPI* framework in *python* language. Its front-end utilizes several powerful visualization tools such as *chart.js*, *TagCloud.js*, and *dataTables.js* to provide the end-users with an interactive experience. All functions of CRYPTEXT are equipped with secured public APIs, allowing users to utilize *Look Up*, *Normalization* and *Perturbation* in bulks. Accessing such APIs requires an authorization token that will be provided upon request. All of the data utilized by CRYPTEXT are stored in a MongoDB database. Since some queries might take a longer time to process, a *Redis cache* is adapted to temporarily store and re-use recent queried results to enhance the user experience. Furthermore, we set up a crawler that regularly collects recent tweets (via Twitter's public stream API[2]) to continually enrich CRYPTEXT's database with novel perturbed tokens online.

IV. LIMITATION AND CONCLUSION

Currently, the *Social Listening* function is limited to Reddit data and we plan to support other platforms in future. To the best of our knowledge, CRYPTEXT is the first interactive database system that helps discover and utilize human-written text perturbations online. Database provided by CRYPTEXT can also help evaluate the robustness of textual ML models on noisy human-written inputs and uncover sensitive online discussions. Especially, CRYPTEXT constantly updating its DB with new perturbations via Twitter's stream API.

REFERENCES

[1] J. Li, S. Ji, T. Du, B. Li, and T. Wang, "TextBugger: Generating Adversarial Text Against Real-world Applications," *NDSS'18*, 2018.
[2] S. Eger, G. G. Şahin, A. Rücklé, J.-U. Lee, C. Schulz, M. Mesgar, K. Swarnkar, E. Simpson, and I. Gurevych, "Text processing like humans do: Visually attacking and shielding NLP systems," in *NAACL'19*, Minneapolis, Minnesota, Jun. 2019, pp. 1634–1647. [Online]. Available: https://aclanthology.org/N19-1165
[3] J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi, "Black-box generation of adversarial text sequences to evade deep learning classifiers," in *SPW'18*. IEEE, 2018.
[4] E. Kochkina, M. Liakata, and A. Zubiaga, "All-in-one: Multi-task learning for rumour verification," in *ACL'18*. Santa Fe, New Mexico, USA: ACL, Aug. 2018. [Online]. Available: https://www.aclweb.org/anthology/C18-1288
[5] R. Gomez, J. Gibert, L. Gomez, and D. Karatzas, "Exploring hate speech detection in multimodal publications," in *WACV'20*, 2020, pp. 1470–1478.
[6] E. Wulczyn, N. Thain, and L. Dixon, "Wikipedia talk labels: Personal attacks," Feb 2017.
[7] C. Stephenson, "The methodology of historical census record linkage: A user's guide to the soundex," *Journal of Family History*, vol. 5, no. 1, pp. 112–115, 1980.
[8] A. Białecki, R. Muir, G. Ingersoll, and L. Imagination, "Apache lucene 4," in *SIGIR 2012 workshop on open source information retrieval*, 2012, p. 17.
[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *NAACL-HLT'19*, 2019, pp. 4171–4186.

[1]https://github.com/pushshift/api

[2]https://developer.twitter.com/