Breaking the Cloak! Unveiling Chinese Cloaked Toxicity with Homophone Graph and Toxic Lexicon

Anonymous ACL submission

Abstract

Social media platforms have experienced a significant rise in toxic content, including abusive language and discriminatory remarks, presenting growing challenges for content moderation. Some users evade censorship by deliberately disguising toxic words through homophonic cloak, which necessitates the task of unveiling cloaked toxicity. Existing methods are mostly designed for English texts, while Chinese cloaked toxicity unveiling has not been solved yet. To tackle the issue, we propose C^2TU, a novel training-free and prompt-free method for Chinese cloaked toxic content unveiling. It first employs substring matching to identify candidate toxic words based on Chinese homo-graph and toxic lexicon. Then it filters those candidates that are non-toxic and corrects cloaks to be their corresponding toxicities. Specifically, we develop two model variants for filtering, which are based on BERT and LLMs, respectively. For LLMs, we address the auto-regressive limitation in computing word occurrence probability and utilize the full semantic contexts of a text sequence to reveal cloaked toxic words. Extensive experiments demonstrate that C^2TU can achieve superior performance on two Chinese toxic datasets. In particular, our method outperforms the best competitor by up to 71% on the F1 score and 35% on accuracy, respectively. Our code and data are available at `https://anonymous.4open.science/r/C2TU/`.

Disclaimer: *The paper contains content that may be profane, vulgar, or offensive.*

1 Introduction

With the exponential growth of user-generated contents, online social media platforms have become an inevitable communication tool with massive users (Omipidan and Sanusi, 2024; Briukhanova, 2021; Bala, 2014). Although the rapid information dissemination has clear benefits, a significant amount of toxic contents have emerged on social platforms over the past decade, such as abuse, discrimination, and cyberbullying (Sheth et al., 2022; Recuero, 2024; Kim et al., 2021; Hanscom et al., 2024). Social networks are thus facing increasingly severe challenges in governing toxicity.

Toxicity detection has recently attracted extensive attention (Anjum and Katarya, 2024; Lima et al., 2024; Schmidhuber and Kruschwitz, 2024), but most of the proposed methods (Phute et al., 2023; Zhang et al., 2024; Deng et al., 2022; Zhao et al., 2024) can only output binary predictions without clarifying the true toxicity the given content violates. Further, to evade censorship, some users on social platforms intentionally disguise toxic words by replacing parts or all of the words with homophonic characters or emojis (Xiao et al., 2024; Zhang et al., 2024; Kirk et al., 2022). For example, replace "操" (cào, means "f*ck") with "草" (cǎo, means "grass"), or "垃圾" (lā jī, means "rubbish") with "辣鸡" (là jī, means "spicy chicken") in Chinese. These deliberate cloaks significantly degrade the effectiveness of existing toxicity detection methods (Xiao et al., 2024) and also necessitate the task of *unveiling cloaked toxicity*, i.e., *correct cloaked toxic words into proto toxic words*. While there have been some methods (Gong et al., 2019; Dong et al., 2023) proposed to solve the task, most of them are specially designed for English texts, which cannot be directly used for Chinese due to the different characteristics of the two languages. Further, although a recent work (Xiao et al., 2024) points out the existence of Chinese cloak toxic content, they fail to offer a solution. Therefore, to bridge the gap, a research question naturally arises: *Can we develop a model to unveil Chinese cloaked toxic contents?*

We notice that Chinese Spelling Correction (CSC), which aims to correct misspelled Chinese characters, shares some similarities with our task. However, the two tasks are not entirely equiva-

lent. While CSC deals with unintentional user typos, misspellings in toxic content are often deliberate. Moreover, the corpora used in CSC tasks typically consist of more standardized language, which differs significantly in distribution from the toxicity found on the internet. Therefore, although models like SCOPE (Li et al., 2022) and Simple-CSC (Zhou et al., 2024) have demonstrated strong performance on CSC tasks, directly applying them to reveal Chinese cloaked toxicity may not yield satisfied results (see Section 4).

In this paper, we study the problem of Chinese Cloaked Toxicity Unveiling, and propose the C^2TU model (see Figure 1). Specifically, we utilize a homo-graph and a toxic lexicon to identify potential toxic words within the input text, thereby transforming the problem into a candidate toxic words filtering task. This task aligns well with BERT model's pretraining objective of masked language modeling. Hence, we first employ BERT model to compute the probability difference between tokens derived from raw and toxic words at specific positions in a sentence. If toxic tokens have larger probabilities, we then unveil the cloak. After that, we further explore using LLMs to address the filtering task. Similar to the BERT-based approach, we attempt to calculate word occurrence probabilities to decide whether a replacement should occur. Due to the auto-regressive nature of LLMs, the computation of word probabilities only conditions on the left-side context and thus lacks access to right-side semantics. Since LLMs are well-suited for computing sentence-level likelihoods, we mathematically reformulate the word-level probability difference into the sentence-level one based on Bayes' theorem to leverage the full semantic context of a sentence. Finally, we summarize the main contributions of our paper as:

- We propose a training-free and prompt-free method C^2TU for Chinese cloaked toxicity unveiling. Different from most existing methods for English, to our best knowledge, we are the first to solve the problem under Chinese.

- We leverage both BERT and LLMs to compute the occurrence probability of words for filtering candidate toxic words, respectively. In particular, we address the limitation of LLMs' auto-regressive nature, allowing them to compute the word probability differences based on the full semantic contexts.

- We conduct extensive experiments to evaluate the model performance on two Chinese toxic datasets. The results show that our methods are more competitive than other baselines w.r.t. both F1 score and accuracy metrics. In particular, our method outperforms the best competitor by up to 71% on the F1 score and 35% on accuracy.

2 Related Work

2.1 Cloaked Toxicity Unveiling

While extensive researches have been conducted on toxic content detection (Davidson et al., 2017; Jafari et al., 2024; Arora et al., 2023; Ghosh et al.; Phute et al., 2023; Zhang et al., 2024; Deng et al., 2022; Lu et al., 2023), most of them are not specially designed for cloaked toxic words (Xiao et al., 2024). Further, the reveal of deliberately cloaked toxic words emerges as a critical research challenge worthy of attention.

Some recent efforts have been made to address the issue by handling intentional obfuscation through character scrambling and misspellings in English. For example, some works (Gong et al., 2019) leverage contextual information to correct intentionally misspelled words, while others (Dong et al., 2023) focus on addressing word order disruptions by employing LLMs for robust comprehension of noisy text. However, these methods are designed for English. Unlike English—a word-based language where obfuscation typically alters intra-word characters—Chinese operates at the character level but cloaks content at the word level. This fundamental linguistic disparity prevents direct application of existing English-based methods to Chinese. However, research on unveiling Chinese cloaked toxic content remains largely unexplored.

2.2 Chinese Spelling Correction

Our task shares similarities with the CSC task, which aims to identify and correct Chinese spelling errors based on contextual, phonetic and graphemic information. Some existing methods are based on small language models (e.g., BERT) (Zhang et al., 2020; Cheng et al., 2020; Li et al., 2022). For example, SCOPE (Li et al., 2022) utilize phonetic knowledge of Chinese characters in conjunction with BERT for spelling correction. With the advent of LLMs, there are also LLM-based models (Dong et al., 2024; Zhou et al., 2024; Li et al., 2024). A representative model Simple-CSC (Zhou et al.,

2024) is a training-free and prompt-free method, treating LLMs purely as language models to perform spelling correction at the token level.

However, the CSC task is not well aligned with our task, as the substitution of toxic words is often intentional, and toxic texts themselves are abnormal texts that deviate from regular language usage. Therefore, we pay attention to Chinese cloaked toxicity unveiling in this paper.

3 Methodology

3.1 Graph and Lexicon Construction

3.1.1 Chinese Homo-graph

Given a Chinese toxic speech dataset, we first construct a homophone graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ to capture the likely cloak between tokens, where each node in $\mathcal{N}$ represents a Chinese character, and each edge in $\mathcal{E}$ indicates the phonetically similar relation. Specifically, we extract all Chinese characters from the dataset as nodes, then use the open-source library pypinyin[1] to obtain their pinyin regardless of tones. Then characters with identical pinyin are connected. Each node also has a self-loop, as each character is considered to have a homophonic relation with itself. We further consider polyphonic characters and phonetically similar pronunciations based on regional dialects.

Polyphonic Characters: Some Chinese characters have multiple pronunciations. If a character has at least one pronunciation matching another character's, an edge is linked between them.

Dialectal Relationships: There also exist dialectal phonetic confusions in Chinese, where users may replace characters based on their dialect pronunciations. Common confusions include:

• Retroflex and non-retroflex sounds(Dong et al., 2006), such as "山" (shān, means "mountain") and "三" (sān, means "three").

• Front and back nasal sounds(Chen and Guion-Anderson, 2011), such as "应" (yīng, means "should") and "因" (yīn, means "reason").

• Initial consonants "n" and "l"(Zhou and Wu, 2019; Koffi, 2019), such as "男" (nán, means "male") and "蓝" (lán, means "blue").

Considering these confusions, we introduce five types of additional relations: *"n" ↔ "l", "zh" ↔ "z", "ch" ↔ "c", "sh" ↔ "s" and "*ng" ↔ "*n"*. For example, "男" (nán) and "蓝" (lán) share an *"n" ↔ "l"* confusion, so an edge is added between node "男" and node "蓝", although their pinyin representations are not entirely identical.

3.1.2 Toxic Lexicon

We also utilize a toxic lexicon to introduce external knowledge about toxicity. The lexicon is a set of toxic words denoted as $\mathcal{L} = \{l_1, \cdots, l_m\}$. While we notice that some public datasets (e.g., `ToxiCN` (Lu et al., 2023)) have their own toxic lexicon released, they also contain many toxic words with homophone cloak. As these cloaks are frequently and commonly used on social platforms, they gradually become substitutes of their corresponding toxic contents. We take "垃圾" (lā jī, means "rubbish") and "辣鸡" (là jī, means "spicy chicken") as an example. Here, the proto toxic word is "垃圾", while "辣鸡" is a cloaked substitution commonly used in social media, even more frequently than the protoword. These clocks should also be removed instead of remaining in the vocabulary as toxic knowledge. Therefore, to keep a simple and clean toxic lexicon, we only retain protowords.

3.2 Toxic Word Matching

Given a text sequence X, we enumerate substring w in X and match each w against toxic words $l \in \mathcal{L}$ using the homo-graph $\mathcal{G}$. Formally, a substring w of length N is considered as a candidate toxic word if and only if $\exists l \in \mathcal{L}$, $\text{len}(w) = \text{len}(l)$ and $\forall k \in \{1, 2, \cdots, N\}$, $\mathcal{G}.\text{HasEdge}(w_k, l_k) = 1$, where w_k is k-th Chinese character in w, and so does l_k. Here, $\text{len}(\cdot)$ is the length of the input string. The function $\mathcal{G}.\text{HasEdge}(\cdot, \cdot) = 1$ indicates that two nodes are linked in $\mathcal{G}$.

The above matching method could have a small *false negative rate*, i.e., any homophonic substitution will be unveiled given a comprehensive homo-graph $\mathcal{G}$ and lexicon $\mathcal{L}$. However, it may also yield a high *false positive rate* by incorrectly matching non-toxic word as toxic word. For example, "干净" (gān jìng, means "clean") will be matched to "杠精" (gàng jīng, means "troll"), even though "干净" is typically a non-toxic word. False positives are especially severe for single-character toxic words, as all homophones of the single toxic character (e.g., "操" (cào, means "f*ck")) would be matched. The pseudocode of the matching algorithm is summarized in Alg. 1, which outputs M candidate toxic words pairs, denoted as $W_p = \{(w^{(i)}, l^{(i)})\}$ for $i = 1, \cdots, M$. Here, $l^{(i)}$ is a probable toxic unveiling of $w^{(i)}$.

[1] https://pypi.org/project/pypinyin/

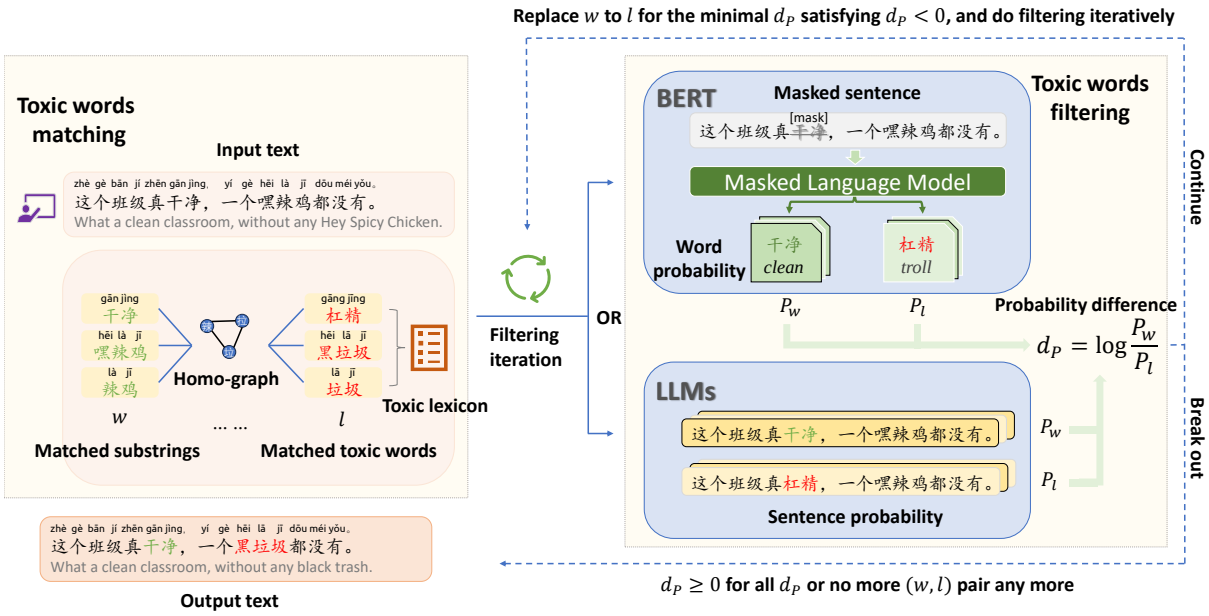

Figure 1: The main workflow of C^{2}TU method. C^{2}TU consists of two key stages: matching and filtering. In the matching stage, we identify candidate toxic terms in the input text by leveraging a homo-graph and a toxic lexicon. In the filtering stage, we use a language model (BERT model or LLM) to compute the probability gap for each (w, l) pair and iteratively replace the pair with the most significant gap. Once the iteration terminates, the unveiled sentence is returned.

3.3 Filtering Candidate Toxic Words

To further filter out the incorrect matches (see Figure 2) in W_p, we leverage the full semantics of the text sequence X. Formally, for each (w, l) pair, we define X_{pre} and X_{tail} as the prefix sequence and suffix sequence of w in X, respectively. Given X_{pre} and X_{tail}, we next calculate the probabilities $P(w|X_{\text{pre}}, X_{\text{tail}})$ and $P(l|X_{\text{pre}}, X_{\text{tail}})$, shorted as P_w and P_l, respectively. For all pairs (w, l) that satisfy $P_w < P_l$, we select the one that yields the largest difference, denoted as $(\tilde{w}, \tilde{l})$. Formally, the probability difference is calculated by:

$$\text{ProbDiff}(P_w, P_l) = \log P_w - \log P_l = \log \frac{P_w}{P_l}. \tag{1}$$

After that, in the given text sequence X, we replace $\tilde{w}$ with $\tilde{l}$, remove $(\tilde{w}, \tilde{l})$ from W_p and then repeat the previous process until W_p is empty or all the remaining pairs in W_p have $P_w \geq P_l$. Note that in each iteration, we need to recalculate the probabilities for each pair in W_p because once $\tilde{w}$ is replaced, X_{pre} and X_{tail} for remaining pairs change.

3.4 Computing Probability Difference

In this section, we introduce two methods for computing $P(x|X_{\text{pre}}, X_{\text{tail}})$, where x can be either w or l, with both BERT and LLMs.

3.4.1 BERT-based Method

During the pre-training phase, BERT utilizes an unsupervised Masked Language Modeling (MLM) task, whose goal is to predict tokens that are randomly masked. Hence, BERT is well-suited for computing $P(x|X_{\text{pre}}, X_{\text{tail}})$.

Given a text sequence $X = [X_{\text{pre}}, x, X_{\text{tail}}]$, for the i-th token x_i in x, we first mask it with the [mask] token and derive a text sequence $X_i^{\text{mask}} = [X_{\text{pre}}, \cdots, x_{i-1}, \texttt{[mask]}, x_{i+1}, \cdots, X_{\text{tail}}]$. Then we can output the probability for the masked token $P_{\text{BERT}}(x_i|X_i^{\text{mask}})$. For a toxic word x that includes N tokens, we sequentially predict each of these tokens and combine them by:

$$P_{\text{BERT}}(x|X_{\text{pre}}, X_{\text{tail}}) = \prod_{i=1}^{N} P_{\text{BERT}}(x_i|X_i^{\text{mask}}). \tag{2}$$

Due to the varying word lengths, we adjust Equation 2 to calculate the geometric mean of all N probabilities. Finally, the log probability to predict x is formally given by:

$$\begin{aligned} \log P_x &= \log \sqrt[N]{P_{\text{BERT}}(x|X_{\text{pre}}, X_{\text{tail}})} \\ &= \frac{1}{N} \sum_{i=1}^{N} \log P_{\text{BERT}}(x_i|X_i^{\text{mask}}) \end{aligned} \tag{3}$$

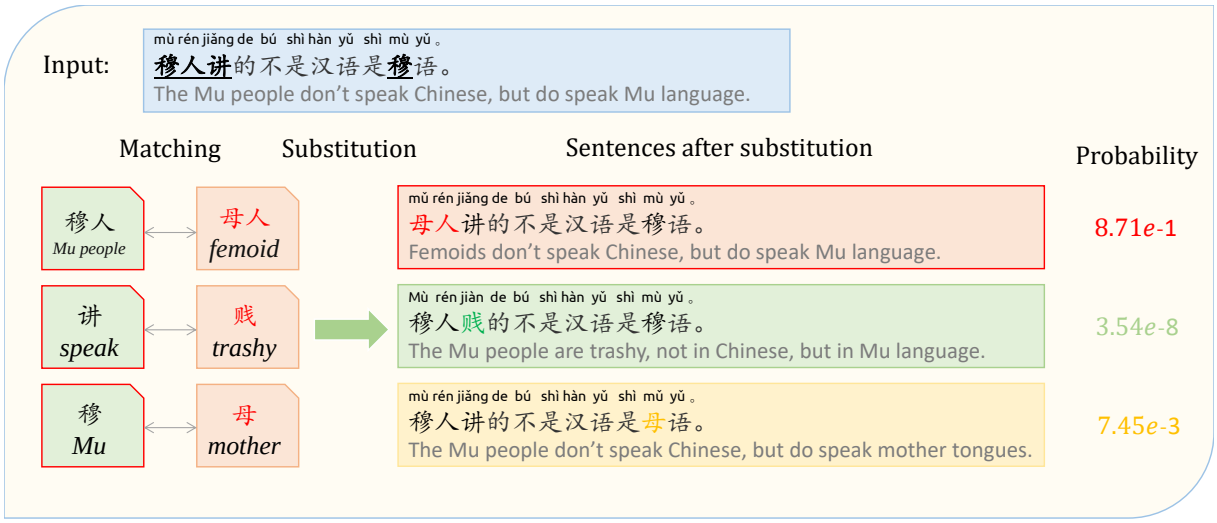

Figure 2: Three candidate toxic words pairs are matched in the input text. Each pair is individually substituted to generate three new sentences. It is evident that the pair ["讲", "贱"] is incorrect, as the resulting sentence is highly unnatural in meaning, leading to a significantly smaller probability. In contrast, the substitutions of ["穆人", "母人"] and ["穆", "母"] are correct, and their corresponding sentences yield higher probabilities.

3.4.2 LLMs-based Method

We next explore to use LLMs to filter candidate toxic words. Due to the auto-regressive nature, mainstream LLMs compute the occurrence probability of x based on the pre-context X_{pre} by:

$$P_{\text{LLM}}(x|X_{\text{pre}}) = \prod_{i=1}^{N} P_{\text{LLM}}(x_i|X_{\text{pre}}, x_1, \cdots, x_{i-1}). \tag{4}$$

In this way, only pre-context is used while the post-context is totally ignored. To address the issue, instead of directly computing the probability for a word, we calculate the probability difference, and transform the word probability difference into the sentence probability difference based on Bayes' theorem (Bayes, 1958). Formally, we prove:

Theorem 1 *Given a text sequence* $X = [X_{pre}, w, X_{tail}]$ *and a toxic word* l *with* $\text{len}(w) = \text{len}(l) = N$, *let* X' *denote the new sequence with* w *replaced by* l. *Then, the following equation holds:*

$$\text{ProbDiff}(P_w, P_l) = \log P_{\text{LLM}}(X) - \log P_{\text{LLM}}(X'), \tag{5}$$

where $P_{\text{LLM}}(X)$ *and* $P_{\text{LLM}}(X')$ *denote the probability of* X *and* X' *output by LLM, respectively. Both the probabilities can be calculated in the auto-regressive manner as Equation 4.*

In this way, we can capture the semantics of the entire sentence, rather than only the pre-context information. The proof of Theorem 1 is provided in Appendix A.

3.5 Time Complexity Analysis

C^{2}TU consists of two main stages: toxic word matching and filtering. Suppose we have an input text sequence X of length $|X|$ and a toxic lexicon with m toxic words.

In the matching stage, we enumerate substrings in X and match them with toxic words in the lexicon, whose largest length is N_{max}. We enumerate substrings of length $\leq N_{max}$ for matching. Considering that $N_{max} \ll |X|$, the number of enumerated substrings is linear to $|X|$. For each substring, we only need to match it with the toxic word of the same length, whose average number is assumed to be $\frac{m}{N_{max}}$. Therefore, the overall matching time complexity is $O(\frac{m}{N_{max}}|X|)$.

In the filtering stage, we iteratively handle M candidate (w, l) pairs. In each iteration, for each pair (w, l), we calculate the occurrence probability difference of the two words, leading to a time complexity of $O(M|X|^2)$ no matter whether BERT or LLMs are used. The overall time complexity is thus $O(M^2|X|^2)$. Different from other LLM-based models that generate tokens with a time complexity of $O(|X|^3)$, our method C^{2}TU-LLM only uses LLMs to output the probability difference, which is highly more efficient.

4 Experiment

4.1 Experimental Details

Datasets We conduct experiments on two public and well-known datasets `ToxicloakCN` (Xiao

et al., 2024) and COLDataset (Deng et al., 2022). Further, for fairness, we fine-tune the BERT-based CSC model on a subset of CHSD (Rao et al., 2023) named as CHSD-subset. Details on these datasets are given in Appendix C.

Baselines To our best knowledge, we are the first to unveil Chinese cloaked toxicity. Considering the relevance between our task and CSC task, we thus compare our method with three categories of baselines. (1) *Naive*: Directly replace all candidate toxic words without any filtering. (2) *BERT-based method*: SCOPE (Li et al., 2022) is a BERT-based CSC model that introduces an auxiliary task of Chinese pronunciation prediction to improve CSC task. We also finetune the pre-trained SCOPE model on CHSD-subset, obtaining the ft-SCOPE model. (3) *LLM-based method*: i) The prompt-based LLMs Baichuan2-7B-Base (Yang et al., 2023), Deepseek-V3 (Liu et al., 2024) and GPT-4o (Hurst et al., 2024). ii) Simple-CSC (Zhou et al., 2024) is a training-free and prompt-free LLM-based CSC model that considers character pronunciation and shape similarities.

Setup We construct homo-graph by pypinyin, and utilize lexicons after manual correction and deduplication for toxicity matching. Our filtering model consists of two versions: BERT-based and LLM-based. For the former, we utilize bert-chinese-base (Devlin et al., 2019). For the latter, we use Baichuan2-7B-Base (Yang et al., 2023) as our backbone model. None of the models' parameters have been modified, nor have any prompts been used, as our method is entirely training-free and prompt-free. For SCOPE, we follow its original pre-training stage and finetune it on CHSD-subset. For Simple-CSC, we employ it on Baichuan2-13B-Base as its original paper. All models can be downloaded on Huggingface[2]. We also notice that there exist some other CSC baselines. However, some of them (Zhang et al., 2020; Dong et al., 2024) underperform the two used CSC baselines, while others (Cheng et al., 2020; Li et al., 2024) lack complete pretraining datasets and source codes. Thus, we exclude them from comparison. The prompt template used for prompt-based LLMs is presented in Appendix E. Our experiments are conducted on a server equipped with an Nvidia A800 80G GPU.

[2] https://huggingface.co/

4.2 Main Results

We evaluate the model performance by F1 score and accuracy from two aspects (sentence-level and character-level), with each one containing both detection and correction tasks. The sentence-level task requires the complete detection/correction of all Chinese cloaked characters in a sentence and keeps other non-cloaked characters not mis-detected/mis-corrected; otherwise, the task fails. For the character-level task, detection and correction are applied exclusively to individual characters. The effectiveness of the model is reflected by its ability to identify and correct more cloaked characters while minimizing modifications to non-toxic ones—the higher the true positive rate and the lower the false positive rate, the better the performance. Details on the evaluation metrics can be found in Appendix B. From Table 1, we observe:

(1) The Naive model generally performs poorly across the datasets, which shows the importance of candidate toxic words filtering.

(2) ft-SCOPE consistently outperforms SCOPE across all the metrics and datasets. This shows the significant domain gap between general CSC pre-training data and toxic speech corpora. Further, our method C^{2}TU-BERT, which is not fine-tuned, beats ft-SCOPE across all the comparisons. Although fine-tuning helps memorize toxic knowledge, it only memorizes fixed toxicity mapping patterns in the training data. However, our model injected with homo-graph and toxic lexicon has stronger generalizability, which can perceive more cloaked toxicities.

(3) The LLM-based model Simple-CSC exhibits undesired results in some cases, which are even worse than BERT-based models. For example, in the sentence-level toxicity detection on the COLDataset, Simple-CSC achieves a accuracy of 53.85% while that of ft-SCOPE is 63.52%. This can be attributed to the limited cloaked toxic knowledge of LLMs, which is in accordance with the poor performance of prompt-based LLM models Baichuan2-7B-Base, Deepseek-V3 and GPT-4o.

(4) C^{2}TU-LLM and C^{2}TU-BERT take the first two ranks in all contests. On the one hand, they use homo-graph and toxic lexicon to match all the potential toxic words in a given text sequence, which increases true positive rate. On the other hand, they further utilize a filtering step to mitigate the risk that transforms non-toxic words into toxic, which decreases false positive rate.

Table 1: The main results of all baseline models and our method w.r.t. the F1 score/% and the accuracy/% metrics, where Det. and Cor. mean detection and correction. We highlight the **best** score in bold and underline the runner-up one.

Metric/%	Method	ToxicloakCN				COLDataset			
		Sentence		Character		Sentence		Character	
		Det.	Cor.	Det.	Cor.	Det.	Cor.	Det.	Cor.
F1 score	Naive	3.96	2.77	20.29	17.31	72.32	64.00	90.42	87.41
	SCOPE	13.23	6.33	7.81	2.22	14.80	9.91	18.41	7.01
	ft-SCOPE	35.79	26.99	17.48	11.73	62.66	59.15	41.21	35.83
	Simple-CSC	47.81	41.28	50.18	43.77	59.60	57.36	71.60	69.28
	Baichuan2-7B-Base	2.62	2.18	3.46	0.77	1.26	1.03	5.31	0.90
	Deepseek-V3	28.66	20.21	17.15	8.56	40.89	31.99	40.15	26.84
	GPT-4o	27.69	15.98	22.00	9.66	40.02	29.49	45.86	29.92
	C^2TU-BERT	62.65	58.53	69.77	65.81	87.06	82.56	91.00	90.00
	C^2TU-LLM	**74.67**	**70.01**	**78.56**	**75.04**	**90.52**	**88.54**	**93.73**	**93.30**
Accuracy	Naive	7.28	6.41	81.08	80.69	71.87	64.00	98.73	98.38
	SCOPE	35.35	31.33	79.42	78.79	15.47	11.22	78.35	76.78
	ft-SCOPE	52.88	47.36	80.72	80.01	63.52	60.14	82.32	81.33
	Simple-CSC	66.38	63.58	98.13	97.97	53.85	51.99	97.15	96.97
	Baichuan2-7B-Base	40.03	39.86	92.55	92.44	4.55	4.43	90.99	90.78
	Deepseek-V3	46.36	41.04	83.46	82.57	40.59	32.26	85.67	83.82
	GPT-4o	33.43	25.01	87.58	86.55	39.91	29.74	88.67	86.71
	C^2TU-BERT	76.32	74.38	98.66	98.53	84.65	80.51	98.94	98.84
	C^2TU-LLM	**83.60**	**81.36**	**99.02**	**98.89**	**88.37**	**86.55**	**99.25**	**99.20**

Table 2: Character-level analysis between the Naive method with C^2TU on `ToxicloakCN` and `COLDataset`. T/F means True/False. P/N means Positive/Negative. All results are normalized by dividing the raw values by the total number of sentences and characters in the dataset. We highlight the **best** score in bold.

Dataset	ToxicloakCN											
	Detection						Correction					
Metrics/%	TP↑	FP↓	FN↓	TN↑	recall↑	precision↑	TP↑	FP↓	FN↓	TN↑	recall↑	precision↑
Naive	**2.41**	18.74	**0.19**	78.67	**92.74**	11.39	**2.02**	18.74	**0.58**	78.67	**77.83**	9.74
C^2TU-BERT	1.55	0.29	1.05	97.11	59.54	84.25	1.42	0.29	1.18	97.11	54.50	83.04
C^2TU-LLM	1.79	**0.18**	0.80	**97.23**	69.09	**91.04**	1.67	**0.18**	0.93	**97.23**	64.13	**90.41**
Dataset	COLDataset											
	Detection						Correction					
Metrics/%	TP↑	FP↓	FN↓	TN↑	recall↑	precision↑	TP↑	FP↓	FN↓	TN↑	recall↑	precision↑
Naive	**5.99**	0.94	**0.33**	92.74	**94.72**	86.50	**5.64**	0.94	**0.69**	92.74	**89.11**	85.77
C^2TU-BERT	5.35	0.08	0.98	93.59	86.83	99.51	5.24	0.08	1.09	93.59	85.51	99.50
C^2TU-LLM	5.6	**0.02**	0.73	**93.66**	88.42	**99.71**	5.55	**0.02**	0.78	**93.66**	87.66	**99.70**

4.3 Analysis of Filtering Stage

As mentioned in Section 3.2, the Naive method that includes toxic word matching only and discards candidate toxic word filtering could lead to small false negative rate at the cost of high false positive rate. To verify the claim, we next show the results of *True/False Positive/Negative rate*, *recall* and *precision* in Table 2. From the table, Naive has better TP and FN rates than our methods, which results in higher recall values. This is because Naive regards all the candidate toxic substrings as the "true" toxic words. However, this leads to over-detection and over-correction, which categorizes a large number of non-toxic words to be toxic. Such indiscriminate filtering severely compromises the semantic integrity of benign sentences, making the Naive approach impractical for real-world scenarios where preserving the original meaning is crucial. Our methods further filter out candidate toxic words that are non-toxic and thus achieve better FP and TN rates, and higher precision scores. For example, On the `ToxicloakCN` dataset, in the correction task, Naive has a precision of 9.74%, while that of our model C^2TU-LLM is 90.41%, which is around

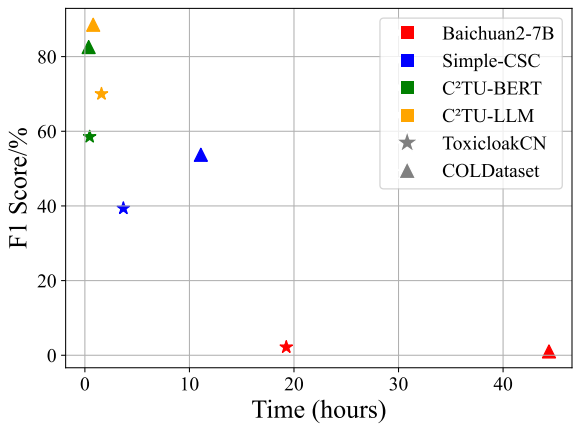

Figure 3: Efficiency study on the sentence-level correction task.

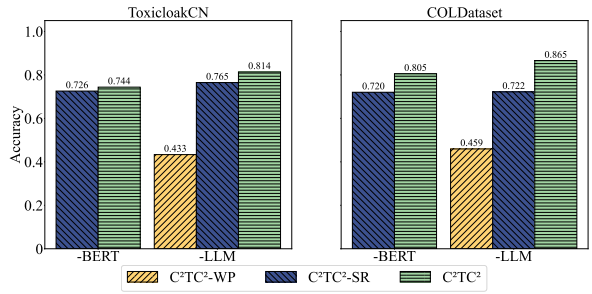

Figure 4: Sentence level correction accuracy gap between C^{2}TU , C^{2}TU-WP and C^{2}TU-SR on both datasets. The BERT-based model is employed on bert-chinese-base and the LLM-based model is employed on Baichuan2-7B-Base.

10× larger. In summary, our methods significantly improve precision, at the cost of a mild drop in recall, which finally lead to larger F1 scores.

4.4 Efficiency study

We next analyze the model efficiency. The results are presented in Figure 3, where ▲ indicates `COLDataset` and ★ represents `ToxicloakCN`. We use different colors to denote different methods including C^{2}TU-LLM, C^{2}TU-BERT, Simple-CSC and Baichuan2-7B-Base. For fairness, all LLM-based methods are based on Baichuan2-7B-Base. From the figure, we observe: (1) As LLM-based methods, C^{2}TU-LLM is significantly more efficient than Simple-CSC and Baichuan2-7B-Base. This is because C^{2}TU-LLM only uses LLM to output sentence probability difference, while the other two methods perform token generation. (2) C^{2}TU-BERT is more efficient than C^{2}TU-LLM due to the less number of model parameters, but it is at the cost of lower model effectiveness. In summary, our models are highly effective and also efficient.

4.5 Ablation Study

We also conduct an ablation study on C^{2}TU to understand the characteristics of its main components. Given the auto-regressive nature of LLMs, it is direct to predict the occurrence probability of a word based only on its preceding context X_{pre}. This helps us understand the importance of the sentence probability method that leverages full context in the filtering stage. We call this method **C^{2}TU-WP** (**W**ord **P**robability). Further, filtering candidate toxic words in Algorithm 2 involves multiple rounds. We introduce a single round strategy, in which each (w, l) pair satisfying $P_w < P_l$ is replaced. This helps us understand the importance of the iterative strategy for filtering. We call this method **C^{2}TU-SR** (**S**ingle **R**ound).

We then compare C^{2}TU with C^{2}TU-WP and C^{2}TU-SR. Due to the space limitation, we only show the results on the accuracy of sentence-level correction in Figure 4. For other cases, we observe similar results, detailed in Appendix G. From the figure, we see that: (1) C^{2}TU-LLM significantly beats C^{2}TU-WP. This indicates that X_{tail} also provides essential semantic information and leveraging the full text context for filtering candidate toxic words is necessary. (2) For both C^{2}TU-BERT and C^{2}TU-LLM, they consistently outperform C^{2}TU-SR across the datasets. This is because C^{2}TU replaces the (w, l) pair with the largest probability difference in each round, which iteratively enhances the context of remaining (w, l) pairs and the correction accuracy.

5 Conclusion

In this paper, we presented C^{2}TU, a training-free and prompt-free method for unveiling Chinese cloaked toxic contents on social platforms. Specifically, our method consists of two stages: Chinese cloaked toxicity matching and over-unveiling candidates filtering. For the former, we constructed the homo-graph to capture similar homophone relationships between Chinese characters. Subsequently, we matched toxic words via the homo-graph and toxic lexicon. To mitigate over-correction issues, we employed context-aware language models (BERT-based and LLM-based) to filter candidate toxic words iteratively. Comprehensive experimental results show that C^{2}TU achieves superior performance in Chinese cloaked toxicity unveiling task with high efficiency.

Limitations

While our method solves character-substitution cloak (the dominant case), other cloak types like character-splitting, cross-lingual homophones, and emoji replacements are not addressed in our paper, which will be studied in our future work.

Ethical Considerations

Our original intent was to unveil cloaked toxicity online in order to optimize cyber governance and purify the digital environment, given that cloaked toxicity is indeed a pervasive phenomenon.

We acknowledge that the toxic word candidates identified in Stage 1 - derived from homophonic graph and a toxic lexicon - while comprehensive, inevitably include numerous false positives. This occurs because simple homophonic matching with toxicity inherently produces substantial misclassifications, which will result in a significant number of normal expressions being misclassified as toxicity.

To address this, we implement our Stage 2 filtering method to systematically eliminate normal expressions that were mistakenly flagged. Our experimental results in Table 2 demonstrate that this filtering approach significantly reduces the false positive rate, albeit at some cost to true positive rate. This design choice prioritizes protecting users' rights to express legitimate content while maintaining detection efficacy.

From an academic research perspective, this represents the fullest extent of what we can achieve. Our technology is designed solely for academic research purposes, with no intended application in commercial or governmental contexts.

In the future, if this research ever be implemented in real-world content moderation systems, we maintain that human review remains indispensable. Our methodology can identify the vast majority of cloaked toxicity cases, which would then undergo human verification. Compared to comprehensive manual screening, this approach would significantly reduce the workload for human reviewers while ensuring that every content flagged as toxic receives human validation.

To address the potential misuse of this technology for censorship, we require all users to obtain our authorization. We promise that we will not grant permission to government agencies or companies that are closely affiliated with governments.

References

AI@Meta. 2024. Llama 3 model card.

Anjum and Rahul Katarya. 2024. Hate speech, toxicity detection in online social media: a recent survey of state of the art and opportunities. *International Journal of Information Security*, 23(1):577–608.

Arnav Arora, Preslav Nakov, Momchil Hardalov, Sheikh Muhammad Sarwar, Vibha Nayak, Yoan Dinkov, Dimitrina Zlatkova, Kyle Dent, Ameya Bhatawdekar, Guillaume Bouchard, and 1 others. 2023. Detecting harmful content on online platforms: what platforms need vs. where research efforts go. *ACM Computing Surveys*, 56(3):1–17.

Kiran Bala. 2014. Social media and changing communication patterns. *Global Media Journal: Indian Edition*, 5(1).

Thomas Bayes. 1958. An essay towards solving a problem in the doctrine of chances. *Biometrika*, 45(3-4):296–315.

Galina Briukhanova. 2021. Social media as a new communication platform in the context of the information eco strategy.

Ying Chen and Susan Guion-Anderson. 2011. Perceptual confusabiltiy of word-final nasals in southern min and mandarin: Implications for coda nasal mergers in chinese. In *ICPhS*, pages 464–467.

Xingyi Cheng, Weidi Xu, Kunlong Chen, Shaohua Jiang, Feng Wang, Taifeng Wang, Wei Chu, and Yuan Qi. 2020. Spellgcn: Incorporating phonological and visual similarities into language models for chinese spelling check. *arXiv preprint arXiv:2004.14166*.

Chengyu Cui. 2020. Jionlp.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Proceedings of the international AAAI conference on web and social media*, volume 11, pages 512–515.

Jiawen Deng, Jingyan Zhou, Hao Sun, Chujie Zheng, Fei Mi, Helen Meng, and Minlie Huang. 2022. Cold: A benchmark for chinese offensive language detection. *arXiv preprint arXiv:2201.06025*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. 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)*, pages 4171–4186.

Bin Dong, QW Zhao, and YH Yan. 2006. Automatic scoring of flat tongue and raised tongue in computer-assisted mandarin learning. In *Proc. Int. Symp. on Chinese Spoken Language Processing*, pages 580–591.

Guanting Dong, Jinxu Zhao, Tingfeng Hui, Daichi Guo, Wenlong Wang, Boqi Feng, Yueyan Qiu, Zhuoma Gongque, Keqing He, Zechen Wang, and 1 others. 2023. Revisit input perturbation problems for llms: A unified robustness evaluation framework for noisy slot filling task. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 682–694. Springer.

Ming Dong, Yujing Chen, Miao Zhang, Hao Sun, and Tingting He. 2024. Rich semantic knowledge enhanced large language models for few-shot chinese spell checking. *arXiv preprint arXiv:2403.08492*.

Shaona Ghosh, Prasoon Varshney, Erick Galinkin, and Christopher Parisien. Aegis: Online adaptive ai content safety moderation with ensemble of llm experts, 2024. *URL https://arxiv. org/abs/2404.05993*.

Hongyu Gong, Yuchen Li, Suma Bhat, and Pramod Viswanath. 2019. Context-sensitive malicious spelling error correction. In *The World Wide Web Conference*, pages 2771–2777.

Rhett Hanscom, Tamara Silbergleit Lehman, Qin Lv, and Shivakant Mishra. 2024. The toxicity phenomenon across social media. *arXiv preprint arXiv:2410.21589*.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Nazanin Jafari, James Allan, and Sheikh Muhammad Sarwar. 2024. Target span detection for implicit harmful content. In *Proceedings of the 2024 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 117–122.

Aiqi Jiang, Xiaohan Yang, Yang Liu, and Arkaitz Zubiaga. 2022. Swsr: A chinese dataset and lexicon for online sexism detection. *Online Social Networks and Media*, 27:100182.

Jin Woo Kim, Andrew Guess, Brendan Nyhan, and Jason Reifler. 2021. The distorting prism of social media: How self-selection and exposure to incivility fuel online comment toxicity. *Journal of Communication*, 71(6):922–946.

Hannah Kirk, Bertie Vidgen, Paul Rottger, Tristan Thrush, and Scott A. Hale. 2022. Hatemoji: A test suite and adversarially-generated dataset for benchmarking and detecting emoji-based hate. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1352–1368, Seattle, United States. Association for Computational Linguistics.

Ettien Koffi. 2019. An acoustic phonetic account of the confusion between [l] and [n] by some chinese speakers. *Linguistic Portfolios*, 8(1):6.

Jiahao Li, Quan Wang, Zhendong Mao, Junbo Guo, Yanyan Yang, and Yongdong Zhang. 2022. Improving chinese spelling check by character pronunciation prediction: The effects of adaptivity and granularity. *arXiv preprint arXiv:2210.10996*.

Kunting Li, Yong Hu, Liang He, Fandong Meng, and Jie Zhou. 2024. C-llm: Learn to check chinese spelling errors character by character. *arXiv preprint arXiv:2406.16536*.

Luiz Henrique Quevedo Lima, Adriana Silvina Pagano, and Ana Paula Couto da Silva. 2024. Toxic content detection in online social networks: a new dataset from brazilian reddit communities. In *Proceedings of the 16th International Conference on Computational Processing of Portuguese-Vol. 1*, pages 472–482.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Junyu Lu, Bo Xu, Xiaokun Zhang, Changrong Min, Liang Yang, and Hongfei Lin. 2023. Facilitating fine-grained detection of chinese toxic language: Hierarchical taxonomy, resources, and benchmarks. *arXiv preprint arXiv:2305.04446*.

Ismail Adewale Omipidan and Bernice Oluwalanumi Sanusi. 2024. Rise of social media in the digital age: Whatsapp a threat to effective communication. *IMSU Journal of Communication Studies*, 8(1):142–153.

Mansi Phute, Alec Helbling, Matthew Hull, ShengYun Peng, Sebastian Szyller, Cory Cornelius, and Duen Horng Chau. 2023. Llm self defense: By self examination, llms know they are being tricked. *arXiv preprint arXiv:2308.07308*.

Xiaojun Rao, Yangsen Zhang, Shuang Peng, Qilong Jia, and Xueyang Liu. 2023. Chinese hate speech detection method based on RoBERTa-WWM. In *Proceedings of the 22nd Chinese National Conference on Computational Linguistics*, pages 501–511, Harbin, China. Chinese Information Processing Society of China.

Raquel Recuero. 2024. The platformization of violence: Toward a concept of discursive toxicity on social media. *Social Media+ Society*, 10(1):20563051231224264.

Maximilian Schmidhuber and Udo Kruschwitz. 2024. Llm-based synthetic datasets: Applications and limitations in toxicity detection. *LREC-COLING 2024*, 37.

Amit Sheth, Valerie L Shalin, and Ugur Kursuncu. 2022. Defining and detecting toxicity on social media: context and knowledge are key. *Neurocomputing*, 490:312–318.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Yunze Xiao, Yujia Hu, Kenny Tsu Wei Choo, and Roy Ka-wei Lee. 2024. Toxicloakcn: Evaluating robustness of offensive language detection in chinese with cloaking perturbations. *The 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP 2024)*.

Aiyuan Yang, Bin Xiao, Bingning Wang, Borong Zhang, Ce Bian, Chao Yin, Chenxu Lv, Da Pan, Dian Wang, Dong Yan, and 1 others. 2023. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*.

Hengxiang Zhang, Hongfu Gao, Qiang Hu, Guanhua Chen, Lili Yang, Bingyi Jing, Hongxin Wei, Bing Wang, Haifeng Bai, and Lei Yang. 2024. Chinesesafe: A chinese benchmark for evaluating safety in large language models. *arXiv preprint arXiv:2410.18491*.

Shaohua Zhang, Haoran Huang, Jicong Liu, and Hang Li. 2020. Spelling error correction with soft-masked bert. *arXiv preprint arXiv:2005.07421*.

Yibo Zhao, Jiapeng Zhu, Can Xu, and Xiang Li. 2024. Enhancing llm-based hatred and toxicity detection with meta-toxic knowledge graph. *The 63rd Annual Meeting of the Association for Computational Linguistics*.

Houquan Zhou, Zhenghua Li, Bo Zhang, Chen Li, Shaopeng Lai, Ji Zhang, Fei Huang, and Min Zhang. 2024. A simple yet effective training-free prompt-free approach to Chinese spelling correction based on large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17446–17467, Miami, Florida, USA. Association for Computational Linguistics.

Jingyan Zhou, Jiawen Deng, Fei Mi, Yitong Li, Yasheng Wang, Minlie Huang, Xin Jiang, Qun Liu, and Helen Meng. 2022. Towards identifying social bias in dialog systems: Framework, dataset, and benchmark. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3576–3591, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yuyan Zhou and Ying Wu. 2019. An erp study for phonemic merger in chinese dialects. *Modern Linguistics*, 7(1).

A Theoretical Proof

Proof A.1 *Given a text sequence* $X = [X_{pre}, w, X_{tail}]$ *and a toxic word* l *with* $\mathrm{len}(w) = \mathrm{len}(l) = N$*, let* X' *denote the new sequence with* w *replaced by* l*. Then, the following equation holds:*

$$\mathrm{ProbDiff}(P_w, P_l) = \log P_{\mathrm{LLM}}(X) - \log P_{\mathrm{LLM}}(X') \tag{6}$$

where $P_{\mathrm{LLM}}(X)$ *and* $P_{\mathrm{LLM}}(X')$ *denote the probability of* X *and* X' *output by LLM, respectively. Both the probabilities can be calculated in the autoregressive manner as Equation 4.*

Firstly, we derive the general formula for $P_{\mathrm{LLM}}(w|X_{\mathrm{pre}}, X_{\mathrm{tail}})$*:*

$$\begin{aligned} P_{\mathrm{LLM}}(w|X_{\mathrm{pre}}, X_{\mathrm{tail}}) &= \frac{P_{\mathrm{LLM}}(X_{\mathrm{pre}}, w, X_{\mathrm{tail}})}{P_{\mathrm{LLM}}(X_{\mathrm{pre}}, X_{\mathrm{tail}})} \\ &= \frac{P_{\mathrm{LLM}}(X)}{P_{\mathrm{LLM}}(X_{\mathrm{pre}}, X_{\mathrm{tail}})} \end{aligned} \tag{7}$$

Analogously:

$$P_{\mathrm{LLM}}(l|X_{\mathrm{pre}}, X_{\mathrm{tail}}) = \frac{P_{\mathrm{LLM}}(X')}{P_{\mathrm{LLM}}(X_{\mathrm{pre}}, X_{\mathrm{tail}})} \tag{8}$$

Therefore:

$$\begin{aligned} &\mathrm{ProbDiff}(P_w, P_l) \\ &= \log P_w - \log P_l \\ &= \log P_{\mathrm{LLM}}(w|X_{\mathrm{pre}}, X_{\mathrm{tail}}) \\ &\quad - \log P_{\mathrm{LLM}}(l|X_{\mathrm{pre}}, X_{\mathrm{tail}}) \\ &= \log \frac{P_{\mathrm{LLM}}(X)}{P_{\mathrm{LLM}}(X_{\mathrm{pre}}, X_{\mathrm{tail}})} - \log \frac{P_{\mathrm{LLM}}(X')}{P_{\mathrm{LLM}}(X_{\mathrm{pre}}, X_{\mathrm{tail}})} \\ &= \log(\frac{P_{\mathrm{LLM}}(X)}{P_{\mathrm{LLM}}(X_{\mathrm{pre}}, X_{\mathrm{tail}})} \cdot \frac{P_{\mathrm{LLM}}(X_{\mathrm{pre}}, X_{\mathrm{tail}})}{P_{\mathrm{LLM}}(X')}) \\ &= \log \frac{P_{\mathrm{LLM}}(X)}{P_{\mathrm{LLM}}(X')} \\ &= \log P_{\mathrm{LLM}}(X) - \log P_{\mathrm{LLM}}(X') \end{aligned} \tag{9}$$

Q.E.D.

B Evaluation Metrics

We evaluate the model performance by F1 score and accuracy from two aspects (sentence-level and character-level), with each one containing both detection and correction tasks, which is common metrics in the CSC field.

For sentence level, given the sample pair of (source sentence and target sentence) and the corresponding corrected sentence, if the length of corrected sentence is not equal to the target sentence, the result is FN for both detection and correction tasks. Otherwise, we have the following definition:

Sentence Level Detection (1) If the source sentence is different to the target sentence, the source sentence needs to be corrected. In this situation, if all wrong characters in source sentence is corrected (no matter correctly or not), and no proper character is over-corrected, the result is TP. Otherwise, the result is FN. (2) If the source sentence is identical to the target sentence, the source sentence doesn't need to be corrected. In this situation, if the corrected sentence is same to the target sentence, the result is TN. Otherwise, the result is FP.

Sentence Level Correction (1) If the source sentence is different to the target sentence, the source sentence needs to be corrected. In this situation, if all wrong characters in source sentence is corrected correctly, and no proper character is over-corrected, the result is TP. Otherwise, the result is FN. (2) If the source sentence is identical to the target sentence, the source sentence doesn't need to be corrected. In this situation, if the corrected sentence is same to the target sentence, the result is TN. Otherwise, the result is FP.

For character level, given the sample pair of (source character and target character) and the corresponding corrected character:

Character Level Detection (1) If the source character is different to the target character, the source character needs to be corrected. In this situation, if the corrected character is not same to the source character (no matter the correction is correct or not), the result is TP. Otherwise, the result is FN. (2) If the source character is identical to the target character, the source character doesn't need to be corrected. In this situation, if the corrected character is same to the target character, the result is TN. Otherwise, the result is FP.

Character Level Correction (1) If the source character is different to the target character, the source character needs to be corrected. In this situation, if the source character is corrected correctly, the result is TP. Otherwise, the result is FN. (2) If the source character is identical to the target character, the source character doesn't need to be corrected. In this situation, if the corrected character is same to the target character, the result is TN. Otherwise, the result is FP.

Based on these definitions, we can calculate the F1 score (F_1) and the accuracy (Acc) by:

$$F_1 = \frac{2pr}{p + r} \tag{10}$$

$$Acc = \frac{TP + TN}{TP + FN + FP + TN} \quad (11)$$

where $p = \frac{TP}{TP+FP}$ and $r = \frac{TP}{TP+FN}$.

C Datasets

We conduct experiments on two public datasets: ToxicloakCN (Xiao et al., 2024) and COLDataset (Deng et al., 2022), and finetune SCOPE (Li et al., 2022) model on CHSD-subset constructed from CHSD (Rao et al., 2023). Details on these datasets are given in Table 3.

Table 3: Dataset size and lexicon size of ToxicloakCN, COLDataset and CHSD-subset.

Dataset	Size	Lexicon size
ToxicloakCN	4,586	387
COLDataset	10,415	115
CHSD-subset	2,790	602

Homo-Graph We use pypinyin to capture tone-ignored pinyin of all Chinese characters and connect characters with identical or phonetically similar pinyins, which is defined in Section 3.1.1. The details of homo-graphs is shown in Table 4.

Table 4: Homo-graph size of ToxicloakCN and COLDataset.

Dataset	Node size	Edge size
ToxicloakCN	3,613	141,683
COLDataset	3,879	163,969

Toxic Lexicon (1) ToxicloakCN is an enhanced dataset derived from ToxiCN (Lu et al., 2023), where JioNLP (Cui, 2020) and NMSL[3] are applied to perform homophone substitution and semantically similar emoji substitution on toxic speech, respectively (Xiao et al., 2024). For keyword-based substitution, the toxic lexicon from ToxiCN is used as keywords. For cleaner lexicon without cloak, we further correct and deduplicate the ToxiCN's lexicon. We first manually correct all toxic words in the original lexicon back to their protowords, and then retain each protoword. (2) COLDataset is collected through web crawling based on keywords, comprising 37,480 contents and spanning a wide range of topics related to race, gender, and regional issues. Since no existing work has compiled a toxic lexicon for COLDataset, we instead utilize the crawler keywords used during dataset construction(Deng et al., 2022) as toxic lexicon. Based on the lexicon, we merge the validation set with the test set and apply JioNLP (Cui, 2020) to perform homophone substitution and finally get the cloaked dataset, following the noise injection procedure of ToxicloakCN (Xiao et al., 2024).

[3] https://github.com/THUzhangga/NMSL

Table 5: The distribution of the CHSD-subset. T.L means ToxiCN's lexicon, while C.L means COLDataset's lexicon.

	w/ T.L	w/o T.L
w/ C.L	1,706	542
w/o C.L	542	0

Dataset for Finetuning To ensure fair comparison between our model and CSC model, we further fine-tune BERT-based CSC model SCOPE (Li et al., 2022) on a subset of CHSD (Rao et al., 2023), which integrates datasets COLDataset (Deng et al., 2022), SWSR (Jiang et al., 2022) and CDIAL-BIASDATASET (Zhou et al., 2022) to balance the distribution difference of toxic contents (see Table 5). To construct the CHSD-subset, we first filter out samples that appear in the part of COLDataset used in the test set. Next, we select samples based on toxic keywords from the two lexicons of ToxicloakCN and COLDataset, ensuring that our fine-tuning data contains relevant toxic content. Then we merge the two lexicons and apply JioNLP (Cui, 2020) to introduce noise. Note that we use the original lexicon of ToxiCN here. Finally, we construct CHSD-subset with 2790 samples that consists of 80% training and 20% validation data. Then we finetune the pre-trained SCOPE model with following settings: learning rate=$5e-5$, batch size=32, accumulate grad batches=2, epoch=20, warmup proporation=0.1. We first train the SCOPE model and it achieves a character-level detection F1-score of 0.8007 and a character-level corection F1-score of 0.7810 on CSC dataset. We then conducted fine-tuning on CHSD-subset and it achieves a character-level detection F1-score of 0.6026 and a character-level correction F1-score of 0.5531. Details of CHSD-subset are presented in Table 5.

D Pseudocode of Matching and Filtering

D.1 Toxic Words Matching Algorithm

Algorithm 1 Toxic Words Matching

Input: User comment X, lexicon $\mathcal{L}$, homo-graph $\mathcal{G}$
Output: Potentially cloaked toxic words W_p

$W \leftarrow \text{SlidingWindow}(X), W_p \leftarrow \emptyset$
for each $w^{(i)}$ in W, each $l^{(j)}$ in $\mathcal{L}$ **do**
  if $\text{len}(w^{(i)}) = \text{len}(l^{(j)})$ **then**
    $N \leftarrow \text{len}(w^{(i)}), flag \leftarrow \text{True}$
    for each $k \in \{1, 2, \cdots, N\}$ **do**
      if $\mathcal{G}.\text{HasEdge}(w_k^{(i)}, l_k^{(j)}) \neq 1$ **then**
        $flag \leftarrow \text{False}$
      end if
    end for
    if $flag = \text{True}$ **then**
      $W_p \leftarrow W_p \cup \{(w^{(i)}, l^{(j)})\}$
    end if
  end if
end for
return W_p

D.2 Filtering Toxic Words Algorithm

Algorithm 2 Filtering Toxic Words

Input: $X, W_p, P(\cdot)$
Output: Corrected text X'

Initialize $X' \leftarrow X$
while W_p is not empty **do**
  for each tuple $(w^{(i)}, l^{(j)})$ in W_p **do**
    $X_{\text{pre}}, X_{\text{tail}} \leftarrow \text{GetPreTail}(X', w^{(i)})$
    $P_{w^{(i)}} \leftarrow P(w^{(i)} \mid X_{\text{pre}}, X_{\text{tail}})$
    $P_{l^{(i)}} \leftarrow P(l^{(i)} \mid X_{\text{pre}}, X_{\text{tail}})$
    $d_i \leftarrow \text{ProbDiff}(P_{w^{(i)}}, P_{l^{(i)}})$
  end for
  Get the index with the most significant difference: $k \leftarrow \underset{i}{\text{argmin}}\ d_i$
  if $P_{w^{(k)}} < P_{l^{(k)}}$ **then**
    $X_{\text{pre}}, X_{\text{tail}} \leftarrow \text{GetPreTail}(X', w^{(k)})$
    Replace: $X' \leftarrow [X_{\text{pre}}, l^{(k)}, X_{\text{tail}}]$
    $W_p \leftarrow W_p \setminus \{(w^{(k)}, l^{(k)})\}$
  else
    break
  end if
end while
return X'

E Prompt Template

Our prompt-based framework utilizes the following template structure (optimized version, generated by GPT-4o) to unveiling the cloaked toxicity:

Prompt template for prompt-based method.

- System:

任务：
针对中文敏感信息中带掩盖的错别字纠正。
Task:
Correcting obfuscated typos in Chinese toxic text.

要求：
1. 需要严格保证输入句子和输出句子长度一致；
2. 只需要返回纠正后的句子，不要输出任何其他内容。
Requirements:
1. Input-output length consistency is mandatory;
2. Return only the corrected sentence without any additional output.

- User: <Input sentence>

- Assistant: <Output sentence>

F Performance Across LLMs

We evaluate the performance of C^{2}TU-LLM across different LLMs, including LLaMA3-8B (AI@Meta, 2024), Qwen2.5-7B (Team, 2024) and Baichuan2-7B (Yang et al., 2023). We take Simple-CSC and Deepseek-V3 as our baselines, as they are also LLM-based. The results are shown in Table 6. While Simple-CSC employs Baichuan2-13B-Base and Deepseek-V3 is 671B, our method consistently outperforms them with different LLMs of smaller parameter scales (7B and 8B). This further shows the effectiveness of our method with homo-graph and Chinese toxic lexicon, which inject knowledge of cloaked toxicity into LLMs.

G Full Results of Ablation Study

Full results of ablation study between C^{2}TU, C^{2}TU-WP and C^{2}TU-SR on both datasets. The C^{2}TU-BERT is employed on bert-chinese-base and the C^{2}TU-LLM is employed on Baichuan2-7B-Base.

H Human Evaluation

Given that our evaluation metric for unveiling is based on rule matching rather than semantics, some sentences may not be corrected to their original form even though the unveiling are indeed valid.

Table 6: The results of C^{2}TU-LLM with different LLMs w.r.t. F1 score and accuracy metrics, where Det. and Cor. mean detection and correction. We highlight the **best** score in bold.

Metric/%	Method	Model	ToxicloakCN				COLDataset			
			Sentence		Character		Sentence		Character	
			Det.	Cor.	Det.	Cor.	Det.	Cor.	Det.	Cor.
F1 score	Simple-CSC	Baichuan2-13B	47.81	41.28	50.18	43.77	59.60	57.36	71.60	69.28
	Prompt-based	Deepseek-V3	28.66	20.21	17.15	8.56	40.89	31.99	40.15	26.84
	C^{2}TU-LLM	LLaMA3-8B	68.74	64.30	74.44	70.98	89.55	86.25	92.74	91.97
		Qwen2.5-7B	67.20	63.52	73.46	70.19	89.37	86.30	92.41	91.73
		Baichuan2-7B	**74.67**	**70.01**	**78.56**	**75.04**	**90.52**	**88.54**	**93.73**	**93.30**
Accuracy	Simple-CSC	Baichuan2-13B	66.38	63.58	98.13	97.97	53.85	51.99	97.15	96.97
	Prompt-based	Deepseek-V3	46.36	41.04	83.46	82.57	40.59	32.26	85.67	83.82
	C^{2}TU-LLM	LLaMA3-8B	79.83	77.71	98.85	98.73	87.02	83.98	99.14	99.06
		Qwen2.5-7B	78.81	77.04	98.80	98.69	86.94	84.12	99.10	99.03
		Baichuan2-7B	**83.60**	**81.36**	**99.02**	**98.89**	**88.37**	**86.55**	**99.25**	**99.20**

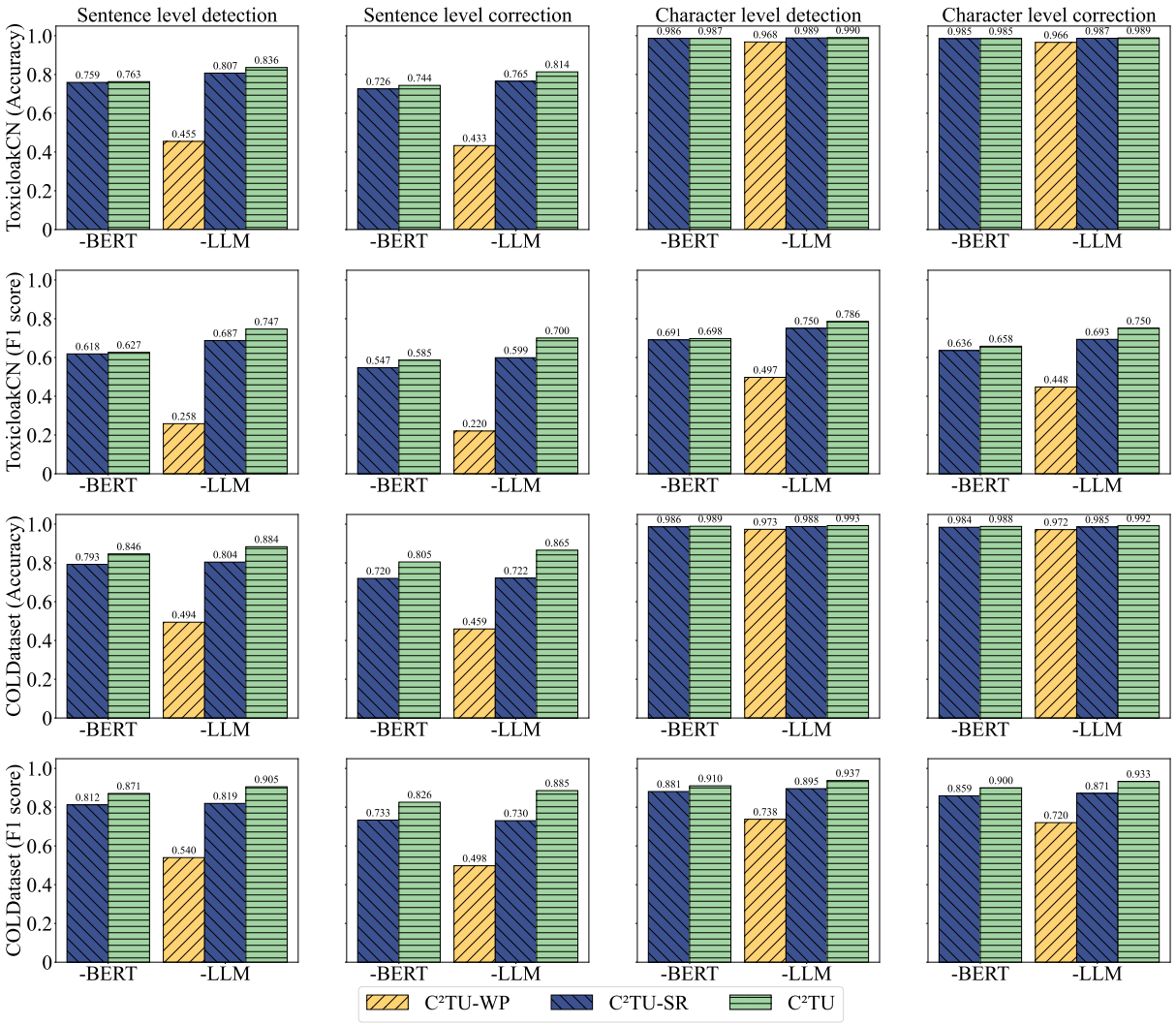

Figure 5: Full results of ablation study.

Therefore, we conducted a human evaluation based on sampling. Specifically, we randomly select 50 samples from the `ToxicloakCN` dataset and perform unveiling using three models: Simple-CSC,

C^2TU-BERT, and C^2TU-LLM, which represent the top3-performing models.

For evaluation, we recruit three evaluators (two female and one male) with prior knowledge of online toxic vocabulary. All of them are highly interested in cloaked toxicity topic and volunteer to undergo the experiment. They are recruited as Research Assistants (RAs) and remunerated in accordance with the wage standards prescribed by local laws.

All RAs receive detailed task instructions and are shown the toxic lexicon. During assessment, each evaluator is presented with five versions of each sentence: the raw sentence, cloaked sentence, and the three model-generated unveiled sentences. They are then asked to label each model's output: marking it as "1" if they believed the sentence contains no remaining cloaking, and "0" otherwise. The final evaluation result for each sentence is determined by majority voting among the three evaluators' labels.

Table 7: The results of human evaluation.

	Simple-CSC	C^2TU-BERT	C^2TU-LLM
Accuracy	60%	78%	84%

The results in Table 7 demonstrate that the C^2TUmodel significantly outperforms the baseline model, with the LLM-based version showing slightly better performance than the BERT-based version.