MALCOM: Generating Malicious Comments to Attack Neural Fake News Detection Models

Thai Le, Suhang Wang, and Dongwon Lee
The Pennsylvania State University, USA
{thai.le, szw494, dongwon}@psu.edu

***Abstract*—In recent years, the proliferation of so-called "fake news" has caused much disruptions in society and weakened the news ecosystem. Therefore, to mitigate such problems, researchers have developed state-of-the-art (SOTA) models to auto-detect fake news on social media using sophisticated data science and machine learning techniques. In this work, then, we ask "what if adversaries attempt to attack such detection models?" and investigate related issues by (i) proposing a novel attack scenario against fake news detectors, in which adversaries can post malicious comments toward news articles to mislead SOTA fake news detectors, and (ii) developing Malcom, an end-to-end adversarial comment generation framework to achieve such an attack. Through a comprehensive evaluation, we demonstrate that about 94% and 93.5% of the time on average Malcom can successfully mislead five of the latest neural detection models to always output targeted real and fake news labels. Furthermore, Malcom can also fool black box fake news detectors to always output real news labels 90% of the time on average. We also compare our attack model with four baselines across two real-world datasets, not only on attack performance but also on generated quality, coherency, transferability, and robustness. We release the source code of Malcom at https://github.com/lethaiq/MALCOM[1].**

***Index Terms*—Fake News Detection, Malicious Comments, Adversarial Examples**

I. INTRODUCTION

Circulation of fake news, i.e., false or misleading pieces of information, on social media is not only detrimental to individuals' knowledge but is also creating an erosion of trust in society. Fake news has been promoted with deliberate intention to widen political divides, to undermine citizens' confidence in public figures, and even to create confusion and doubts among communities [11]. Hence, any quantity of fake news is intolerable and should be carefully examined and combated [2]. Due to the high-stakes of fake news detection in practice, therefore, tremendous efforts have been taken to develop fake news detection models that can auto-detect fake news with high accuracies [1], [4], [25], [27].Figure 1 (on top) shows an example of a typical news article posted on the social media channels such as Twitter and Facebook. A fake news detection model then uses different features of the article (e.g., headline and news content) and outputs a prediction on whether such an article is real or fake. Further, recent research has shown that users' engagement (e.g., user comments or replies) on public news channels on which these articles are shared become a critical signal to flag questionable news [25]. Hence, some of the state-of-the-art (SOTA) fake news detection models [4], [24], [25], [27] have exploited these user engagement features into their prediction models with great successes.

[1]This work was in part supported by NSF awards #1742702, #1820609, #1909702, #1915801, #1934782, and #IIS1909702

Fig. 1: A malicious comment generated by Malcom misleads a neural fake news detector to predict real news as fake.

Real Comment: admitting i'm not going to read this (...)
Malcom: *he's a conservative from a few months ago*
Prediction Change: Real News ⟶ Fake News

Despite the good performances, the majority of SOTA detectors are deep learning based, and thus become vulnerable to the recent advancement in adversarial attacks [21]. As suggested by [38], for instance, a careful manipulation of the title or content of a news article can mislead the SOTA detectors to predict fake news as real news and vice versa. [12] also shows that hiding questionable content in an article or replacing the source of fake news to that of real news can also achieve the same effect. However, these existing attack methods suffer from three key limitations: (i) unless an attacker is also the publisher of fake news, she cannot exercise *post-publish* attacks, i.e., once an article is published, the attacker cannot change its title or content; (ii) an attacker generates adversarial texts either by marginally tampering certain words or characters using pre-defined templates (e.g., "hello" → "he11o", "fake" → "f@ke" [20]), appending short random phrases (e.g., "zoning tapping fiennes") to the original text [3], or flipping a vulnerable character or word (e.g. "opposition" → "oBposition") [5], all of which can be easily detected by a careful examination with naked eyes; and (iii) they largely focus on the vulnerabilities found in the title and content, leaving social responses, i.e., *comments* and *replies*, unexplored.

2374-8486/20/$31.00 ©2020 IEEE
DOI 10.1109/ICDM50108.2020.00037

Authorized licensed use limited to: SARDAR VALLABHBHAI NATIONAL INSTITUTE OF TECH. Downloaded on November 29,2023 at 08:58:58 UTC from IEEE Xplore. Restrictions apply.

Since many SOTA neural fake news detectors exploit users' comments to improve fake news detection, this makes them highly vulnerable from attacks via adversarial comments. Figure 1 shows an example of such an attack. Before the attack, a fake news detector correctly identifies a real article as real. However, using a malicious comment as part of its inputs, the same detector is misled to predict the article as fake instead. Compared with manipulating news title or content, an attack by adversarial comments have several advantages: (i) *accessibility:* as it does not require an ownership over the target article, an attacker can easily create a fake user profile and post malicious comments on any social media news posts; (ii) *vulnerability:* it is less vulnerable than attacking via an article's title or content, as the comments written by general users often have a higher tolerance in their writing quality (e.g., using more informal language, slang, or abbreviations is acceptable in user comments) compared to that of an article's title or content. This makes any efforts to detect adversarial comments more challenging. Despite these advantages, to our best knowledge, there exist few studies on the vulnerability of neural fake news detectors via malicious comments.

Therefore, in this paper, we formulate a novel problem of adversarial comment generation to fool fake news detectors. Generating adversarial comments is non-trivial because adversarial comments that are misspelled or irrelevant to the news can raise a red flag by a defense system and be filtered out before it has a chance to fool fake news detector. Thus, we are faced with two challenges: (i) *how to generate adversarial comments that can fool various cutting-edge fake news detectors to predict target class?*; and (ii) *how to simultaneously generate adversarial comments that are realistic and relevant to the article's content;* In an attempt to solve these challenges, we propose **MALCOM**, a novel framework that can generate realistic and relevant comments in an end-to-end fashion to attack fake news detection models, that works for both black box and white box attacks. The main contributions are:

- This is the first work proposing an attack model against neural fake news detectors, in which adversaries can post malicious comments toward news articles to mislead cutting-edge fake news detectors.
- Different from prior adversarial literature, our work generates adversarial texts (e.g., comments, replies) with *high quality* and *relevancy* at the sentence level in an end-to-end fashion (instead of the manipulation at the character or word level).
- Our model can fool five top-notch neural fake news detectors to *always* output real news and fake news 94% and 93.5% of the time on average. Moreover, our model can mislead black-box classifiers to always output real news 90% of the time on average.

II. Related Work

A. Fake News Detection Models

In terms of computation, the majority of works focus on developing machine learning (ML) based solutions to automatically detect fake news. Feature wise, most models use an article's title, news content, its social responses (e.g., user comments or replies) [27], relationships between subjects and publishers [37] or any combinations of them [25]. Specifically, social responses have been widely adopted and proven to be strong predictive features for the accurate detection of fake news [25], [27]. Architecture wise, most detectors use *recurrent neural network (RNN)* [25], [27] or *convolutional neural network (CNN)* [24] to encode either the news content (i.e., article's content or micro-blog posts) or the sequential dependency among social comments and replies. Other complex architecture includes the use of *co-attention* layers [30] to model the interactions between an article's content and its social comments (e.g., dEFEND [27]) and the adoption of variational auto-encoder to generate synthetic social responses to support early fake news detection (e.g., TCNN-URG [24]).

B. Attacking Fake News Detectors

Even though there have been several works on general adversarial attacks, very few addressed on the attack and defense of fake news detectors. [38] argues that fake news models purely based on *natural language processing (NLP)* features are vulnerable to attacks caused by small fact distortions in the article's content. Thus, they propose to use a fact-based knowledge graph curated from crowdsourcing to augment a classifier. In a similar effort, [12] examines three possible attacks to fake news detectors. They are hiding questionable content, replacing features of fake news by that of real news, and blocking the classifiers to collect more training samples. The majority of proposed attacks leverage an article's title, content, or source. They assume that the attacker has a full ownership over the fake news publication (thus can change title or content). This, however, is *not* always the case. In this paper, therefore, we assume a stricter attack scenario where the attacker has no control over the article's source or content, particularly in the case where the attacker is different from the fake news writer. Moreover, we also conjecture that the attacker can be hired to either: (i) promote fake news as real news and (ii) demote real news as fake news to create confusion among the community [11]. To achieve this, instead of focusing on attacking an article's content or source, we propose to generate and inject **new** malicious comments on the article to fool fake news detectors.

C. Adversarial Text Generation

Text generation is notoriously a complex problem mainly due to the discrete nature of text. Previous literature in text generation include generating clickbaits [19], [29], text with sentiment [13], user responses [24], and fake news [36]. Generating text under adversarial setting, i.e., to attack ML classifiers, is more challenging [10]. Yet there have been tireless efforts to construct adversarial samples to attack text-based ML models [5], [20], [31]. Most of them focus on making marginal modifications (e.g., addition, removal, replacement, etc.) in character [5], [20] or word level [5], [31] of a span of text, either through a set of predefined templates [20] or through a searching mechanism with constraints [5],

Authorized licensed use limited to: SARDAR VALLABHBHAI NATIONAL INSTITUTE OF TECH. Downloaded on November 29,2023 at 08:58:58 UTC from IEEE Xplore. Restrictions apply.

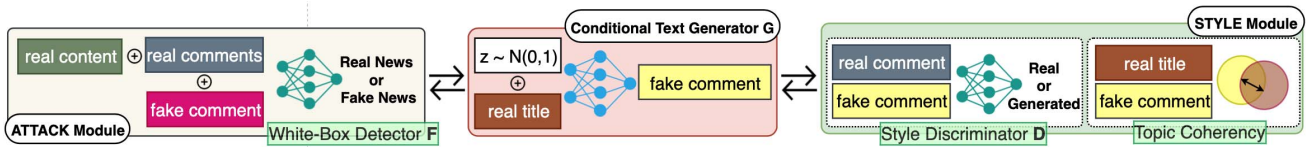

Fig. 2: Malcom Architecture.

[31]. Even though these methods have achieved some degree of success, they are only designed for attacking static features such as the title and content of an article. They are not developed for dynamic sequential input like comments where new text can be added over time. Adversarial text generated by previous methods are usually misspelled ("f@ke" v.s. "fake", "lo ve" v.s. "love") [20], or distorted from the original context or meaning (e.g., [20], [5]). Hence, these attacks can easily be filter-out by a robust word recognizer (e.g. [23]) or even by manual visual examination. Because of this, we propose an end-to-end framework to generate stealthy and context-dependent adversarial comments that achieve a high attack performance.

III. Problem Formulation

We propose to attack fake news detectors with three phrases. Phrase I: identifying target articles to attack. Phrase II: generating malicious comments. Phrase III: appending generated comments on the target articles. In this paper, we focus on the *phrase II* of the attack, which is formally defined as follows. Let $f(\cdot)$ be a target neural network fake news classifier. Denote $\mathcal{X} = \{\boldsymbol{x}_i, C_i\}_{i=1}^N$, $\mathcal{Y} = \{y_i\}_{i=1}^N$ as the features of articles and their ground-truth labels (e.g., fake news or real news) of a dataset $\mathcal{D}$ on which $f(\cdot)$ is trained, with N being the total number of articles. Let $\boldsymbol{x}_i^{title}$, $\boldsymbol{x}_i^{content}$, C_i be the title, content, and a list of all comments of $\boldsymbol{x}_i$, respectively. Then, we want to train a generator G such that, given an unseen article $\{\boldsymbol{x}, C\} \notin \mathcal{X}$ and a target prediction label L^*, G generates a set of M malicious comments C^{adv} to achieve the following objectives:

Objective 1: High quality in writing and relevancy: C^{adv} needs to mimic real comments both in writing quality and relevancy to $\boldsymbol{x}$'s content. This will prevent them from being detected by a robust adversarial text defense system (e.g., [23], [32]). Even though generating realistic comments [29] is not the main goal of our paper, it is a necessary condition for successful attacks in practice.

Objective 2: Successful attacks: This is the main objective of the attacker. The attacker contaminates a set of an article's existing comments C by **appending** C^{adv} such that $f : \boldsymbol{x}, C^* \mapsto L^*$, where $C^* \longleftarrow C \oplus C^{adv}$ with $\oplus$ denoting concatenating, and L^* is the target prediction label. When $L^* \leftarrow 0$, C^{adv} ensures that, after posted, an article $\boldsymbol{x}$ will not be detected by f as fake (and not to be removed from the news channels). When $L^* \leftarrow 1$, C^{adv} helps demote real news as fake news (and be removed from the news channels). There are two types of attacks: (i) white box and (ii) black box attack. In a white box attack, we assume that the attacker has access to the parameters of f. In a black box attack, on the other hand, the f's architecture and parameters are unknown to adversaries. This leads to the next objective below.

Objective 3: Transferability: C^{adv} needs to be transferable across different fake news detectors. In a black box setting, the attacker uses a surrogate white box fake news classifier f^* to generate C^{adv} and transfer C^{adv} to attack other black box models f. Since fake news detectors are high-stack models, we impose a stricter assumption compared to previous literature (e.g., [20]) where public APIs to target fake news classifiers are inaccessible. In practice, the training dataset of an unseen black box model will be different from that of a white box model, yet they can be highly overlapped. Since fake news with reliable labels are scarce and usually encouraged to be publicized to educate the general public (e.g., via fact-check sites), fake news defenders have incentives to include those in the training dataset to improve the performance of their detection models. To simplify this, we assume that both white box and black box models share the same training dataset.

IV. Adversarial Comments Generation

In this paper, we propose MALCOM, an *end-to-end **Mal**icious **Com**ment Generation Framework*, to attack fake news detection models. Figure 2 depicts the Malcom framework. Given an article, Malcom generates a set of malicious comments using a conditional text generator G. We train G together with STYLE and ATTACK modules. While the STYLE module gradually improves the writing styles and relevancy of the generated comments, the ATTACK module ensures to fool the target classifier.

A. Conditional Comment Generator: G

$G(\boldsymbol{x}, \mathbf{z})$ is a conditional sequential text generation model that generates malicious comment c^* by sampling one token at a time, conditioned on (i) previously generated words, (ii) article $\boldsymbol{x}$, and (iii) a random latent variable $\mathbf{z}$. Each token is sequentially sampled according to conditional probability function:

$$p(c^*|\boldsymbol{x}; \theta_G) = \prod_{t=1}^{T} p(c_t^*|c_{t-1}^*, c_{t-2}^*, \ldots, c_1^*; \boldsymbol{x}; \mathbf{z}) \quad (1)$$

where c_t^* is a token sampled at time-step t, T is the maximum generated sequence length, and θ_G is the parameters of G to be learned. G can also be considered as a conditional language model, and can be trained using *MLE* with the *teacher-forcing* [18] by maximizing the negative log-likelihood (*NLL*) for all comments conditioned on the respective articles in $\mathcal{X}$. We want to optimize the objective function:

$$\min_{\theta_G} \mathcal{L}_G^{MLE} = -\sum_{i=1}^{N} c_i \log p(c_i^*|\boldsymbol{x}_i; \theta_G) \quad (2)$$

Authorized licensed use limited to: SARDAR VALLABHBHAI NATIONAL INSTITUTE OF TECH. Downloaded on November 29,2023 at 08:58:58 UTC from IEEE Xplore. Restrictions apply.

B. Style Module

Both writing style and topic coherency are crucial for a successful attack. Due to its high-stake, a fake news detector can be self-guarded by a robust system where misspelled comments or ones that are off-topic from the article's content can be flagged and deleted. To overcome this, we introduce the STYLE module to fine-tune G such that it generates comments with (i) high quality in writing and (ii) high coherency to an article's content.

First, we utilize the *GAN* [8] and employ a comment style discriminator D to co-train with G in an adversarial training schema. We use *Relativistic GAN (RSGAN)* [15] instead of standard GAN loss [8]. In RSGAN, the generator G aims to generate realistic comments to fool a discriminator D, while the discriminator D aims to discriminate whether the comment c is more realistic than randomly sampled fake data generated by G. Specifically, we alternately optimize D and G with the following two objective functions:

$$\begin{aligned}\min_{\theta_G} \mathcal{L}_G^D &= -\mathbb{E}_{(x,c)\sim p_D(\mathcal{X});z\sim p_z}[log(\sigma(D(c) - D(G(\boldsymbol{x}, z)))] \\ \min_{\theta_D} \mathcal{L}_D &= -\mathbb{E}_{(x,c)\sim p_D(\mathcal{X});z\sim p_z}[log(\sigma(D(G(\boldsymbol{x}, z)) - D(c)))]\end{aligned} \tag{3}$$

where σ is a $sigmoid$ function, θ_G is the parameters of G and θ_D is the parameters of D. By using D, we want to generate comments that are free from misspellings while resembling realistic commenting styles.

Second, to enhance the relevancy between the generated comments and the article, we minimize the mutual information gap between comments generated by G and the article's titles. Specifically, we use *maximum mean discrepancy (MMD)*, which has been shown to be effective in enforcing mutual information. The loss function can be written as:

$$\begin{aligned}\min_{\theta_G} \mathcal{L}_G^H &= MMD(\mathcal{X}^{title}, G(\mathcal{X})) \\ &= \big[\mathbb{E}_{\boldsymbol{x},\boldsymbol{x'}\sim p_{\mathcal{D}}(\mathcal{X})}\mathbf{k}(\boldsymbol{x}^{title}, \boldsymbol{x'}^{title}) \\ &+ \mathbb{E}_{\boldsymbol{x}\sim p_{\mathcal{D}}(\mathcal{X});c^*,c^{*'}\sim G(\boldsymbol{x},z)}\mathbf{k}(c^*, c^{*'}) \\ &- 2\mathbb{E}_{\boldsymbol{x}\sim p_{\mathcal{D}}(\mathcal{X}),c^*\sim G(\boldsymbol{x},z)}\mathbf{k}(\boldsymbol{x}^{title}, c^*)\big]^{\frac{1}{2}}\end{aligned} \tag{4}$$

where the MMD compares the distribution $\mathcal{X}^{title}$ of real articles' titles and that of generated comments $G(\mathcal{X})$ by projecting them into Hilbert spaces (RKHS) using a Gaussian kernel $\mathbf{k}$. Intuitively, we want to minimize the information gap between the real titles and the generated comments. Moreover, we use $\boldsymbol{x}^{title}$ (i.e., the title of $\boldsymbol{x}$) instead of $\boldsymbol{x}^{content}$ (i.e., the content of $\boldsymbol{x}$) because: (i) an article's content is usually much longer, hence requiring more computation, (ii) an article's title is a summary of the article's content, and (iii) prior studies (e.g., [6]) show that social media users actually rely more on the headlines rather than the actual content for commenting, sharing, and liking.

C. Attack Module

This module guides G to generate comments that can fool the target classifier. In a white box setting, the fake news classifier can be directly used to guide the learning of G. In a black box setting, a surrogate fake news classifier can be used to generate and transfer malicious comments to unseen fake news detectors. We denote $f(\boldsymbol{x}_i, \mathbf{C}_i)$ parameterized by θ_f as the surrogate white box classifier, predicting whether or not $\boldsymbol{x}_i$ is fake news. f can be trained using *binary-cross-entropy loss* over $\mathcal{D}$ as:

$$\min_{\theta_f} \mathcal{L}_f = -\frac{1}{N}\sum_i^N (\mathbf{y}_i log(f(\boldsymbol{x}_i, C_i))) + (1-\mathbf{y}_i) log(1-f(\boldsymbol{x}_i, C_i))$$

To use this trained model f to guide G, we use signals back-propagated from f to force G to generate a new comment c^* such that $f(\boldsymbol{x}_i, C_i^*)$ (where $C_i^* \longleftarrow C_i \oplus \{c^*\}$) outputs a target prediction label $\mathbf{L}^* \in \{0, 1\}$ for the article $\boldsymbol{x}_i$. Specifically, we want to optimize the objective function:

$$\min_{\theta_G} \mathcal{L}_G^{f(\mathbf{L}^*)} = -\frac{1}{N}\sum_i^N (\mathbf{L}^* log(f(\boldsymbol{x}_i, C_i^*))) + (1-\mathbf{L}^*) log(1-f(\boldsymbol{x}_i, C_i^*)) \tag{5}$$

One obvious attack scenario is for an attacker to promote fake news, i.e., to generate comments to mislead the target classifier to classify fake news as real news ($\mathbf{L}^* \longleftarrow 0$). Adversely, an attacker might also want to fool the target classifier to classify real news as fake news ($\mathbf{L}^* \longleftarrow 1$).

Algorithm 1 Generating Adversarial Comments Algorithm

1: Pre-train G with *teacher-forcing* and MLE using Eq. (2) with *train* set.
2: Pre-train a surrogate fake news classifier f using Eq. (IV-C) with *train* set.
3: **repeat**
4: Training G with D using Eq. (3) in mini-batch from *train* set.
5: Training G using Eq. (4) in mini-batch from *train* set.
6: Training G with f using Eq. (5) in mini-batch from *train* set.
7: **until** convergence

D. Objective Function of Malcom

At the end, an attacker aims to generate realistic and relevant comments to attack a target fake news classifier by optimizing objective functions as follows.

$$\min_{\theta_f} \mathcal{L}_f; \quad \min_{\theta_D} \mathcal{L}_D; \quad \min_{\theta_G} (\mathcal{L}_G^{MLE} + \mathcal{L}_G^D + \mathcal{L}_G^H + \mathcal{L}_G^{f(\mathbf{L}^*)}) \tag{6}$$

where each term in Eq. (6) equally contributes to the final loss function. We use Adam [7] to optimize the objective functions with a mini-batch training approach. Alg. 1 shows the overall training algorithm.

E. Implementation Details

Training with Discrete Data: We need to back-propagate the gradients of the loss in Eq. (3, 4, 5) through discrete tokens sampled by G from the multinomial distribution c_t^* at each time-step t. Since this sampling process is not differentiable, we employ *Gumbell-Softmax* [14] relaxation trick with a τ parameter (i.e., generation temperature) to overcome this. We refer interested readers on the elaborate discussion of the Gumbell-Softmax technique and the effects of τ on generation quality and diversity to [14], [33].

Generation Strategy: For each article $\boldsymbol{x}$, we generate new comment $c \longleftarrow G(\boldsymbol{x}, z)$ where $z \sim \mathcal{N}(0, 1)$. To minimize

Authorized licensed use limited to: SARDAR VALLABHBHAI NATIONAL INSTITUTE OF TECH. Downloaded on November 29,2023 at 08:58:58 UTC from IEEE Xplore. Restrictions apply.

TABLE I: Dataset Statistics and Details of Target Classifiers and Their Fake News Detection Performance

Dataset	#articles	#comments	#fake	#real
GOSSIPCOP	4,792	116,308	1,894	2,898
PHEME	5,476	52,612	1,980	3,486

Classifier	GOSSIPCOP		PHEME	
	Accuracy	F1	Accuracy	F1
f_{CNN}	0.74	0.74	0.77	0.77
f_{RNN}	0.70	0.69	0.71	0.71
CSI\t [25]	0.65	0.70	0.61	0.61
TEXTCNN [16]	0.68	0.68	0.76	0.76
dEFEND [27]	0.76	0.76	0.78	0.78

the risk of being detected by a defender, an attacker desires to select the best set of comments to attack, especially those that are highly relevant to the target article. Hence, for each article $\boldsymbol{x}$, we sample different z to generate different malicious comments and select c that is the most coherent to the article. To measure such the coherency, we derive function $T_k(c, \boldsymbol{x}^{title})$ which will be introduced in Sec. (V-A5).

Architectures and Parameters Setting: We employ *Relational Memory Recurrent Network (LMRN)* [26], [33] and multi discriminative representations (MDR) [33] as the cornerstone of G and D architecture. We also observe that D with a CNN-based architecture works very well in our case. The LMRN model is adapted from the SONNET model[2]. The MDR implementation is publicly available [3]. We will release all datasets, codes, and parameters used in our experiments upon the acceptance of this paper.

V. EVALUATION

In this section, we evaluate the effectiveness of Malcom and try to answer the following analytical questions (AQs):

AQ1 Quality, Diversity, and Coherency: How realistic are the generated comments in terms of their writing styles and as well as coherency to the original articles' contents?

AQ2 Attack Performance: How effective are generated comments in attacking white box and black box detectors?

AQ3 Attack Robust Fake News Detectors: How effective are generated comments in attacking fake news detectors safe-guarded by a robust comments filtering feature?

AQ4 Robustness: How many malicious comments do we need and how early can they effectively attack the detectors?

We plan to release all datasets, codes, and parameters used in our experiments (upon the acceptance of this paper).

A. Experimental Set-Up

1) Datasets: We experiment with two popular public benchmark datasets, i.e., **GOSSIPCOP** [28] and **PHEME** [17]. GOSSIPCOP is a dataset of fake and real news collected from a fact-checking website, *GossipCop*, whereas PHEME is a dataset of rumors and non-rumors relating to nine different breaking events. These datasets are selected because they include both veracity label and relevant social media discourse content on Twitter[4].

[2]https://github.com/deepmind/sonnet

[3]https://github.com/williamSYSU/TextGAN-PyTorch

[4]We exclude another popular dataset, POLITIFACT, also from [28] because it is much smaller and less diverse in terms of topics

2) Data Processing and Partitioning: For each dataset, we first clean all of the comments (e.g., remove mentions, hashtags, URLs, etc.). We also remove non-English comments, and we only select comments that have length from 5 and 20. We split the original dataset into *train* and *test* set with a split ratio of 9:1. Since PHEME dataset does not include articles' contents, we use their titles as alternatives. Table I shows the statistics of the post-processed datasets. We use the *train* set to train both G and target fake news classifiers f. All the experiments are done *only* on the *test* set, i.e., we evaluate quality and attack performance of generated comments on *unseen* articles and their ground-truth comments.

3) Target Classifier: . We experiment Malcom with SOTA and representative fake news classifiers, which are summarized in Table I. Note that both datasets are challenging ones as SOTA methods can only achieve 0.76 and 0.78 in F1 using the first 10 comments of each article. These classifiers are selected because they cover a variety of neural architectures to learn representations of each article and its comments, which is eventually input to a *softmax* layer for prediction. In all of the following classifiers, we encode the article's content into a feature vector of $\mathbb{R}^{512}$ by using Universal Sentence Encoder (USE) [3], followed by a *fully-connected-network (FCN)* layer.

- $\boldsymbol{f_{CNN}}$: This classifier uses CNN layers to encode each comment into a vector. Then, it concatenates the average of all encoded comments with the feature vector of the article's content as the article's final representation.
- $\boldsymbol{f_{RNN}}$: This classifier uses a RNN layer to model the sequential dependency among its comments, output of which is then concatenated with the vectorized form of the article's content as the article's final representation. We utilize *Gated Recurrent Unit (GRU)* as the RNN architecture because it has been widely adopted in previous fake news detection literatures (e.g., [25], [27]).
- **TEXTCNN** [16]: TEXTCNN uses a CNN architecture to encode the mean of vector representations of all of its comments. The output vector is then concatenated with the article's content vector as the article's final representation.
- **CSI\t** [25]: CSI uses *GRU* to model the sequential dependency of textual features of user comments and the network features among users participating in an article's discourse to detect fake news. Different from f_{RNN}, this model does not use an article's content as an input. We use a modified version, denoted as CSI\t, that does not use the network features as such information is not available in both datasets.
- **dEFEND** [27]: Given an article, this algorithm utilizes a co-attention layer between an article's content and comments as input to make a final prediction.

Other methods are surveyed but not included in the experiments because: (i) overall, their accuracies were reported inferior to those of **dEFEND**, **CSI**, (ii) SAME [4] only uses extracted sentiments of the comments, not the whole text as input, (iii) FAKEDETECTOR [37] mainly uses graph-based features, which is not within our scope of study, and (iv) TCNN-URG [24] focuses only on early fake news detection.

Authorized licensed use limited to: SARDAR VALLABHBHAI NATIONAL INSTITUTE OF TECH. Downloaded on November 29,2023 at 08:58:58 UTC from IEEE Xplore. Restrictions apply.

TABLE II: Comparison among Attack Methods

Method	end-to-end generation	generalization via learning	level of attack
CopyCat	○	○	sentence
HotFlip	○	○	character/word
UniTrigger	○	●	multi-level
TextBugger	○	○	character/word
Malcom	●	●	sentence

TABLE III: Examples of Generated Malicious Comment. Spans in **purple** and *italics* are retrieved from the train set and carefully crafted. Spans in **blue** are generated in end-to-end fashion.

Title	why hollywood won't cast renee zellweger anymore
Content	so exactly what led renée zellweger, an oscar (...)
CopyCat	*her dad gave her a great smile*
+HotFlip	*her dad* ~~***gave***~~ ***got*** *her a great smile*
+UniTrigger	***edit season edit*** *her dad gave her a great smile*
+TextBugger	*her dad* ~~***gave***~~ ***ga ve*** *her a great smile*
Malcom	**why do we need to ruin this season**

4) Compared Attack Methods: We compared Malcom with representative and SOTA adversarial text generators (Table II).

- **CopyCat Attack**: We created this method as a trivial attack baseline. CopyCat randomly retrieves a comment from a relevant article in the train set which has the target label. We use *USE* to facilitate semantic comparison among articles' contents.
- **HotFlip Attack [5]**: This attack finds the most critical word in a sentence and replaces it with a similar one to fool the target classifier. Since HotFlip does not generate a whole sentence but make modifications on an existing one, we first use the comment retrieved by CopyCat as the initial malicious comment.
- **Universal Trigger (UniTrigger) Attack [3]**[5]: It searches and appends a *fixed* and *universal* phrase to the end of an existing sentence to fool a text classifier. In this case, we want to find an universal topic-dependent prefix to prepend to every comment retrieved by CopyCat to attack. For a fair comparison and to ensure the coherency with the target article's content, we restrict replacement candidates to the top *q=30* words (for GossipCop dataset) and *q=30* words (for Pheme dataset) representing the article's topic. *q* is chosen such that replacement candidates are distinctive enough among different topics. These words and topics are retrieved from a topic modeling function $LDA_k(\cdot)$.
- **TextBugger Attack [20]**: This method generates "bugs", i.e., carefully crafted tokens, to replace words of a sentence to fool text classifiers. This attack also requires an existing comment to attack. Therefore, we first use CopyCat to retrieve an initial text to attack. Next, we search for "bugs" using one of the following strategies *insert, delete, swap, substitute-c, substitute-w* as described in [20] to replace one of the words in a comment that achieves the attack goal.

5) Evaluation Measures:

1) **Success Attack Rate (Atk%)**: This quantifies the effectiveness of the attack. For example, a target-real attack on f with Atk% score of 80% indicates that an attacker can fool f to predict *real-news* 80% of the time on all news articles that F should have otherwise predicted correctly.

[5]https://github.com/Eric-Wallace/universal-triggers

TABLE IV: Quality, Diversity, Coherency and White Box Attack

Model	GossipCop Dataset			
	↑Quality	↓Diversity	↑Coherency	↑Atk%
CopyCat	0.650	-	0.585	0.497
+HotFlip	0.618	-	0.565	0.803
+UniTrigger	0.545	-	0.725	0.929
+TextBugger	0.643	-	0.561	0.749
Malcom\Style	0.740	2.639	0.659	**0.986**
Malcom	**0.759**	**2.520**	**0.730**	0.981
Model	Pheme Dataset			
	↑Quality	↓Diversity	↑Coherency	↑Atk
CopyCat	0.697	-	0.578	0.784
+HotFlip	0.657	-	0.530	0.958
+UniTrigger	0.608	-	0.595	0.951
+TextBugger	0.617	-	0.528	0.975
Malcom\Style	0.517	2.399	0.732	**1.000**
Malcom	**0.776**	**1.917**	**0.812**	0.966

"-": NLL_{gen} cannot be computed for retrieval-based method
All experiments are averaged across 3 different runs

2) **Quality and Diversity**: We use **BLEU** and *negative-log-likelihood loss* **(NLL_gen)** scores to evaluate how well generated comments are in terms of both quality and diversity, both of which are widely adopted by previous text generation literature (e.g., [10], [33], [34]). While *BLEU* scores depict the quality of the generated text compared with an out-of-sample test set of human-written sentences (the higher the better), *NLL_gen* signals how diverse generated sentences are (the lower the better).
3) **Topic Coherency**: We derive a topic coherency score of a set of arbitrary comments C and its respective set of articles $\mathcal{X}$ of size N as follows: $T_k(X,C) = \frac{1}{N}\sum_{i=0}^{N}[1 - \cos(LDA_k(\boldsymbol{x}_i^{content}), LDA_k(c_i))]$, where $\cos(\cdot)$ is a *cosine similarity* function. $LDA_k(\cdot)$ is a Latent Dirichlet Allocation (LDA) model that returns the distribution of k different topics of a piece of text. We train LDA_k on all of the articles' contents using unsupervised learning with hyper-parameter k. The larger the score is, the more topic-coherent the comment gets to the article. Because different comments generation algorithms work well with different values of k, for a fair comparison, we report the averaged topic coherency across different values of k as the final **Coherency score**: $Coherency = \sum_{k\in\mathcal{K}} T_k(X,C)$. We select $k \in \mathcal{K}$ such that the averaged entropy of topic assignment of each article is minimized, i.e., to ensure that the topic assigned to each article is distinctive enough to have meaningful evaluation.

B. *AQ1. Quality, Diversity and Coherency*

Tables III and IV show the examples of the generated comments by all attacks and their evaluation results on the quality, diversity, and topic coherency. Malcom generates comments with high quality in writing and topic coherency. However, we do not discount the quality of human-written comments. The reason why BLEU scores of real comments, i.e., CopyCat, are lower than that of Malcom is because they use a more diverse vocabulary and hence reduce n-gram matching chances with reference text in the *test* set. The user-study in Sec. VI-A will later show that it is also not trivial to differentiate between Malcom-generated and human-written comments even for human users.

Authorized licensed use limited to: SARDAR VALLABHBHAI NATIONAL INSTITUTE OF TECH. Downloaded on November 29,2023 at 08:58:58 UTC from IEEE Xplore. Restrictions apply.

TABLE V: **Black Box** Attack Performance on Different Attack Strategies and Target Classifier Architectures (Atk%)

Attack/Model	GOSSIPCOP Dataset						PHEME Dataset					
	f^*_{RNN}	f_{RNN}	f_{CNN}	CSI\t	TEXTCNN	dEFEND	f^*_{RNN}	f_{RNN}	f_{CNN}	CSI\t	TEXTCNN	dEFEND
BASELINE	0.416	0.509	0.499	0.516	0.548	0.652	0.533	0.514	0.498	0.606	0.537	0.543
COPYCAT	0.497	0.689	0.688	0.670	0.774	0.802	0.784	0.783	0.766	0.821	0.716	0.644
+HOTFLIP	0.803	0.813	0.765	0.820	0.838	0.866	0.958	0.850	0.845	0.879	0.811	0.711
+UNITRIGGER	0.929	0.763	0.722	0.803	0.745	0.817	0.951	0.783	0.782	0.783	0.781	0.730
+TEXTBUGGER	0.749	0.736	0.742	0.742	0.784	0.832	0.975	0.832	0.852	0.872	0.823	0.705
MALCOM\STYLE	**0.986**	**0.973**	0.939	0.875	**0.888**	**0.930**	**1.000**	**0.959**	**0.965**	0.880	**0.963**	**0.865**
MALCOM	0.981	0.963	**0.941**	**0.911**	0.876	0.912	0.966	0.893	0.893	**0.888**	0.889	0.760

(*) indicates white box attacks. All experiments are averaged across 3 different runs. MALCOM\STYLE: MALCOM *without* the STYLE module.

Different from all recent attacks, Malcom is an end-to-end generation framework, and can control the trade-off between quality and diversity by adjusting the τ parameter accordingly (Sec. IV-E). Thus, by using a less diverse set of words that are highly relevant to an article, Malcom can focus on generating comments with both high quality and coherency. The STYLE module significantly improves the writing style and boosts the relevancy of generated comments. Without the STYLE module, we observe that a model trained with only the ATTACK module will quickly trade in writing style for attack performance, and eventually become stuck in mode-collapse, i.e., when the model outputs only a few words repeatedly. We also observe that the STYLE module helps strike a balance between topic coherency and attack performance.

C. *AQ2*. Attack Performance

In this section, we evaluate Atk% of all attack methods under the most optimistic scenario where the target article is just published and contains no comments. We will evaluate their attack robustness under other scenarios in later sections.

1) White Box Attack: We experiment a white box attack with f_{RNN} target classifier. *RNN* architecture is selected as the white box attack due to its prevalent adoption in various fake news and rumors detection models. Table V describes white box attack in the first column of each dataset. We can observe that Malcom is very effective at attacking white box models (98% Atk% and 97% Atk% in GOSSIPCOP and PHEME dataset). Especially, MALCOM\STYLE is able to achieve near perfect Atk% scores. Comparable to Malcom are UNITRIGGER and TEXTBUGGER. While other attacks such as TEXTBUGGER only performs well in one dataset, Malcom and UNITRIGGER perform consistently across the two datasets with a very diverse writing styles. This is thanks to the "learning" process that helps them to generate malicious comments from not only a single but a set of training instances. On the contrary, TEXTBUGGER for example, only exploits a specific pre-defined weakness of the target classifier (e.g. the vulnerability where misspellings are usually encoded as unknown tokens [20]) and requires no further learning.

2) Black Box Attack: Let's use a surrogate f_{RNN} model as a proxy target classifier to generate malicious comments to attack black box models. We test their transferability to black box attacks to five unseen fake news classifiers described in Sec. V-A3. Table V shows that comments generated by MALCOM does not only perform well on white box but also on black box attacks, achieving the best transferability across all types of black box models. Our method is especially able

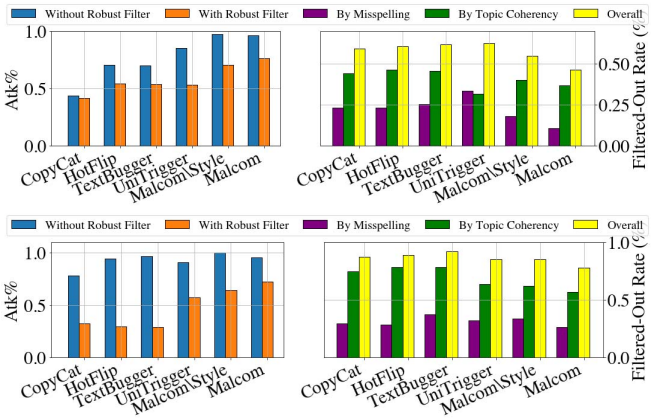

Fig. 3: Attack Robust Fake News Detector. Top: GOSSIPCOP Dataset. Bottom: PHEME Dataset

to attack well-known models such as CSI\t and dEFEND with an average of 91% and 85% of Atk% in GOSSIPCOP and PHEME dataset. However, other strong white box attacks such as UNITRIGGER, TEXTBUGGER and HOTFLIP witness a significant drop in Atk% with black box target classifiers. Particularly, UNITRIGGER experiences the worst transferability, with its Atk% drops from an average of 94% to merely over 77% across all models in both datasets. On the other hand, Malcom performs as much as 90% Atk% across all black box evaluations. This shows that our method generalizes well not only on fake news detector trained with different set of inputs (e.g., with or without title), but also with different modeling variants (e.g., with or without modeling dependency among comments) and architectures (*RNN*, *CNN*, *Attention*).

D. *AQ3*. Attack Robust Fake News Detection

This section evaluates attack performance of all methods under a *post training defense*. This defense comes after the target classifier has already been trained. Before prediction, we use a robust word recognizer called SCRNN[6] [23] to measure the number of misspellings and detect manipulations in the comments. We also use the *Coherency* (Sec. V-A5) to measure the topic relevancy and set a topic coherency threshold between comments and the target article to filter-out irrelevant comments. We remove any comment that either has more than one suspicious word or have *Coherency* lower than that of the article's title with an allowance margin of 0.05. This defense system is selected because it does not make any assumption on any specific attack methods, hence it is both

[6]https://github.com/danishpruthi/Adversarial-Misspellings

Authorized licensed use limited to: SARDAR VALLABHBHAI NATIONAL INSTITUTE OF TECH. Downloaded on November 29,2023 at 08:58:58 UTC from IEEE Xplore. Restrictions apply.

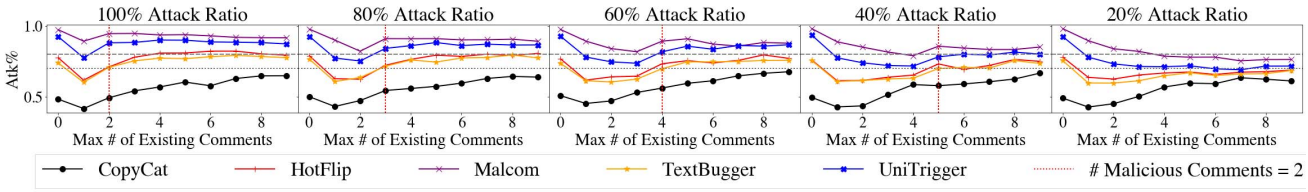

Fig. 4: Robustness of Inter-Attacks: White Box Setting on GOSSIPCOP Dataset

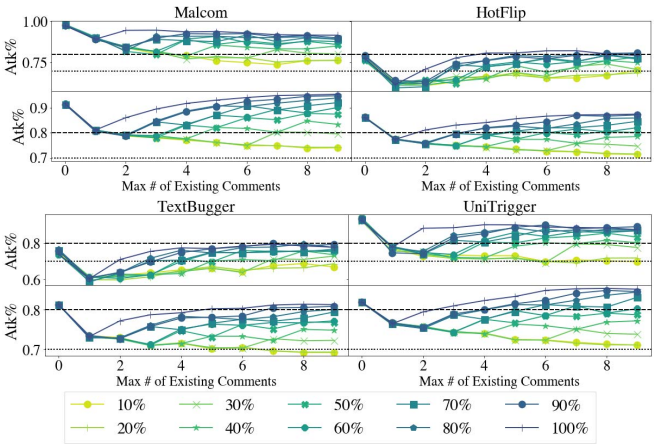

Fig. 5: Robustness of Intra-Attacks: White Box Setting (First Row) & Black Box (Second-Row) on GOSSIPCOP Dataset.

general and practical. We measure both Atk% and the *filter-out rate*, i.e., the percentage of comments that are removed by the defense system, for all of the attack methods.

Figure 3 shows that our method achieves the best Atk% even under a rigorous defense in both datasets. While Atk% of HOTFLIP and TEXTBUGGER drop significantly (around ↓66% and ↓68%) under the defense, that of MALCOM\STYLE decreases to only 0.64% from a nearly perfect Atk%. This confirms that the STYLE module is crucial for generating stealthy comments. Figure 3 also shows that Malcom is the best to bypass the defense system, achieving better, i.e., lower, *filter-out rate* in terms of misspellings compared with real comments retrieved by COPYCAT method. This is because Malcom emphasizes writing quality over diversity to be more stealthy under a robust defense algorithm. Moreover, around 1/3 of real comments selected by COPYCAT are filtered-out by SCRNN. This confirms that real comments written on social media are messy and not always free from grammatical errors.

E. *AQ4. Robustness*

There is always a trade-off between the # of comments collected for prediction and how early to detect fake news. This section evaluates Atk% w.r.t different # of existing comments for early fake news detection. We evaluate on GOSSIPCOP dataset as an example. We assume that the target classifier only has access to a few existing comments, a maximum of 20 in this case. First, we define the *attack ratio* as the ratio of # of malicious comments over # of existing comments. Next, we evaluate the robustness among all attack methods in two categories, namely *Inter- and Intra-Comparison*.

TABLE VI: Results of User-Study on Generation Quality

Hypothesis	z-score	p-value	Accuracy	#response
$\mathcal{H}_1$	1.4940	0.0676	0.6087	46
$\mathcal{H}_2$	1.2189	0.1114	0.5818	56
$\mathcal{H}_3$	0.9122	0.1808	0.5416	120

Inter-Comparison: How attack methods perform differently with the same attack ratio? Figure 4 shows that Malcom outperforms other baselines under all attack ratios. Moreover, our method consistently maintains Atk% of at least 80% under a different # of existing comments with an attack ratio as low as 40%. On the contrary, to achieve the same performance, UNITRIGGER, the second-best attack in terms of robustness, would require a 100% attack ratio. *Intra-Comparison: For each method, how many malicious comments are needed for an effective attack performance under different # of existing comments?* Figure 5 shows the results on white box and black box attacks. Under the white box attack, all methods display more or less the same performance with more than 1 malicious comment. However, the black box setting observes more variance across different attack ratios. Specifically, Malcom's performance continuously improves as the # of existing comments increases under any attack ratios $\geq$ 40%. On the contrary, existing attacks show little improvement even with an increasing # of malicious comments.

VI. DISCUSSION

A. *Prevent Malicious Comments with Human Support*

We examine whether malicious comments generated by MALCOM can be easily flagged by human, i.e., the Turing Test. We use Amazon Mechanical Turk (AMT) to recruit over 100 users to distinguish comments generated by MALCOM (machine-generated) and human. We examine the following alternative hypothesises using one-tailed statistical testing.

1) $\mathcal{H}_1$: Given a comment, the users can correctly detect if the comment is generated by machine (not by human).
2) $\mathcal{H}_2$: Given a comment, the users can correctly detect if the comment is generated by human (not by machine).
3) $\mathcal{H}_3$: Given a machine-generated and a human-written comment, the users can correctly identify the machine-generated.

For quality assurance, we recruit only the users with 95% approval rate, randomly swap the choices and discard responses taking less than 30 seconds. We test on comments generated for 187 unseen and unique articles in the PHEME dataset's *test* set. Table VI shows that we *fail to reject* the null-hypothesises of both $\mathcal{H}_1, \mathcal{H}_2$ and $\mathcal{H}_3$ (p-value $>$ 0.05). While comments generated by MALCOM is not perfectly

Authorized licensed use limited to: SARDAR VALLABHBHAI NATIONAL INSTITUTE OF TECH. Downloaded on November 29,2023 at 08:58:58 UTC from IEEE Xplore. Restrictions apply.

stealthy (accuracy of $\mathcal{H}_1 > 0.5$), it is still very challenging to distinguish between human-written and MALCOM-generated comments. In fact, human-written comments on social media are usually messy, which lead users to be unable to distinguish between machine-generated ones (accuracy of $\mathcal{H}_2 < 0.6$). Thus, even if human are employed to filter out suspicious or auto-generated comments with a mean accuracy of 60% ($\mathcal{H}_1$), MALCOM can still effectively achieve over 80% Atk% on average with a remaining 40% of the malicious comments (see Sec. V-E). Hence, we need to equip human workers with intensive training to better identify malicious comments. Nevertheless, this can be labor intensive and costly due to a large amount of comments published everyday.

B. Prevent Malicious Comments with Machine Support

One advantage of the defense system introduced in Sec. V-D is that filtering out comments based on misspellings and topic coherency does not make any assumption on any specific attack methods, hence it is both general and practical. In the most optimistic scenario where we expects *only* attacks from MALCOM, we can train a ML model to detect MALCOM-generated comments. We use LIWC [22] dictionary to extract 91 psycholinguistics features and use them to train a *Random Forest* classifier to differentiate between human-written, i.e., COPYCAT, and MALCOM-generated comments based on their linguistic patterns. The 5-fold cross-validation accuracy is $0.68(+/-0.1)$, which means around 70% and 30% of MALCOM-generated and human-written comments will be flagged and removed by the classifier. From Figure 5, if the initial *attack ratio* of 100%, one can then effectively put an upper-bound of around 80% Atk% rate on MALCOM (new *attack ratio* of 40%).

Other ways to defend against Malcom is to only allow users with verified identifications to publish or engage in discussion threads, or to utilize a fake account detection system (e.g., [35]) to weed out suspicious user accounts. We can also exercise adversarial learning [9] and train a target fake news classifier together with malicious comments generated by potential attack methods. Social platforms should also develop their own proprietary fake news training dataset and rely less on public datasets or fact-checking resources such as *GossipCop* and *PolitiFact*. While this may adversely limit the pool of training instances, it will help raise the bar for potential attacks.

C. Real News Demotion Attack

An attacker can be paid to either promote fake news or demote real news. By demoting real news, in particular, not only can the attacker cause great distrust among communities, but the attacker can also undermine the credibility of news organizations and public figures who publish and share the news. In fact, Malcom can also facilitate such an attack, i.e., to fool fake news detectors to classify real news as fake news, by simply specifying the target label $L^* \longleftarrow 1$ (Sec. IV-C). Our experiments show that Malcom can achieve a real news demotion white box attack with Atk% of around 92% and 95% in GOSSIPCOP and PHEME datasets. Figure 1 shows an example of a real news demotion attack. The real news article was posted by *The Guardian*, a reputable news source, on Twitter on June 4, 2018. The article is first correctly predicted as real news by a *RNN*-based fake news classifier. However, the attacker can post a malicious yet realistic-looking comment "*he's a conservative from a few months ago*" to successfully fool the classifier to predict the article as fake instead.

TABLE VII: Ablation Test

Models	GOSSIPCOP Dataset			
	↑Quality	↓Diversity	↑Coherency	↑Atk%
\STYLE\ATTACK	0.645	**1.664**	0.727	0.392
\ATTACK	0.741	2.032	**0.769**	0.434
\STYLE	0.740	2.639	0.659	**0.986**
MALCOM	**0.759**	2.520	0.730	0.981
Models	**PHEME Dataset**			
	↑Quality	↓Diversity	↑Coherency	↑Atk%
\STYLE\ATTACK	0.759	**1.273**	0.845	0.519
\ATTACK	0.741	**0.786**	1.431	**0.850**
\STYLE	0.517	2.399	0.732	**1.000**
MALCOM	0.776	1.917	**0.812**	0.966

\STYLE, \ATTACK: MALCOM with the STYLE, ATTACK module removed.

D. Ablation Test

This section carries out ablation test to show the effectiveness of STYLE and ATTACK component of Malcom. Specifically, we valuate the quality, diversity, coherency and as well as white box attack performance of different variants of Malcom. Figure VII demonstrates that STYLE module enhances the writing quality and coherency by large margin from the model without STYLE module. Especially, STYLE module is crucial in improving topic coherency score, which then makes generated comments more stealthy under robust fake news defense system (Sec. V-D). Figure VII also shows that ATTACK module is critical in improving attack performance. While STYLE and ATTACK each trades off quality, coherency with attack success rate and vice versa, the full Malcom achieves a balanced performance between a good writing style and high attack success. This makes our framework both powerful and practical.

E. Baselines' Dependency on COPYCAT

Compared to Malcom, one disadvantage of HOTFLIP, UNITRIGGER and TEXTBUGGER is that they all require an initial comment to manipulate. In theory, manually crafting an initial comment is feasible, yet demands a great labor cost. In practice, an attacker can directly use the target's title or an existing comment as the initial comment to begin the attack. Instead, in this paper, we use COPYCAT to retrieve the initial comment. COPYCAT considers both the topic of the target article and the target label into consideration. Hence, it can help complement other baseline attacks in terms of both Atk% and topic coherency. Our experiments show that attacks using comments retrieved by COPYCAT achieve much better averaged Atk% across both white box and black box attacks (89% Atk%), compared to the ones using the title (75% Atk%) or a single existing comment (78% Atk%) of the target article. This further justifies the use of COPYCAT together with other baseline attacks in our experiments.

Authorized licensed use limited to: SARDAR VALLABHBHAI NATIONAL INSTITUTE OF TECH. Downloaded on November 29,2023 at 08:58:58 UTC from IEEE Xplore. Restrictions apply.

VII. Limitations and Future Work

In this work, we assume that the attacker and the model provider share the same training dataset. In practice, their training datasets might be overlapped but not exactly the same. Moreover, whether or not comments generated using one sub-domain (e.g., political fake news) can be transferable to another (e.g., health fake news) is also out of scope of this paper. Hence, we leave the investigation on the proposed attack's transferability across different datasets for future work. Moreover, we also plan to extend our method to attack graph-based fake news detectors (e.g., [24]), and evaluate our model with other defense mechanisms such as adversarial learning, i.e., to train the target fake news classifier with both real and malicious comments to make it more robust. We also want to exploit similar attack strategy in areas that utilize sequential dependency among text using ML such as fake reviews detection.

VIII. Conclusion

To our best knowledge, this paper is the first attempt to attack existing neural fake news detectors via malicious comments. Our method does not require adversaries to have an ownership over the target article, hence becomes a practical attack. We also introduce Malcom, an end-to-end malicious comments generation framework that can generate realistic and relevant adversarial comments to fool five of most popular neural fake news detectors to predict fake news as real news with attack success rates of 94% and 90% for a white box and black box settings. Not only achieving significantly better attack performances than other baselines, Malcom is shown to be more robust even under the condition when a rigorous defense system works against malicious comments. We also show that Malcom is capable of not only promoting fake news but also demoting real news. Due to the high-stakes of detecting fake news in practice, in future, we hope that this work will attract more attention from the community towards developing fake news detection models that are accurate yet resilient against potential attacks.

References

[1] M. Aldwairi and A. Alwahedi, "Detecting fake news in social media networks," *Procedia Computer Science*, vol. 141, pp. 215–222, 2018.

[2] J. Allen, B. Howland, M. Mobius, D. Rothschild, and D. J. Watts, "Evaluating the fake news problem at the scale of the information ecosystem," *Science Advances*, vol. 6, no. 14, p. eaay3539, 2020.

[3] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar *et al.*, "Universal sentence encoder," *arXiv preprint arXiv:1803.11175*, 2018.

[4] L. Cui and S. W. D. Lee, "SAME: Sentiment-Aware Multi-Modal Embedding for Detecting Fake News," *IEEE/ACM ASONAM'19*.

[5] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou, "Hotflip: White-box adversarial examples for text classification," *arXiv preprint arXiv:1712.06751*, 2017.

[6] M. Gabielkov, A. Ramachandran, A. Chaintreau, and A. Legout, "Social Clicks: What and Who Gets Read on Twitter?" Jun. 2016. [Online]. Available: https://hal.inria.fr/hal-01281190

[7] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[8] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *NIPS'14*.

[9] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *arXiv preprint arXiv:1412.6572*, 2014.

[10] B. Guo, H. Wang, Y. Ding, S. Hao, Y. Sun, and Z. Yu, "c-TextGen: Conditional Text Generation for Harmonious Human-Machine Interaction," *arXiv preprint arXiv:1909.03409*.

[11] M. Hindman and V. Barash, "Disinformation, and influence campaigns on twitter," *Knight Foundation: George Washington University*, 2018.

[12] B. D. Horne, J. Nørregaard, and S. Adali, "Robust fake news detection over time and attack," *ACM TIST'9*.

[13] Z. Hu, Z. Yang, X. Liang, R. Salakhutdinov, and E. P. Xing, "Toward controlled generation of text," in *ICML'17*.

[14] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," *arXiv preprint arXiv:1611.01144*, 2016.

[15] A. Jolicoeur-Martineau, "The relativistic discriminator: a key element missing from standard gan," *arXiv preprint arXiv:1807.00734*, 2018.

[16] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[17] E. Kochkina, M. Liakata, and A. Zubiaga, "All-in-one: Multi-task learning for rumour verification," in *ACL'18*.

[18] A. M. Lamb, A. G. A. P. Goyal, Y. Zhang, S. Zhang, A. C. Courville, and Y. Bengio, "Professor forcing: A new algorithm for training recurrent networks," in *NIPS'16*.

[19] T. Le, K. Shu, M. D. Molina, D. Lee, S. S. Sundar, and H. Liu, "5 sources of clickbaits you should know! using synthetic clickbaits to improve prediction and distinguish between bot-generated and human-written headlines," *IEEE/ACM ASONAM'19*.

[20] J. Li, S. Ji, T. Du, B. Li, and T. Wang, "TextBugger: Generating Adversarial Text Against Real-world Applications," *NDSS'18*.

[21] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," *IEEE Euro S&P'16*.

[22] J. W. Pennebaker, R. L. Boyd, K. Jordan, and K. Blackburn, "The development and psychometric properties of liwc2015," 2015. [Online]. Available: https://eprints.lancs.ac.uk/id/eprint/134191

[23] D. Pruthi, B. Dhingra, and Z. C. Lipton, "Combating adversarial misspellings with robust word recognition," *arXiv preprint arXiv:1905.11268*, 2019.

[24] F. Qian, C. Gong, K. Sharma, and Y. Liu, "Neural user response generator: Fake news detection with collective user intelligence." in *IJCAI'18*.

[25] N. Ruchansky, S. Seo, and Y. Liu, "CSI: A hybrid deep model for fake news detection," in *CIKM'17*.

[26] A. Santoro, R. Faulkner, D. Raposo, J. Rae, M. Chrzanowski, T. Weber, D. Wierstra, O. Vinyals, R. Pascanu, and T. Lillicrap, "Relational recurrent neural networks," in *NIPS'18*.

[27] K. Shu, L. Cui, S. Wang, D. Lee, and H. Liu, "dEFEND: Explainable Fake News Detection," *KDD'19*.

[28] K. Shu, D. Mahudeswaran, S. Wang, D. Lee, and H. Liu, "FakeNewsNet: A Data Repository with News Content, Social Context and Dynamic Information for Studying Fake News on Social Media," *arXiv preprint arXiv:1809.01286*, 2018.

[29] K. Shu, S. Wang, T. Le, D. Lee, and H. Liu, "Deep headline generation for clickbait detection," in *ICDM'18*.

[30] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS'17*.

[31] E. Wallace, S. Feng, N. Kandpal, M. Gardner, and S. Singh, "Universal adversarial triggers for nlp," *arXiv preprint arXiv:1908.07125*, 2019.

[32] Z. Wang, H. Liu, J. Tang, S. Yang, G. Y. Huang, and Z. Liu, "Learning multi-level dependencies for robust word recognition," *arXiv preprint arXiv:1911.09789*, 2019.

[33] N. N. Weili Nie and A. Patel, "RelGAN: Relational Generative Adversarial Networks for Text Generation," in *ICLR'19*.

[34] L. Yu, W. Zhang, J. Wang, and Y. Yu, "SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient," in *AAAI'17*.

[35] D. Yuan, Y. Miao, N. Z. Gong, Z. Yang, Q. Li, D. Song, Q. Wang, and X. Liang, "Detecting fake accounts in online social networks at the time of registrations," in *CCS'19*, pp. 1423–1438.

[36] R. Zellers, A. Holtzman, H. Rashkin, Y. Bisk, A. Farhadi, F. Roesner, and Y. Choi, "Defending against neural fake news," *arXiv preprint arXiv:1905.12616*, 2019.

[37] J. Zhang, B. Dong, and P. Yu, "FAKEDETECTOR: Effective fake news detection with deep diffusive neural network," in *ICDE'20*.

[38] Z. Zhou, H. Guan, M. M. Bhat, and J. Hsu, "Fake news detection via NLP is vulnerable to adversarial attacks," in *ICAART'19*.

Authorized licensed use limited to: SARDAR VALLABHBHAI NATIONAL INSTITUTE OF TECH. Downloaded on November 29,2023 at 08:58:58 UTC from IEEE Xplore. Restrictions apply.