Automated Crowdturfing Attacks and Defenses in Online Review Systems

Yuanshun Yao
ysyao@cs.uchicago.edu
University of Chicago

Bimal Viswanath
viswanath@cs.uchicago.edu
University of Chicago

Jenna Cryan
jennacryan@cs.uchicago.edu
University of Chicago

Haitao Zheng
htzheng@cs.uchicago.edu
University of Chicago

Ben Y. Zhao
ravenben@cs.uchicago.edu
University of Chicago

ABSTRACT

Malicious crowdsourcing forums are gaining traction as sources of spreading misinformation online, but are limited by the costs of hiring and managing human workers. In this paper, we identify a new class of attacks that leverage deep learning language models (Recurrent Neural Networks or RNNs) to automate the generation of fake online reviews for products and services. Not only are these attacks cheap and therefore more scalable, but they can control rate of content output to eliminate the signature burstiness that makes crowdsourced campaigns easy to detect.

Using Yelp reviews as an example platform, we show how a two phased review generation and customization attack can produce reviews that are indistinguishable by state-of-the-art statistical detectors. We conduct a survey-based user study to show these reviews not only evade human detection, but also score high on "usefulness" metrics by users. Finally, we develop novel automated defenses against these attacks, by leveraging the lossy transformation introduced by the RNN training and generation cycle. We consider countermeasures against our mechanisms, show that they produce unattractive cost-benefit tradeoffs for attackers, and that they can be further curtailed by simple constraints imposed by online service providers.

CCS CONCEPTS

• **Security and privacy → Social aspects of security and privacy**; • **Computing methodologies → Natural language generation**; **Neural networks**;

KEYWORDS

Web Security; Crowdturfing; Fake Review; Opinion Spam

CCS '17, October 30-November 3, 2017, Dallas, TX, USA
© 2017 Association for Computing Machinery.
ACM ISBN 978-1-4503-4946-8/17/10...$15.00
https://doi.org/10.1145/3133956.3133990

1 INTRODUCTION

The Internet is no longer the source of reliable information it once was. Today, misinformation is being used as a tool to harm competitors, win political campaigns, and sway public opinion. Clashes between conflicting accounts occur daily on social networks and online discussion forums, and the trustworthiness of many online information sources is now in question.

One highly effective weapon for spreading misinformation is the use of crowdturfing campaigns [30, 46, 74], where bad actors pay groups of users to perform questionable or illegal actions online. Crowdturfing marketplaces are the corrupt equivalents of Amazon Mechanical Turk, and are rapidly growing in China, India and the US [30, 74]. For example, an attacker can pay workers small amounts to write negative online reviews for a competing business, often fabricating nonexistent accounts of bad experiences or service. Since these are written by real humans, they often go undetected by automated tools looking for software attackers.

Thankfully, two factors limit the growth and impact of crowdturfing campaigns. First, they require monetary compensation for each task performed. Larger campaigns can incur significant cost on an attacker, and this limits the scale of many campaigns. Second, the predictable reaction of workers often produce actions (and output) synchronized in time, which can be effectively used as a feature to classify and identify crowdturfing campaigns [30, 74]. A more knowledgeable attacker can apply adversarial techniques (*e.g.* poisoning training data, targeted evasion) against machine learning (ML) classifiers, but such techniques have limited impact, and require significant coordination across workers [73].

But just as ML classifiers can effectively detect these attacks, advances in deep learning and deep neural networks (DNNs) can also serve to make these attacks much more powerful and difficult to defend. Specifically, we believe that in limited application contexts, DNNs have reached a point where they can produce sufficiently clear and correct content effectively indistinguishable from those produced by humans. To illustrate our point, we focus on the domain of online reviews for e-commerce products and services, where millions of users upload reviews to sites such as Yelp, TripAdvisor and Amazon. Online reviews tend to be short, and focused on a limited range of topics defined by the application domain, *e.g.* quality of food and service at a restaurant. We believe that well designed and tuned DNNs are now capable of producing realistic online reviews. If successful, attack campaigns using DNN-based fake reviews would be much more powerful, because they are highly scalable (no per-review payments to human writers) and harder to detect, as scripts can control the rate of review generation to avoid the telltale burstiness that makes crowdsourced reviews so easily detectable [30, 73].

In this paper, we identify a class of attacks based on DNN-based fake review generation. We demonstrate that DNN-based review generators are practical and effective, using a combination of ML-trained review generation and context-based customization. Using Yelp restaurant reviews as a target platform, we show empirically that synthetic reviews generated by our tools are effectively indistinguishable from real reviews by state-of-the-art detectors relying on linguistic features. We carry out a user study (N=600) and show that not only can these fake reviews consistently avoid detection by real users, but they provide the same level of user-perceived "usefulness" as real reviews written by humans.

We then examine potential defenses, and propose an ML-classifier based defense that leverages the inherent computational limitations of most RNNs against the attacker. This leverages the fact that generative language models build fixed memory presentations of the entire training corpus, which limits the amount of information that can be captured from the training corpus. We show that the cycle of processing real reviews through a RNN-based model training and text generation is lossy, and the resulting loss can be detected by comparing the character level distribution of RNN-generated reviews against those written by real users. We also consider potential countermeasures, and show that increasing model complexity produces diminishing returns in evasion, while resource costs increase dramatically.

In summary, our work produces several key takeaways:

(1) We demonstrate the feasibility of automated generation of product reviews for online review sites, using a RNN-based approach for review generation and customization. Our key insight is that while automated generation of arbitrary length content is challenging, generation of shorter text in fixed application domains is practical today.
(2) We show that RNN-based synthetic reviews are robust against state of the art statistical and ML-based detectors. In addition, our user-study shows they are largely indistinguishable from real reviews to human readers, and appear to provide similar levels of "usefulness" utility as determined by readers.
(3) We propose a novel defense that leverages the information loss inherent in an RNN training process to identify statistically detectable variations in the character-level distribution of machine-generated reviews. We show that our defense is robust against countermeasures, and that avoiding detection involves the attacker paying rapidly accelerating costs for diminishing returns.

We believe this is a practical new attack that can have significant impact on not only user-generated review sites like Yelp, but potential attacks on content generation platforms such as Twitter and online discussion forums. We hope these results will bring attention to the problem and encourage further analysis and development of new defenses.

2 PRELIMINARIES

We begin our discussion with background material on online review systems, and content generation based on deep learning networks (RNNs in particular). For simplicity, we focus our discussion on online review systems such as Yelp, Amazon and TripAdvisor.

2.1 Crowdsourced Attacks on Review Systems

Most popular e-commerce sites today rely on user feedback to rate products, services, and online content. Crowdsourced feedback typically includes a *review* or opinion, describing a user's experience of a product or service, along with a rating score (usually on a 1 to 5 scale).

Unfortunately, many review systems are plagued by *fake reviews*, *e.g.,* Yelp [7], Amazon [37], iTunes [40] and TripAdvisor [6], where an attacker manipulates crowd opinion using fake or deceptive reviews. To boost their reputation or to damage that of a competitor, businesses can solicit fake reviews that express an overly positive or negative opinion about a business [7, 30, 74]. Studies on Yelp found that a one star rating increase for restaurants can lead to a 5–9% boost in revenue [34].

Sites like Yelp and Amazon have been consistently engaged in a cat and mouse battle with fake reviews, as attackers try to adapt and bypass various defense schemes [18]. Yelp's *review filter* system flags suspicious reviews and even raises an alert to the consumer if a business is suspected of engaging in large-scale opinion manipulation [80].

Recently, attacks have been known to generate highly deceptive (authentic looking) fake reviews written by paid users. Much of this comes from malicious crowdsourcing marketplaces, known as *crowdturfing* systems, where a large pool of human workers provide on-demand effort for completing various malicious tasks [54, 59]. In the next section, we introduce an attack powered by an AI program that can replace human writers and achieve high attack success.

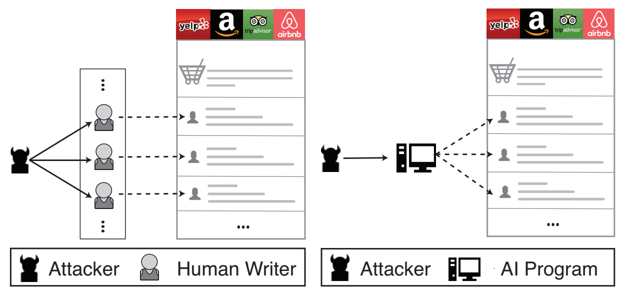
2.2 Our Attack Model

We assume the attacker's goal is to use an AI program to generate fake reviews that are indistinguishable from real reviews written by human users. We only focus on the generation of review text, which is crucial to deceive users and to manipulate their opinion. We do not consider the manipulation of metadata associated with a review or reviewer. Metadata can include any information other than the textual content, *e.g.,* reviewer reputation, review history, posting date and IP address.

Key Insight. There have been significant recent advances in building probabilistic *generative language models* on Neural Networks, specifically *Recurrent Neural Networks (RNNs).* Even trained on large datasets, RNNs have generally fallen short in producing writing samples that truly mimic human writing [50]. However, our insight is that the quality of RNN-generated text is likely more than sufficient for applications relying on domain-specific, short length user-generated content, *e.g.,* online reviews.

Assumptions.

- The attacker has access to a corpus of real reviews to train the generative language model. Popular sites like Yelp have already released large review datasets [79]. Attackers can also



(a) Human-based attack. (b) Machine-based attack.

Figure 1: Fake review attack: Human-based vs. Machine-based.

download reviews, or large review datasets made public by researchers [15, 35, 38].

- The attacker has knowledge of the domain of a product (*e.g.,* cameras) or business (*e.g.,* restaurants, clothing stores) which allows training on a review corpus that matches the domain.
- The attacker has access to sufficient computational resources for training neural networks. Today, commodity GPU machines can efficiently train DNNs, and can be easily purchased or accessed in the Cloud [36].

2.3 RNNs vs. Crowdsourced Authors

Traditional attacks using fake reviews typically hire human writers to write reviews. Instead, our work considers automated, machine-based review attacks leveraging DNN-based language models (see Figure 1). Here, we compare the two approaches and highlight the benefits of automated review attacks.

The key difference between these two approaches is the quality of writing in the generated text content. To influence user opinion and alter decisions, fake reviews need to be written in such a way to mimic content written by real users. Broken grammar, misspellings and broken context can make a review appear fake. Existing machine-based text generations techniques, such as *n-gram models* and *template-based models* are known to have limitations in generating realistic-looking text [42, 77]. Hence, the reviews generated based on those techniques are likely to be identified as fake by readers [82]. In contrast, a generative RNN model can generate much more coherent text [2, 43], but still falls short for larger types of content [50].

There are some obvious benefits to using a software-controlled RNN to generate fake reviews. First, it removes the cost of paying human writers, which costs \$1-\$10 per review on Yelp according to prior work [55]. To further obtain an independent estimate, we search Blackhat SEO Forum[1] for "Yelp reviews," and observe an average price of \$19.6 per review based on a random sample of 20 posts. Perhaps more importantly, software generators can control the specific timing and output of reviews, so they are harder to detect. When attackers launch large-scale fake review campaigns using human writers, workers tend to rapidly generate the requested reviews, producing a burst in new reviews that is easily detected [30, 73].

[1]https://www.blackhatworld.com/

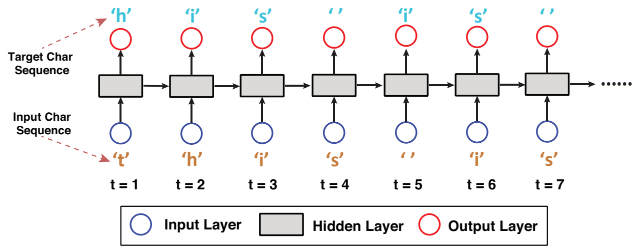


Figure 2: RNN generative model training.

2.4 RNN as a Text Generative Model

We provide background on text generation using Recurrent Neural Networks.

Neural Networks are computational models that use a connected network of *neurons* to solve machine learning tasks. Neurons serve as the basic computational units in a Neural Network. In this work, we focus on a specific class of Neural Networks known as Recurrent Neural Networks (RNNs), which are better suited for sequential data. RNNs can learn from a large corpus of natural language text (character or word sequences), to generate text at different levels of granularity, *i.e.* at the character level or word level. We focus on a character-level RNN due to its recent success on generating high quality text [14, 28]. Also, the memory and computational cost required to train a character-level RNN is lower than word-level RNN since number of words is significantly larger than number of valid characters in the English Language.

RNN Training. As mentioned before, traditional language models (*e.g.,* n-gram models) exhibit limited performance when trained on long text sequences since they are able to look back only a few steps of the sequence. RNN solves this problem by building a more sophisticated "memory" model which maintains long term information about what it has seen so far. In an RNN, "memory" is a set of high dimensional weights (hidden states) learned during the training stage to capture information about all characters seen in the training sequence.

Figure 2 illustrates the training process of an RNN-based text generation model. At each time step t, a new character $\mathrm{x_t}$ is fed as input to a memory unit of the RNN that maintains a hidden state $\mathrm{h_t}$, and provides an output $\mathrm{o_t}$. Then it compares the current output $\mathrm{o_t}$ with the desired output, which is the next character in the text. The error between them is computed and the hidden states are updated towards the direction where the error is minimized. After multiple iterations of updates, the hidden layer will eventually capture the relationship between each input character and all characters prior to it, *i.e.* the conditional probability distribution $P(\mathrm{x_{t+1}}|(\mathrm{x_1}, \ldots, \mathrm{x_t}))$.

Text Sampling. After an RNN model is trained, text can be generated by feeding a character, say $\tilde{\mathrm{x}_0}$, to the trained RNN. The RNN returns a probability distribution that defines which characters are likely to occur next, *i.e.* $P(\tilde{\mathrm{x}_1}|\tilde{\mathrm{x}_0})$. We stochastically sample from this distribution to obtain the next character $\tilde{\mathrm{x}_1}$. Next, by feeding

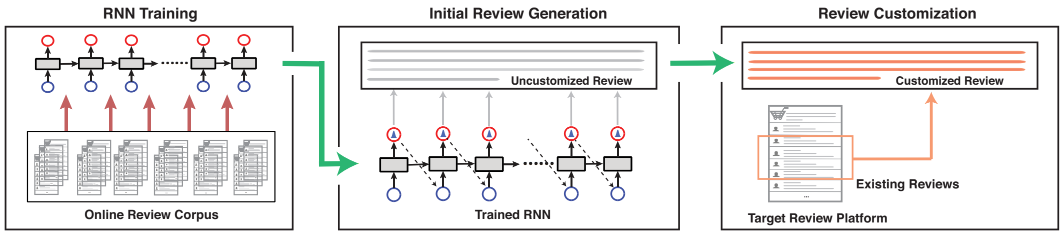


Figure 3: Overview of our attack methodology.

$\tilde{x_1}$ back in to the RNN, we obtain another probability distribution predicting the next character, *i.e.* $P(\tilde{x_2}|\tilde{x_0}, \tilde{x_1})$. This process can be repeated to generate continuous text $\tilde{x_0}, \tilde{x_1}, \tilde{x_2}, \ldots, \tilde{x_N}$.

Temperature Control. An important parameter that we can manipulate during the sampling stage is *temperature*. Temperature is a parameter used in the *softmax* function during the sampling stage when converting the output vector o_t to a probability distribution. Formally:

$$P(x_t|(x_1, \ldots, x_{t-1})) = \text{softmax}(o_t) \tag{1}$$

each o_t is a N-dimentional vector where N is the size of the character vocabulary. The *softmax* function is defined as:

$$P(\text{softmax}(o_t) = k) = \frac{e^{o_t^k/T}}{\sum_{j=1}^{N} e^{o_t^j/T}} \tag{2}$$

Here, o_t^k represents the component of the output corresponding to the character class k (in the vocabulary), at time t, for a given temperature, T.

Temperature controls the "novelty" of generated text. Temperatures lower than 1 amplifies the difference in the sampling probability for each character. In other words, this reduces the likelihood of the RNN to pick characters with lower probabilities, in preference of more common characters. As a result, this constrains the sampling, and generates less diverse text, and more potentially repetitive patterns. As temperature increases, the variation of sampling probability for each character diminishes, and the RNN will generate more "novel" and diverse text. But along with diversity comes a higher risk of mistakes (*e.g.,* spelling errors, context inconsistency errors *etc.*).

3 ATTACK METHODOLOGY AND SETUP

We focus our study on Yelp, the most popular site for collecting and sharing crowdsourcing user reviews. Yelp's review system is representative of other review systems, *e.g.,* Amazon or TripAdvisor. In this section, we describe details of our attack methodology, datasets and training setup.

3.1 Attack Methodology

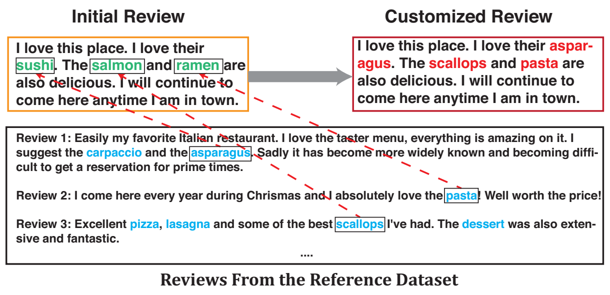
Our attack methodology is illustrated in Figure 3. At a high level, the attack consists of two main stages: (1) The first stage starts by training a generative language model on a review corpus. The language model is then used to generate a set of initial reviews. (2) In the second stage, a customization component further modifies these reviews to capture specific information about the target entity (*e.g.,* names of dishes in a seafood restaurant), and produces the final targeted fake review. In our experiments, the customizable content is extracted from a *reference dataset*, composed of existing reviews associated with the target entity. If there are no existing reviews, an attacker can build a reference dataset using reviews of entities in the same category (*e.g.,* seafood restaurants) as the target. Restaurant category metadata is available on Yelp and similar sites, and can be used to identify similar entities.

Generating Initial Reviews. First, the attacker chooses a training dataset that matches the domain of the target entity. For example, to generate reviews targeting restaurants, the attacker would choose a dataset of restaurant reviews. Next, the attacker trains a generative RNN model using the dataset. Afterwards, the attacker generates review text using the sampling procedure in Section 2.4. Note that the attacker is able to generate reviews at different *temperatures*.

Review Customization. In general, there is no control over the topic or context (*e.g.,* name of a food in a restaurant) generated from the RNN model, since the text is stochastically sampled based on the character distribution. To better target an entity (*e.g.,* restaurant), we further capture the context by customizing the generated reviews with domain-specific keywords. This is analogous to crowdsourced fake review markets, where workers are typically provided additional information about the target entity for a writing task [59]. The information consists of specific nouns (*e.g.,* names of dishes) to be included in the written review. Based on this observation, we propose an automated noun-level word replacement strategy.

Our method works by replacing specific words (nouns) in the initial review with new words that better capture the context of the target entity. This involves three main steps:

(1) *Choose the type of contextual information to be captured.* The attacker first chooses a keyword C that helps to identify the context. For example, if the attacker is targeting a restaurant, the keyword can be "food," which will capture the food-related context. If the target is an online electronic accessories store, then the keyword can be "accessory" or "electronics."
(2) *Identify words in reviews of the reference dataset that capture context.* Next, our method identifies all the nouns in the reference dataset that are relevant to the keyword C. Relevancy is estimated by calculating lexical similarity using WordNet [44], a widely used lexical database that groups English words into sets of synonyms and measures their concept relatedness [51]. We identify a set of words p in the reference dataset that have high lexical similarity with the keyword C, using a similarity



I love this place. I love their sushi. The salmon and ramen are also delicious. I will continue to come here anytime I am in town.

I love this place. I love their asparagus. The scallops and pasta are also delicious. I will continue to come here anytime I am in town.

Review 1: Easily my favorite Italian restaurant. I love the taster menu, everything is amazing on it. I suggest the carpaccio and the asparagus. Sadly it has become more widely known and becoming difficult to get a reservation for prime times.

Review 2: I come here every year during Chrismas and I absolutely love the pasta! Well worth the price!

Review 3: Excellent pizza, lasagna and some of the best scallops I've had. The dessert was also extensive and fantastic.

....

Figure 4: Example of review customization.

Dataset	# of Restaurants	# of Fake Reviews (%)	# of Real Reviews (%)
YelpBos [45]	1,028	28,151 (22.12%)	99,117
YelpSF [45]	3,466	90,777 (9.94%)	822,772
YelpZip [56]	4,204	84,484 (13.76%)	529,569
YelpNYC [56]	914	37,799 (10.48%)	322,858
YelpChi [48]	98	8,401 (12.83%)	57,061
Total	9,710	249,612 (11.99%)	1,831,377

Table 1: Summary of ground-truth dataset.

threshold MIN_{sim}. The set of words p captures the context of the target entity.

(3) *Identify words in initial reviews for replacement.* Finally, we find all the nouns in the review set R that are also relevant to C using the same method in Step 2. We replace them by stochastically sampling words in p based on the lexical similarity score.

A detailed version of the above algorithm is available in Appendix A. Figure 4 shows an example of customizing an initial review that has language more suitable for a Japanese restaurant, to a review more suitable for an Italian restaurant. The nouns to be replaced in the initial review are marked in green, and replacement nouns are marked in blue. Note that we choose this noun-level replacement strategy because of its simplicity, and there is scope for further improvement of this technique.

3.2 RNN Training and Text Generation

Training Process. For all experiments, we use a Long Short-Term Memory (LSTM) model [16], an RNN variant that has shown better performance in practice [22]. We examine multiple RNN training configurations used in prior work [14, 33] and determine the best configuration empirically through experiments. The neural network we used contains 2 hidden layers, each with 1,024 hidden units. For training, the input string is split into batches of size 256. Training loss is computed using *cross-entropy* [12], and weights are updated using *Adam* [26] optimization, a common optimization technique for neural network training. The model is trained for 20 epochs and the learning rate is set to be 2×10^{-3} and decays to half of the current rate every time when the loss increases for 5 successive batches. We also monitor training loss and inspect generated reviews to avoid underfitting or overfitting.

We pre-processed the review text by removing all extra white space and non-ASCII characters. Additionally, we separate the reviews in the corpus by the delimiter tokens "<SOR>" (start of review) and "<EOR>" (end of review), so the model also learns when to start and end a review by generating these two tokens. The RNN is trained using a machine with Intel Core i7 5930K CPU and a Nvidia TITAN X GPU. The training takes ~72 hours.

Text Generation. Once we have trained the language model, we can sample the review text at different temperatures. We generate reviews at 10 different temperatures between 0 and 1: [0.1, 0.2, ..., 1][2]. To start the text generation process, the model is seeded with the start of review delimiter token. Conversely, the model identifies the end of a review by generating the closing delimiter token.

For review customization, we chose the target keyword C to be "food," as a large number of nouns unsurprisingly relate to food. We set the other parameter MIN_{sim} to 0.2. After customization, overall, 98.4% of the reviews have at least one word replaced. The reviews not affected by the customization lack suitable content (or words) that capture the context, including reviews that rarely mention any food or dish in the text, *e.g.,* "I love this place! Will be back again!!"

Generated Text Samples. Table 2 shows several examples of reviews generated at different temperatures. At higher temperatures, the RNN is more likely to generate novel content, while at lower temperatures, the RNN produces repetitive patterns. We include more examples of generated reviews in Appendix B.

3.3 Datasets

For our evaluation, we use different datasets of restaurant reviews on Yelp. Each review in a dataset contains the text of the review, the identity of the target restaurant and a desired rating score, ranging from one to five stars. The rating score determines the sentiment of the review and the desired textual content.

In the rest of the paper, we present results based on the generation of reviews tailored towards a five-star rating. This considers the common scenario of an attack trying to improve the reputation of a restaurant. Our attack methodology is general and would be the same for other ratings as well. Appendix B presents examples of generated reviews tailored towards one-star and three-star ratings.

Three disjoint datasets of Yelp reviews are used for generating and evaluating the attack: a *training*, *ground-truth*, and *attack* dataset.

Training Datasets. We use the Yelp Challenge dataset to train the RNN language model, containing a total of 4.1M reviews by 1M reviewers, collectively targeting 144K businesses [79]. The dataset covers restaurants in 11 cities, spread across 4 countries. We extract reviews corresponding to different ratings, and found 617K reviews with a five-star rating from 27K restaurants. In total, these five-star reviews contain 57M words and 304M characters, a sufficiently large dataset for training an RNN.

[2]We do not experiment with temperatures beyond 1.0, because the sampling distribution would significantly diverge from the true distribution learned from the training corpus, and lead to overly diverse and incoherent text.

Ground-truth Dataset. This dataset, listed in Table 1, comprises of multiple Yelp review datasets released by researchers. By providing ground-truth information about existing fake and real reviews on Yelp, this dataset enables us to build machine learning fake review classifiers to evaluate our attack success (Section 4.1). Similar to previous work [45, 48, 56], we treat Yelp *filtered* and *unfiltered* reviews as ground-truth information for fake and real reviews. Yelp attempts to filter reviews that are "fake, shill or malicious" [78], but acknowledges imperfections in the accuracy of the filter [80]. However, given this is the best information currently available and used by many prior studies on fake reviews, we use it to establish ground-truth. In the rest of the paper, we use *fake* and *real* reviews to refer to Yelp filtered and Yelp unfiltered reviews, respectively.

For each dataset, we only consider reviews targeting restaurants[3]. The resulting dataset contains restaurants in NYC, Chicago, SF, Boston, and several cities in NJ, VT, CT, and PA.

Attack Dataset. This dataset contains the reviews generated by our RNN language model. We use the attack dataset to evaluate attack performance (Section 4) and defense schemes (Section 5).

The datasets contain similar data to the ground-truth dataset, except for replacing all fake reviews with our machine-generated reviews. Using our RNN model, we generate reviews targeting each restaurant in the ground-truth dataset using different temperatures. For each temperature, we generate as many reviews as fake reviews from Yelp for each restaurant, *i.e.* 249,612 machine-generated reviews targeting 9,710 restaurants.

4 EVALUATING QUALITY OF MACHINE-GENERATED REVIEWS

In this section, we evaluate the quality of machine-generated reviews along two dimensions. First, we investigate whether generated reviews can bypass detection by existing algorithmic approaches. Second, we conduct an end-to-end user study, by presenting restaurant reviews containing both generated reviews and real reviews to human judges. Our goal is to understand whether humans can distinguish generated reviews from real reviews.

4.1 Detection by Existing Algorithms

We focus on two popular algorithmic techniques to distinguish machine-generated reviews from real reviews: (1) a supervised ML scheme based on linguistic features, (2) a plagiarism detector to check for duplications between machine-generated reviews and training set (real) reviews.

ML-based Review Filter. Using machine learning classifiers to detect fake reviews is a well studied problem [20, 48, 53]. Most of the prior works rely on the observation that characteristics of fake reviews deviate from real reviews along many linguistic dimensions. We identified 5 groups of linguistic features, consisting of 77 features total that previously demonstrated strong discriminatory power for distinguishing fake and real reviews. We describe the features below:

- *Similarity feature (1)*: Captures inter-sentence similarity within a review at the word level. It is computed as the maximum cosine similarity between unigram features among all pairs of sentences [10, 20, 31, 56].
- *Structural features (4)*: Captures the structural aspects of a review. Individual features include the number of words, the number of sentences, the average sentence length (# of words) and the average word length (# of characters) [20, 56].
- *Syntactic features (6)*: Captures the linguistics properties of the review based on parts-of-speech (POS) tagging. Features include (distinct) percentages of nouns, verbs, adjectives and adverbs, first personal pronouns, and second personal pronouns [20, 31, 56].
- *Semantic features (4)*: Captures the subjectivity and sentiment of the reviews. Features include percentage of subjective words, percentage of objective words, percentage of positive words and percentage of negative words. All these features are defined in SentiWordNet [3], a popular lexical resource for opinion mining [31, 53, 56].
- *LIWC features (62)*: The Linguistic Inquiry and Word Count (LIWC) software [52] is a widely used text analysis tool in the social sciences. It categorizes ~4,500 keywords into ~68 psychological classes (*e.g.,* linguistic processes, psychological processes, personal concerns and spoken categories). We use the percentage of word count in each class as a feature, and exclude the features already included in the previous groups [48, 49].

We train a linear SVM classifier on the Yelp ground-truth dataset, composed of real reviews (Yelp unfiltered reviews), and fake reviews (Yelp filtered reviews). After training with all 77 linguistic features, we tested the performance of the classifier on the Yelp attack dataset, composed of real reviews and machine-generated reviews. We run 10-fold cross validation and report the average performance.

Evaluation of attack performance uses *precision* (percentage of reviews flagged by the classifier that are fake reviews), and *recall* (percentage of fake reviews flagged by the classifier). Figure 5 shows the precision and recall of the classifier when applied to machine-generated reviews generated from different temperatures (along with real reviews), with lower values indicating higher performing attacks. Overall, we observe high performing attacks at all temperatures. The best attack is at temperature 1.0, with a low precision of 18.48%, and a recall of 58.37%. Low precision indicates the inability of the ML classifier to distinguish between real reviews and machine-generated reviews.

In Figure 5, we observe that attack performance increases with temperature. To further understand this trend, we analyze how the linguistic features of the generated text vary as we increase temperature. In Figure 7, we compare the average value of a linguistic feature of generated reviews with real reviews at different temperatures. We show results for 3 linguistic features across 3 categories, and other features exhibit similar trends. In general, feature values of the machine-generated reviews diverge from real reviews at low temperatures, and converge as temperature increases, thus making it harder to distinguish them from real reviews at high temperatures.

Plagiarism Detector. Achieving reasonable linguistic quality does not rule out the possibility of being fake. A simple attack involves generating fake reviews by duplicating or partially copying from real reviews. In such cases, the review quality would be quite good,

[3]Since Yelp also includes reviews for non-restaurant business, *e.g.,* hair salon and car service.

Temperature	Generated Review Text
0.1	I love this place! I have been here a few times and have never been disappointed. The service is always great and the food is always great. The staff is always friendly and the food is always great. I will definitely be back and try some of their other food and service.
0.5	I love this place. I have been going here for years and it is a great place to hang out with friends and family. I love the food and service. I have never had a bad experience when I am there.
0.7	My family and I are huge fans of this place. The staff is super nice and the food is great. The chicken is very good and the garlic sauce is perfect. Ice cream topped with fruit is delicious too. Highly recommended!
1.0	I had the grilled veggie burger with fries!!!! Ohhhh and taste. Omgggg! Very flavorful! It was so delicious that I didn't spell it!!

Table 2: Example of the generated five-star fake reviews targeting at a buffet restaurant in NYC. More examples of generated reviews are included in the Appendix B.

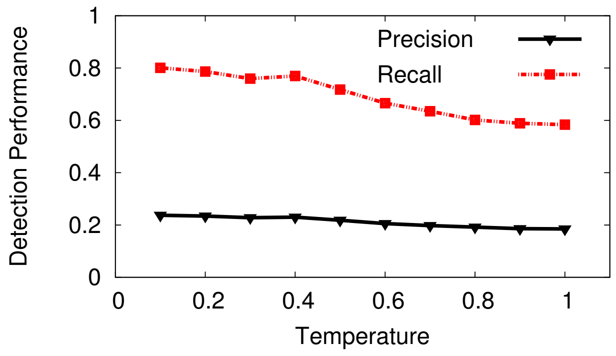


Figure 5: Performance of linguistic classifier on detecting machine-generated reviews.

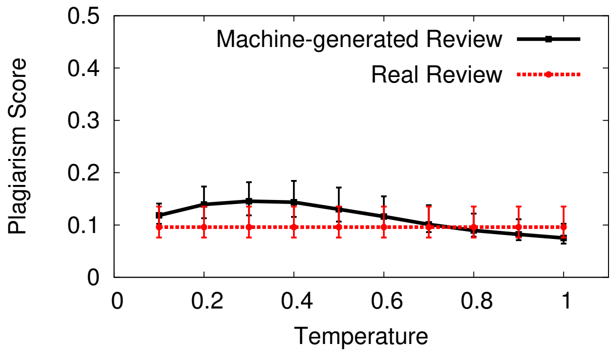
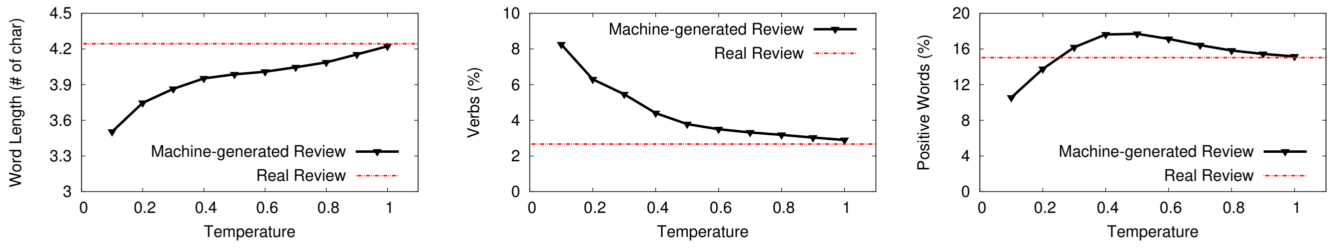


Figure 6: Plagiarism similarity score. Each point shows median, 25th and 75th percentile of the score distribution.



(a) Average word length (structural feature) (b) Ratio of verb usage (syntactic feature) (c) Ratio of positive word usage (semantic feature)

Figure 7: Change of linguistic feature values when temperature varies.

and would pass the linguistic filter. Standard solution is to rely on plagiarism checkers to identify the duplicate or near-duplicate reviews. Given that the RNN model is trained to generate text similar to the training set, we examine if the machine-generated reviews are duplicates or near-duplicates of reviews in the training set.

To conduct a plagiarism check, we assume that the service provider has access to a database of reviews used for training the RNN. Next, given a machine-generated review, the service provider runs a plagiarism check by comparing it with reviews in the database. This is a best case scenario for a plagiarism test, and helps us understand its potential to detect generated reviews.

We use Winnowing [63], a widely used method to identify duplicate or near-duplicate text. For a suspicious text, Winnowing first generates a set of fingerprints by applying a hashing function to a set of substrings in the text, and then compares the fingerprints between the suspicious text and the text in database. Similarity between two reviews is computed using *Jaccard Similarity* [5] of their fingerprints generated from Winnowing. The plagiarism similarity score for a single review is computed as the max similarity with all the other reviews in the dataset, and ranges from 0 to 1 (1 indicates identical reviews).

We pick a random sample of 10K machine-generated reviews for the plagiarism test, and the database (for comparison) includes the entire Yelp training dataset. Figure 6 shows the quantiles of similarity scores at different temperatures. Each point shows median, 25th and 75th percentile of the plagiarism score distribution. In addition, we also show the similarity score distribution for real reviews, which serves as a baseline for comparison. Note that scores for real reviews do not vary with temperature. We obverse that plagiarism scores of machine-generated reviews are low at all temperatures (lower score represents smaller probability of copying) and decrease as temperature increases. In addition, machine-generated reviews and real reviews show similar plagiarism scores, thus making them harder to distinguish. For example, at temperature 1.0, if we set a plagiarism score threshold such that 95% of real reviews are not flagged, we observe that 96% of machine-generated reviews still bypass the check. Thus, it remains hard to detect machine-generated reviews using a plagiarism checker without inadvertently flagging a large number of real reviews. This shows that the RNN does not simply copy the existing reviews from the training set.

4.2 Evaluation by User Study

Regardless of how well machine-generated reviews perform on statistical measures and tests, the real test is whether they can pass for real reviews when read by human users. In this section, we conduct an end-to-end user study to evaluate whether human examination can detect machine-generated reviews. In practice, service providers are known to involve human content moderators to separate machine-generated reviews from real reviews [69]. More importantly, these tests will tell us how convincing these reviews are to human readers, and whether they will accomplish their goals of manipulating user opinions.

User Study to Detect Machine-Generated Reviews. To measure human performance, we conduct surveys[4] on Amazon Mechanical Turk (AMT[5]). Each survey includes a restaurant name, description (explaining the restaurant category and description provided by the business on Yelp), and a set of reviews, which includes machine-generated reviews and real reviews written for that restaurant. We then ask each worker to mark reviews they consider to be fake, using any basis for their judgment.

For our survey, we choose 40 restaurants with the most number of reviews in our ground-truth dataset. For each restaurant, we generate surveys, each of which include 20 random reviews, out of which some portion (X) are machine-generated reviews, and the rest are real reviews from Yelp. The number X is randomly selected between 0 to 5 so that the expected ratio of fake reviews (12.5%) matches the real world setting (11.99% in Table 1). Additionally, we control the quality of real reviews shown in the surveys to cover the full range of usefulness. We leverage the *review usefulness* (a simple count of the number of users who found the review to be useful) metadata provided by Yelp for each review.

[4]Prior to conducting our study, we submitted a human subject protocol and received approval from our local IRB board.
[5]https://www.mturk.com/

For each of our 40 restaurants, we generated reviews using 5 different temperature parameters: [0.1, 0.3, 0.5, 0.7, 1.0]. We give each unique survey to 3 different workers, giving us a total of 600 surveys. Out of these 600 responses, we discarded 6 because they did not mark the gold standard reviews. Gold standard reviews are basically strings of random characters (*i.e.* meaningless text), that looks clearly fake to any worker. Lastly, we only request *master workers*[6] located in the US to guarantee English literacy. We show an example of our survey in the Figure 15(a) in Appendix C.

Figure 8 shows the human performance results as we vary the temperature. First, we observe that machine-generated reviews appear quite robust against a human test. Under the best configuration, the precision is only 40.6% with a recall of 16.2%. In addition, similar to algorithmic detection, attack performance improves as temperature increases. This is surprising, since we would expect that reviews at the extreme high or low temperature parameters would be easily flagged (either too repetitive or too many grammatical/spelling errors). We saw earlier that higher temperature produced reviews more statistically similar to real reviews, but expected errors to make those reviews detectable by humans. Instead, it seems that human users are much more sensitive to repetitive errors than they are to small spelling or grammar mistakes. We do observe that the best attack performance occurs at a high temperature of 0.7, which is marginally better than the performance at temperature of 1.0.

Helpfulness of Machine-Generated Reviews. Previously, we showed that humans tend to mark many machine-generated reviews as real. This raises a secondary question: *For machine-generated reviews that are not caught by humans, do they still have sufficient quality to be considered useful by a user?* Answering this question, takes us a step further towards generating highly deceptive fake reviews. We run a second round of AMT surveys to investigate this question.

In each survey, we first asked the workers to mark reviews as fake or real. Additionally, for the reviews marked as real, we asked for a rating of the usefulness of the review on a scale from 1 to 5 (1 as least useful, 5 as most useful). An example of the survey is shown in the Figure 15(b) in Appendix C. We conduct the survey using reviews generated at a temperature of 0.7, which gave the best performance from the previous round. Also, in this round, we test on 80 restaurants and hire 5 workers for each restaurant. The rest of the survey configuration remains the same as the first round.

We received all 400 responses and discarded 5 of them for failing the gold standard review. The average usefulness score of false negatives (unflagged machine-generated reviews) is close to that of true negatives (unflagged Yelp real reviews): machine-generated reviews have an average usefulness score of 3.15, which is close to the average usefulness score of 3.28 for real Yelp reviews. That is to say, workers think of unflagged machine generated reviews almost as useful as real reviews.

Overall, our experiments find machine-generated reviews very close to mimicking the quality of real reviews. Furthermore, the attacker is incentivized to generate reviews at high temperatures, as such reviews appear more effective at deceiving users.

[6]https://www.mturk.com/mturk/help?helpPage=worker#what_is_master_worker

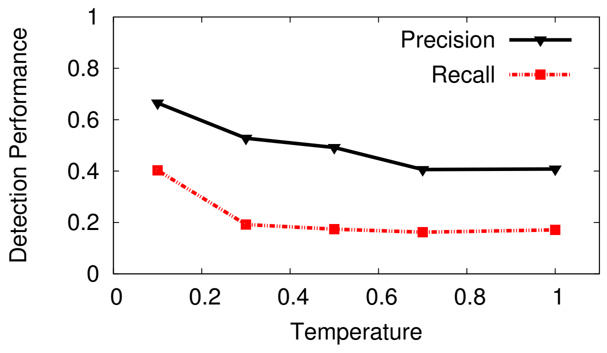


Figure 8: Performance of human judgment on detecting machine-generated review.

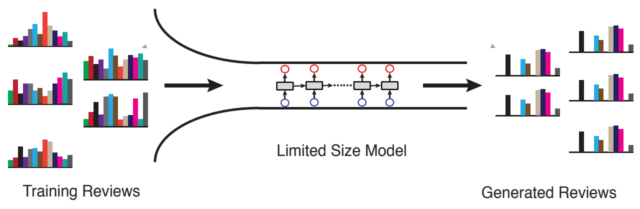


Figure 9: Key insight of our defense. During the training process, the language model builds a fixed memory representation of the large training corpus and its representativity is limited by the model size. The information loss incurred during the training would propagate to the generated text, leading to statistically detectable difference in the underlying character distribution between the generated text and human text.

5 DEFENDING AGAINST MACHINE-GENERATED REVIEWS

In this section, we propose a supervised learning scheme to detect machine-generated reviews with high accuracy.

Assumption. We assume that the service provider has access to a limited set of reviews generated by an attacker's language model and a set of real reviews available on the review site.

Why is defense challenging? Fundamentally, we are trying to develop a machine learning classifier capable of detecting the output of another ML model. In an ideal scenario, this seems impossible, because the generator model and detector are likely using the exactly same metrics on the same inputs (or training datasets). Thus, any metric that an ML detector is capable of detecting can be accounted for by the ML-based generator.

Key Insight. Figure 9 shows the key intuition behind our defense. While we expect that an attacker is aware of any metric used by the ML detector for detection, we can find a sufficiently "complex" metric where accurately capturing (and reproducing) the metric requires an extremely large RNN infrastructure that is beyond the means of most attackers. This leverages the fact that a generative language model builds a fixed memory representation of the entire training corpus, which limits the amount of information that can be learned from a training corpus. More specifically, we observe that text produced naturally (*e.g.,* by a human) diverges from machine generated text when we compare the *character level distribution*, even when higher level linguistic features (*e.g.,* syntactic, semantic features) might be similar. Such divergence in the character level distribution is primarily due to the information loss inherent in the machine-generation process, which is a function of the modeling power of the RNN used.

We note that the character level distribution metric is appropriate for our purposes, because it is the lowest level metric being modeled by our RNN, and therefore most likely to become a complexity bottleneck for the RNN. An attacker might consider an RNN generator that trains using word-level distributions. Fortunately (for our purposes), word-level distributions are more complex and difficult to model, both because the number of words is combinatorially larger than number of valid characters in the English language, and because such a model would need to add additional rules for valid punctuation. Therefore, an RNN generator targeting word-level distributions would be even more computationally constrained (thus incur more information loss), and intuitively, our defense would work at least as well as on a character-level RNN.

5.1 Proposed Methodology

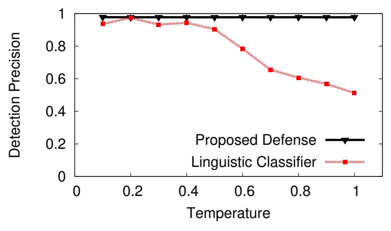
More concretely, consider the following attack scenario: The attacker trains on a set of human generated reviews R_T and builds a character-level RNN language model M, to generate a set of reviews R_F. Even when R_T is chosen in such a way that R_F becomes linguistically similar to the real reviews R_L on the site, we can statistically detect variations in the character level distribution between R_F and R_L.

Algorithm 1 provides details of the method. The service provider maintains access to the set of known machine-generated reviews R_F, along with the set of real reviews R_L, and aims to determine whether a given test review T is fake or real. Based on our insight, we expect the character-level distribution of reviews in set R_F to statistically diverge from that of reviews in set R_L.

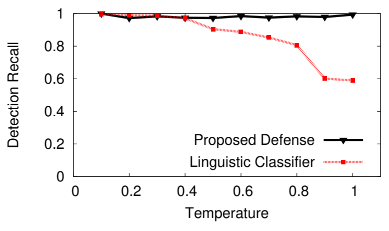
The character-level probability distribution $P(X_{t+1}=x_{t+1}|x_{1,\ldots,t})$, gives the probability of predicting the next character, given the sequence of preceding characters. To capture the divergence in the character distribution, the defender first builds an RNN language model RNN_F using the set of machine-generated reviews R_F and another language model RNN_L using R_L. Next, given a test review T, we feed the review, character by character, into each RNN model to obtain two character distributions, providing statistical representations of review T for each model. Finally, if the log-likelihood ratio of the test review's character distribution closely fits the model RNN_F, then we flag the review as fake.

5.2 Defense Evaluation

We evaluate our defense scheme along two directions. First, we study the detection performance of our scheme and how it compares to an ML scheme based on linguistic features (Section 4.1). Second, we investigate the robustness of our approach to evade detection against two attacker strategies.



(a) Precision



(b) Recall

Figure 10: Performance of proposed defense and linguistic classifier (Section 4.1).

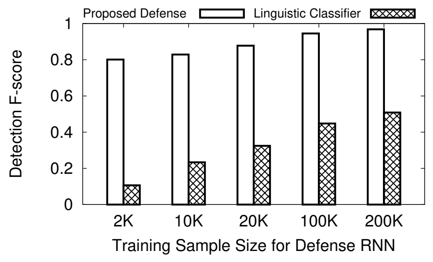


Figure 11: Detection performance given different sizes of defense training samples.

Algorithm 1 Proposed Defense

```
    ▷ input-$R_F$:machine-generated review training set, $R_L$:real review training set, T:test review
1:  procedure DEFENSE($R_F$, $R_L$, T)
2:      N ← length(T)
3:      $RNN_F$ ← trainRNN($R_F$)
4:      $RNN_L$ ← trainRNN($R_L$)
5:      for t = 1:N-1 do
6:          feed $X_t$ into $RNN_F$
7:          $\mathcal{L}_F \leftarrow P_F(X_{t+1}=x_{t+1}|x_{1,\ldots,t})$
8:          feed $X_t$ into $RNN_L$
9:          $\mathcal{L}_L \leftarrow P_L(X_{t+1}=x_{t+1}|x_{1,\ldots,t})$
10:         $\mathcal{L}_t \leftarrow -\log \frac{\mathcal{L}_L}{\mathcal{L}_F}$        ▷ negative log-likelihood ratio
11:     $\bar{\mathcal{L}} \leftarrow \frac{\sum_{i=1}^{N-1} \mathcal{L}_i}{N-1}$
12:     if $\bar{\mathcal{L}} > 0$ then
13:         return FAKE
14:     else
15:         return REAL
```

We follow a standard RNN model training process to train RNN_F and RNN_L. We refer to them as *defense RNN*. Unless otherwise stated, we report performance on 2,000 test reviews (balanced set of real and machine-generated reviews) in the remainder of this section.

Detection Performance. To first understand the performance in a potential best case scenario, we evaluate our scheme by considering a large amount of ground-truth information. Our ground-truth set consists of 120K machine-generated reviews, from our Yelp attack dataset (Section 3.3), and 120K additional real reviews. We set the model configuration for the defense RNN to be the following: 1,024 hidden units, 2 hidden layers, batch size of 128 and 20 training epochs.

As a baseline for comparison, we compute detection performance using the ML scheme described in Section 4.1, which we refer to as the *linguistic classifier*. Note that the ML scheme is based on high level linguistic features and trained using the same ground-truth set of machine-generated and real reviews. We expect the linguistic classifier to perform better than the results in Section 4.1 because the training data now includes machine-generated reviews that we aim to identify directly.

Figure 10 shows the detection performance when we train and test on text generated at different temperatures. Our approach achieves high precision and recall at all temperatures, *i.e.* over 0.98 precision and 0.97 recall. Additionally, we outperform the linguistic classifier at most temperatures, and the gap between the two schemes increases at higher temperatures (*e.g.,* temperature > 0.6). At temperature 1.0, our scheme achieves an *F-score* (the harmonic mean of precision and recall) of 0.98, while the linguistic approach only achieves an F-score of 0.55. Interestingly, the linguistic classifier shows high detection performance at low temperatures. This trend can be explained by our earlier finding that linguistic features diverge more from the real reviews at low temperatures (Figure 7).

Next, we study performance when we limit the amount of ground-truth used for training and focus on text generated at temperature 1.0. Figure 11 shows performance when training data size varies

Training Samples	Hidden Unit Size	Layer Size	Batch Size	Training Epoch
2K	128	1	16	50
10K	256	1	32	50
20K	512	1	56	30
100K	768	2	128	20
200K	1,024	2	128	20

Table 3: Training configurations of defense models when defense training sample size varies.

Hidden Unit Size	Training Samples	Layer Size	Batch Size	Training Epoch
128	10K	1	32	50
256	50K	1	56	50
512	100K	1	128	30
768	500K	2	256	20
1,024	617K	2	256	20
2,048	617K	2	256	50

Table 4: Training configurations of attack models when attack model size varies.

from 2,000 to 200K samples. Each training dataset is a balanced dataset of machine-generated and real reviews. The RNN model configuration used for defense for each training set size is detailed in Table 3. Our scheme significantly outperforms the linguistic classifier for all datasets and achieves a high F-score of 0.80 using only 1,000 machine-generated reviews (2,000 training dataset). Considering the fact that service providers have taken considerable effort to build large fake review datasets (∼250K fake reviews in Table 1), 1,000 reviews is a relatively small sample of known fake reviews. Thus, unlike the linguistic classifier, our defense scheme performs well with highly limited ground-truth information.

Evading Detection by Increasing Attack Model Quality. The attacker can evade detection by improving the quality of the RNN generative model or increasing the memory size of the model. A higher quality model would generate more "natural" text, and thus reduce the divergence in the character distribution. However, this strategy comes with a higher cost for the attack. Training a larger RNN model requires more training data to be collected, more computational resources (*e.g.,* GPUs with more memory and computational capacity), while also imposing additional training time. It is hard to quantify the increase in attack cost when accounting for all these factors. Instead, we focus on the impact on training time as the attacker varies model quality to evade detection.

We vary the attack model size (number of hidden units) from 128 to 2,048. We also vary other model parameters, and the size of the training dataset to avoid underfitting or overfitting. Details of the attack models are described in Table 4. For the defense RNN, we use a configuration based on 2K training samples in Table 3.

Figure 12 shows the tradeoff between decreases in detection performance (F-score) and increases in training time for the attacker. In general, when the attacker trains a larger model, our defense performance would degrade: when doubling the model size from 128 to 256 cells, detection performance drops by 3.95% with training time increasing by 71.86%. As the model size grows further, attacker's gain in evasion rate slows, while the matching training time accelerates significantly. This is due to the increase in computational complexity: from model size 1,024 to 2,048, defense performance only decreases by 2.70% but the attacker's training time raises by 435.1%.

In practice, larger models would require a significantly larger training set as well. For example, Jozefowicz *et al.* [21] trained a 2-layer RNN with $8,192$ hidden units on a dataset with $\sim 0.8B$ words, 14x larger than our training dataset, to achieve state-of-the-art model performance. Therefore, the computational cost and amount of training data required to train a larger model would become prohibitively expensive for all but the most resourceful attackers.

Next, we show there are other ways to further diminish the power of any resource-based countermeasures by the attacker. We observe that our defense scheme performs better on longer reviews, as the scheme has more data to capture divergence in the character distribution. Based on this observation, we propose a simple policy for the service provider to further raise the bar for evasion: set a minimum review length. Figure 13 shows how detection performance varies as we increase the minimum review length requirement against an attack model with $1,024$ hidden units. Note that the average review length on Yelp Training Data (Section 3.3) is 483 characters. When we increase the minimum length requirement to 300 characters (*i.e.* still below the average), F-score increases from 0.80 to 0.86, which nullifies the attack success gained by increasing model size from 512 to 1,024 hidden units (from Figure 12). This means that the attacker now must aim for training a significantly larger attack model, at the expense of increased training cost, to overcome the reduction in attack success.

Evading Detection by Generating Reviews at Different Temperatures. When reviews from a language model are detected as fake, one would expect the attacker to build a new language model (thus raising the cost) for the next attack. Instead, the attacker can try to evade further detection by generating new reviews using the existing model, by changing the temperature parameter (without re-training). While our scheme can detect reviews at different temperatures when we have ground-truth information at those temperatures (Figure 10), performance remains unclear when ground-truth information is unavailable. Put differently: *Using a defense scheme trained on reviews at a specific temperature, can we detect reviews generated at other temperatures?* If this is possible, it would allow the service provider to defend against such attacks without having to collect new ground-truth information.

We investigate the above scenario in Figure 14, where we train the linguistic classifier and our scheme at a specific temperature (T_{train}) and evaluate detection performance at all other temperatures (T_{test}). Both T_{train} and T_{test} are from 0.1 to 1. The training configuration is same as the one used for 2K training samples in Table 3.

As expected, the linguistic classifier exhibits poor defensive power at higher temperatures. At low temperatures, we see that a defense trained at a given temperature maintains effectiveness at temperatures in the "neighborhood" of that region, possibly due to similar linguistic characteristics in the temperature neighborhood. However, the performance drops quickly when T_{train} and T_{test} are far apart.

On the other hand, our approach shows an interesting trend. Unlike the linguistic classifier, our performance maintains robustness whenever $T_{train} > T_{test}$. This is because a review generated at a high temperature would include both infrequent and frequent patterns from the training sequence. As a result, a defense trained at a higher temperature can capture the frequent sequences present in the character distribution of a review at a lower temperature. Hence, our defense scheme maintains high performance even when T_{train} and T_{test} are distant as long as $T_{train} \geq T_{test}$. It should be noted that the attacker has an incentive to generate reviews at high temperatures because they are more likely to deceive users (Section 4.2). As such, the service provider can likely obtain some initial ground-truth information about reviews at high temperatures, and thus build a robust defense.

6 RELATED WORK

Text Generation. There are a number of natural language generation techniques based on pre-defined templates [57, 58, 70]. These systems usually require some domain knowledge and well-designed rules. Recently, learning-based approaches became popular, *i.e.* building a statistical model to learn the language information from a large corpus. In these cases, the quality of the generated text

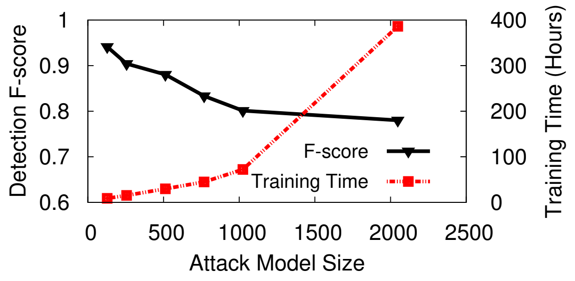


Figure 12: Detection performance against attacks generated by different models and their training costs.

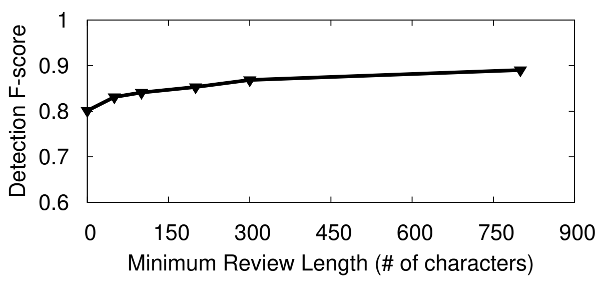
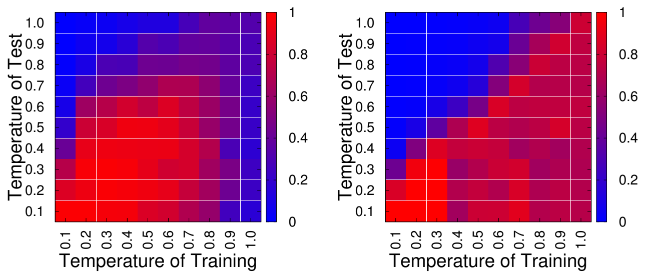


Figure 13: Detection performance when the minimum review length is restricted.



(a) Linguistic classifier (b) Our method

Figure 14: Detection performance (F-score) of training and then applying at different temperatures.

highly correlates with the model quality. Previous work shows that well-trained RNN models outperform simpler language models like N-gram [21] and RNN-based language models have appeared as a promising approach to generate text [14, 67]. Researchers have also achieved successful results in generating text for different domains, including email responses [23], image description [24], movie dialogues [64] and online social network conversations [65].

Our work is closest to studies by Lipton *et al.* [33], Hovy [17] and Lappas [27]. Lipton *et al.* [33] also studies product review generation using an RNN, but does not consider an adversarial setting. Hovy [17] performed a preliminary investigation of n-gram-based review generation in an adversarial setting. Unlike our research, they do not consider more sophisticated generative models (RNN) to evade detection, and do not consider any robust defenses or countermeasures. Lastly, Lappas [27] analyzes fake reviews from the attacker's perspective to determine the factors that enable a successful attack, but does not consider automatic generation of reviews.

Neural Networks and Security. Most prior studies on applying Neural Networks focus on improving existing defenses against various network and web security vulnerabilities. Work in this category mainly includes proposals to improve network intrusion detection [8, 25], malware classification [62, 66, 81], and password-based authentication systems [39].

Similar to our work, a few studies also investigate the feasibility of attacking online systems using Deep Neural Networks. This includes proposals to automatically solve CAPTCHAs using Convolutional Neural Networks [13, 60], and automatically generate malware domains using Generative Adversarial Networks [1]. To the best of our knowledge, our work is the first to explore attacks on online review systems using Deep Neural Networks.

Crowdturfing and Review Spam Detection. Prior work characterized crowdturfing marketplaces that supply human labor to enable attacks on a variety of online platforms, such as review systems, social media, and search engines [29, 73, 74]. In addition, work by Wang *et al.* investigates detection of malicious crowdworkers using machine learning and explores the robustness of machine learning classifiers against evasive tactics by crowdworkers. Furthermore, researchers have extensively studied detection of opinion spam or fake reviews in online review systems using features based on review content [32, 48, 53] and a variety of metadata [10, 19, 20]. We differ from these studies by focusing on automatically generating fake reviews that can evade detection by advanced machine learning classifiers and human investigation.

7 DISCUSSION & CONCLUSION

In this work, we focus on the potential for misuse of deep learning models in the context of attacking online review platforms. Our work shows how RNNs can generate deceptive yet realistic looking reviews targeting restaurants on Yelp. An extensive evaluation of the quality of generated reviews indicates the difficulty in detecting such reviews using existing algorithmic approaches, and even by human examination (which serves as an end-to-end test of our attack).

We propose a novel approach to defend against RNN-based fake reviews, by leveraging a fundamental limitation of an RNN-based model: *information loss incurred during the training process when fitting a large training dataset to a fixed size statistical model.* Due to the information loss, generated reviews diverge from real reviews when comparing their character-level distribution, even when higher level linguistic characteristics are preserved. Our scheme, based on supervised learning, can detect machine-generated reviews with high accuracy (F-score ranging from 0.8 to 0.98 depending on the amount of available ground-truth) and outperforms existing ML-based fake review filters.

Future Work. In terms of potential future work, one direction is to consider the role that user and content metadata can play in both the attack and defense perspectives. Metadata can be crucial in terms of deceiving users (*e.g.,* by increasing the number of friends/contacts on the site) and in assisting defenses [10, 19, 20, 31, 47, 71, 75] (*e.g.,* by analyzing the patterns in timestamps of user activites). Orchestrating the general behavior of user accounts using deep learning to bypass metadata based defenses could be an interesting research challenge. Second, while we limit ourselves to the domain of online review systems and fake review attacks, deep learning-based generative text models can be applied to launch attacks in other scenarios as well. We highlight two of these possible application scenarios.

Strengthening Sybil Attacks. Attackers can use our techniques to generate realistic looking text-based user behavior patterns [4], *e.g.,* posting, commenting and messaging. This can help attackers make Sybil (fake) accounts indistinguishable from legitimate accounts based on textual content. A special case of this involves launching an impersonation attack in online social networks [11].

Fake News Generation. Identifying fake news, *i.e.* "a made-up story with an intention to deceive" [61], currently remains an open challenge [9]. The research community has started to explore the possibility of automating the detection process by building an AI-assisted fact-checking pipeline [41, 72, 76]. We believe that AI can not only assist fake news detection but also generate fake news. Given the availability of large-scale news datasets [68], an attacker can potentially generate realistic looking news articles using a deep-learning approach (RNN). And due to its low economic cost, the attacker can pollute social media newsfeeds with a large number of fake articles.

We hope our results will bring more attention to the problem of malicious attacks based on deep learning language models, particularly in the context of fake content on online services, and encourage the exploration and development of new defenses.

A REVIEW CUSTOMIZATION DETAILS

We show the details of review customization process (Section 3.1) in Algorithm 2.

B MORE EXAMPLES OF THE GENERATED FAKE REVIEWS

We present examples of generated reviews targeting a specific restaurant (description shown below). The samples are generated under different temperatures and from the models trained using five-star, three-star and one-star Yelp training review corpus.

Target Restaurant Description

Experience MKT, a market-to-table San Francisco restaurant and bar. Taste your way through ultra-creative handcrafted cocktails, small plates for sharing, and steaks from the finest California ranches. Meet friends and relax in our retro urban-chic atmosphere. In the heart of SoMa - with panoramic fifth-floor views of Market Street and downtown - this is one of San Francisco's best restaurant venues for social drinks and dining.

B.1 Machine-generated One-Star Reviews

Generated with *temperature* 0.1

I was so excited to try this place out for the first time and the food was awful. I ordered the chicken sandwich and it was so salty that I could not eat it. I was so disappointed that I was so disappointed in the food. I was so disappointed that I was so disappointed with the service.

Generated with *temperature* 0.3

I had a drink at this restaurant. I was there for a couple of months. The restaurant was clean but the food was not good. The service was very slow and the service is so bad. I would not recommend this place to anyone.

Generated with *temperature* 0.5

The worst place I've ever been too. I waited for a while and the place was not available and the service was absolutely terrible. The server was rude and not apologetic and unprofessional. I will not be going back to this location.

Generated with *temperature* 0.7

I have been here twice, very expensive for what you get. We stayed here before at our table and were seated in a line for 15 minutes and the restaurant was full and the cashier recommended the drinks and they forgot the cookie cover charge and we were told it was not an issue. I would never recommend this place to anyone.

Generated with *temperature* 1.0

DO NOT WASTE YOUR TIME AND MONEY! The absolute worst service I have ever experienced. This place is a joke. The waitress was rude and said she would put the manager to come out but never happened. I wish I could give zero star.

B.2 Machine-generated Three-Star Reviews

Generated with *temperature* 0.1

The food was good but not great. I had the lasagne and the sauce was a little too sweet for my taste. The salad was good but the sauce was a little too sweet for my taste. The sauce was a little too salty for my tastes. The sauce was a little too sweet for my taste.

Generated with *temperature* 0.3

I've had the fried chicken and the chicken breast with chicken soup with sweet potato fries. I don't know what a lot of the meat was on the salty side as well. The sauce was a bit salty and didn't have much flavor. The salad was tasty and well presented but the bread was super tasty.

Generated with *temperature* 0.5

I was here for a weekend brunch and the food was ok. I love the pizza that is a chain restaurant. I think the service is excellent. I had the spaghetti and they were very good and the hot dog was good. I got the red velvet chocolate cake special which was very good but the service was a little slow. The food was good, but not up to par with other places nearby.

Generated with *temperature* 0.7

Algorithm 2 Review Customization

```
   ▷ input: R-initial review, T-reference review set, C-topic keyword, $MIN_{sim}$-similarity threshold
1: procedure REVIEW CUSTOMIZATION(R, T, C, $MIN_{sim}$)
   ▷ find nouns in R close to C
2:     P ← ∅
3:     for t ∈ T do
4:         for n ∈ Nouns(t) do
5:             if similarity(n, C) > $MIN_{sim}$ then
6:                 P← P ⋃ n
   ▷ find nouns in R also close to C, replace by sampling from P
7:     for n ∈ Nouns(R) do
8:         if similarity(n, C) > $MIN_{sim}$ then
9:             S ← ∅
10:            for p ∈ P do
11:                S ← S ⋃ similarity(n, p)
12:            z ← softmax(S)
13:            $n^*$ ← sample from P based on z
14:            replace n by $n^*$
15:    return R
```

The food wasn't bad. The cupcakes are okay and the service is excellent but the prices are a bit high. I do like the fresh made salad and drink specials. I would recommend this place for a place to grab a bite for a couple of times.

Generated with *temperature* 1.0

Came here for lunch today and the place was pretty empty. The steak was good but the chicken they had a little less oily and overcooked. I would recommend this place if you are looking for a cheap place to stop by.

B.3 Machine-generated Five-Star Reviews

Generated with *temperature* 0.1

I have been going to this place for a few years now and I have never had a bad experience. The service is great! They are always so friendly and helpful. I will definitely be back and I will be back for sure!

Generated with *temperature* 0.3

This place is amazing! The bartenders are absolutely amazing. The pasta is delicious and I love their pastries and it is amazing. I love the breakfast, friendly staff and the price is very reasonable. I have never had a bad experience here. I will be back for sure!

Generated with *temperature* 0.5

I love this place. I went with my brother and we had the vegetarian pasta and it was delicious. The beer was good and the service was amazing. I would definitely recommend this place to anyone looking for a great place to go for a great breakfast and a small spot with a great deal.

Generated with *temperature* 0.7

I have been a customer for about a year and a half and I have nothing but great things to say about this place. I always get the pizza, but the Italian beef was also good and I was impressed. The service was outstanding. The best service I have ever had. Highly recommended.

Generated with *temperature* 1.0

The food here is freaking amazing, the portions are giant. The cheese bagel was cooked to perfection and well prepared, fresh & delicious! The service was fast. Our favorite spot for sure! We will be back!

C SCREENSHOTS OF USER STUDY SURVEYS

C.1 Fake/Real Review Detection

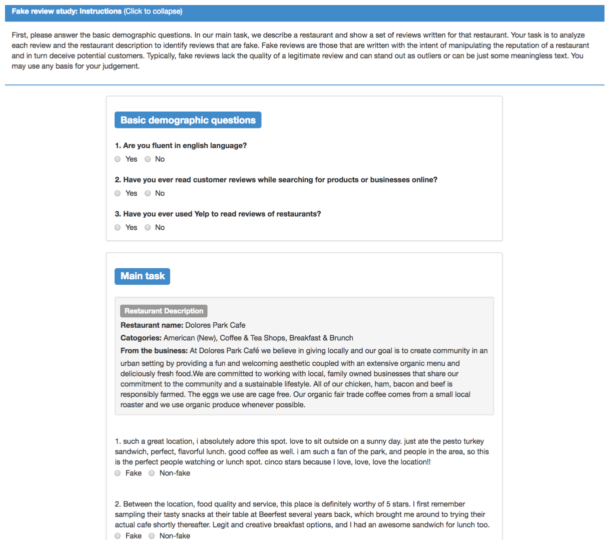
Figure 15(a) shows a screenshot of the survey designed for examining human performance on machine-generated review detection.

C.2 Review Helpfulness Rating

Figure 15(b) shows another screenshot of the second round survey designed for collecting the helpfulness score of the machine-generated reviews.

ACKNOWLEDGMENTS

We wish to thank our anonymous reviewers for their constructive feedback, and William Yang Wang for insightful discussions. This project was supported by NSF grants CNS-1527939 and CNS-1705042. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of any funding agencies.



Fake review study: Instructions (Click to collapse)

First, please answer the basic demographic questions. In our main task, we describe a restaurant and show a set of reviews written for that restaurant. Your task is to analyze each review and the restaurant description to identify reviews that are fake. Fake reviews are those that are written with the intent of manipulating the reputation of a restaurant and in turn deceive potential customers. Typically, fake reviews lack the quality of a legitimate review and can stand out as outliers or can be just some meaningless text. You may use any basis for your judgement.

Basic demographic questions

1. Are you fluent in english language?
Yes No

2. Have you ever read customer reviews while searching for products or businesses online?
Yes No

3. Have you ever used Yelp to read reviews of restaurants?
Yes No

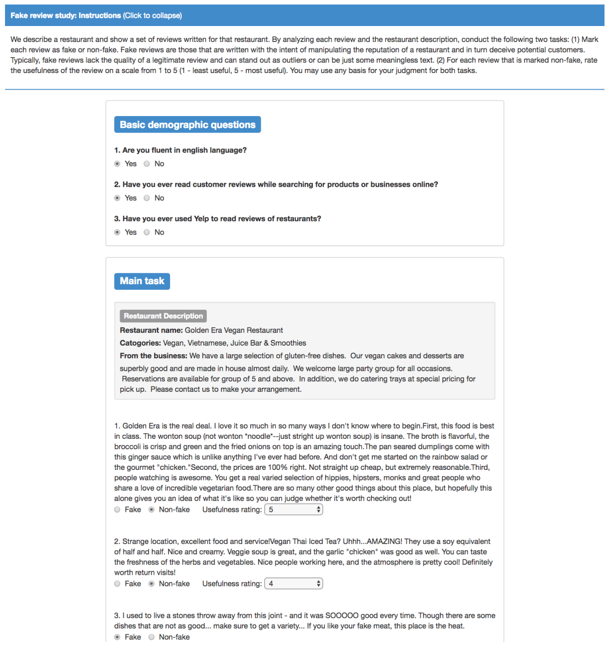
Main task

Restaurant Description
Restaurant name: Dolores Park Cafe
Catogories: American (New), Coffee & Tea Shops, Breakfast & Brunch
From the business: At Dolores Park Café we believe in giving locally and our goal is to create community in an urban setting by providing a fun and welcoming aesthetic coupled with an extensive organic menu and deliciously fresh food.We are committed to working with local, family owned businesses that share our commitment to the community and a sustainable lifestyle. All of our chicken, ham, bacon and beef is responsibly farmed. The eggs we use are cage free. Our organic fair trade coffee comes from a small local roaster and we use organic produce whenever possible.

1. such a great location, i absolutely adore this spot. love to sit outside on a sunny day. just ate the pesto turkey sandwich, perfect, flavorful lunch. good coffee as well. i am such a fan of the park, and people in the area, so this is the perfect people watching or lunch spot. cinco stars because I love, love, love the location!!
Fake Non-fake

2. Between the location, food quality and service, this place is definitely worthy of 5 stars. I first remember sampling their tasty snacks at their table at Beerfest several years back, which brought me around to trying their actual cafe shortly thereafter. Legit and creative breakfast options, and I had an awesome sandwich for lunch too.
Fake Non-fake

(a) Examining human performance on machine-generated review detection.



Fake review study: Instructions (Click to collapse)

We describe a restaurant and show a set of reviews written for that restaurant. By analyzing each review and the restaurant description, conduct the following two tasks: (1) Mark each review as fake or non-fake. Fake reviews are those that are written with the intent of manipulating the reputation of a restaurant and in turn deceive potential customers. Typically, fake reviews lack the quality of a legitimate review and can stand out as outliers or can be just some meaningless text. (2) For each review that is marked non-fake, rate the usefulness of the review on a scale from 1 to 5 (1 - least useful, 5 - most useful). You may use any basis for your judgment for both tasks.

Basic demographic questions

1. Are you fluent in english language?
[✓ Yes] No

2. Have you ever read customer reviews while searching for products or businesses online?
[✓ Yes] No

3. Have you ever used Yelp to read reviews of restaurants?
[✓ Yes] No

Main task

Restaurant Description
Restaurant name: Golden Era Vegan Restaurant
Catogories: Vegan, Vietnamese, Juice Bar & Smoothies
From the business: We have a large selection of gluten-free dishes. Our vegan cakes and desserts are superbly good and are made in house almost daily. We welcome large party group for all occasions. Reservations are available for group of 5 and above. In addition, we do catering trays at special pricing for pick up. Please contact us to make your arrangement.

1. Golden Era is the real deal. I love it so much in so many ways I don't know where to begin.First, this food is best in class. The wonton soup (not wonton *noodle*--just stright up wonton soup) is insane. The broth is flavorful, the broccoli is crisp and green and the fried onions on top is an amazing touch.The pan seared dumplings come with this ginger sauce which is unlike anything I've ever had before. And don't get me started on the rainbow salad or the gourmet "chicken."Second, the prices are 100% right. Not straight up cheap, but extremely reasonable.Third, people watching is awesome. You get a real varied selection of hippies, hipsters, monks and great people who share a love of incredible vegetarian food.There are so many other good things about this place, but hopefully this alone gives you an idea of what it's like so you can judge whether it's worth checking out!
Fake [✓ Non-fake] Usefulness rating: 5

2. Strange location, excellent food and service!Vegan Thai Iced Tea? Uhhh...AMAZING! They use a soy equivalent of half and half. Nice and creamy. Veggie soup is great, and the garlic "chicken" was good as well. You can taste the freshness of the herbs and vegetables. Nice people working here, and the atmosphere is pretty cool! Definitely worth return visits!
Fake [✓ Non-fake] Usefulness rating: 4

3. I used to live a stones throw away from this joint - and it was SOOOOO good every time. Though there are some dishes that are not as good... make sure to get a variety... If you like your fake meat, this place is the heat.
[✓ Fake] Non-fake

(b) Collecting helpfulness rating of the machine-generated reviews.

Figure 15: Screenshots of the survey designed for end-to-end user study.

REFERENCES

[1] Hyrum S. Anderson, Jonathan Woodbridge, and Bobby Filar. 2016. DeepDGA: Adversarially-tuned domain generation and detection. In *Proc. of AISec Workshop*.
[2] Ebru Arisoy, Tara N. Sainath, Brian Kingsbury, and Bhuvana Ramabhadran. 2012. Deep neural network language models. In *Proc. NAACL-HLT Workshop*.
[3] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Proc. of LREC*.
[4] Yazan Boshmaf, Ildar Muslukhov, Konstantin Beznosov, and Matei Ripeanu. 2012. Key challenges in defending against malicious socialbots. In *Proc. of LEET Workshop*.
[5] Andrei Z. Broder, Steven C. Glassman, Mark S. Manasse, and Geoffrey Zweig. 1997. Syntactic clustering of the web. In *Proc. of WWW*.
[6] CNBC 2014. TripAdvisor fined $600,000 for fake reviews. https://www.cnbc.com/2014/12/23/tripadvisor-fined-600000-for-fake-reviews.html. (2014).
[7] Cynthia Boris 2013. Study shows an increase in local competition encourages negative review fraud on Yelp. http://www.marketingpilgrim.com/2013/11/study-shows-an-increase-in-local-competition-encourages-negative-review-fraud-on-yelp.html. (2013).
[8] Bo Dong and Xue Wang. 2016. Comparison deep learning method to traditional methods using for network intrusion detection. In *Proc. of ICCSN*.
[9] Fake news challenge 2017. http://www.fakenewschallenge.org. (2017).
[10] Geli Fei, Arjun Mukherjee, Bing Liu, Meichun Hsu, Malu Castellanos, and Riddhiman Ghosh. 2013. Exploiting burstiness in reviews for review spammer detection. *Proc. of ICWSM*.

[11] Oana Goga, Giridhari Venkatadri, and Krishna P. Gummadi. 2015. The doppelgänger bot attack: Exploring identity impersonation in online social networks. In *Proc. of IMC.*

[12] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep learning.* MIT press.

[13] Ian J. Goodfellow, Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, and Vinay Shet. 2013. Multi-digit number recognition from street view imagery using deep convolutional neural networks. *arXiv:1312.6082* (2013).

[14] Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv:1308.0850* (2013).

[15] F. Maxwell Harper and Joseph A. Konstan. 2016. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems* 5, 4 (2016), 19.

[16] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[17] Dirk Hovy. 2016. The enemy in your own camp: How well can we detect statistically-generated fake reviews–An adversarial study. In *Proc. of ACL (Short Paper).*

[18] Jeff Roberts 2015. Amazon sues people who charge $5 for fake reviews. http://fortune.com/2015/10/19/amazon-fake-reviews. (2015).

[19] Nitin Jindal and Bing Liu. 2007. Review spam detection. In *Proc. of WWW.*

[20] Nitin Jindal and Bing Liu. 2008. Opinion spam and analysis. In *Proc. of WSDM.*

[21] Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noam Shazeer, and Yonghui Wu. 2016. Exploring the limits of language modeling. *arXiv:1602.02410* (2016).

[22] Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. 2015. An empirical exploration of recurrent network architectures. In *Proc. of ICML.*

[23] Anjuli Kannan, Karol Kurach, Sujith Ravi, Tobias Kaufmann, Andrew Tomkins, Balint Miklos, Greg Corrado, László Lukács, Marina Ganea, Peter Young, and others. 2016. Smart reply: Automated response suggestion for email. In *Proc. of KDD.*

[24] Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proc. of CVPR.*

[25] Gyuwan Kim, Hayoon Yi, Jangho Lee, Yunheung Paek, and Sungroh Yoon. 2017. LSTM-based system-call language modeling and robust ensemble method for designing host-based intrusion detection systems. In *Proc. of ICLR.*

[26] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv:1412.6980* (2014).

[27] Theodoros Lappas. 2012. Fake reviews: The malicious perspective. In *Proc. of NLDB.*

[28] Rémi Lebret, David Grangier, and Michael Auli. 2016. Neural text generation from structured data with application to the biography domain. In *Proc. of EMNLP.*

[29] Kyumin Lee, Prithivi Tamilarasan, and James Caverlee. 2013. Crowdturfers, campaigns, and social media: Tracking and revealing crowdsourced manipulation of social media. In *Proc. of ICWSM.*

[30] Kyumin Lee, Steve Webb, and Hancheng Ge. 2014. The dark side of micro-task marketplaces: Characterizing fiverr and automatically detecting crowdturfing. In *Proc. of ICWSM.*

[31] Fangtao Li, Minlie Huang, Yi Yang, and Xiaoyan Zhu. 2011. Learning to identify review spam. In *Proc. of IJCAI.*

[32] Jiwei Li, Myle Ott, Claire Cardie, and Eduard H. Hovy. 2014. Towards a general rule for identifying deceptive opinion spam. In *Proc. of ACL.*

[33] Zachary C. Lipton, Sharad Vikram, and Julian McAuley. 2015. Capturing meaning in product reviews with character-Level generative text models. *arXiv:1511.03683* (2015).

[34] Michael Luca and Georgios Zervas. 2016. Fake it till you make it: Reputation, competition, and Yelp review fraud. *Management Science* 62, 12 (2016), 3412–3427.

[35] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proc. of NAACL-HLT.*

[36] Marco Chiappetta 2016. Amazon boosts cloud-computing performance with new, GPU-accelerated AWS instances. http://www.forbes.com/sites/marcochiappetta/2016/09/30/amazon-boosts-cloud-computing-performance-with-new-gpu-accelerated-aws-instances. (2016).

[37] Mary Pilon 2009. A fake Amazon reviewer confesses. http://blogs.wsj.com/wallet/2009/07/09/delonghis-strange-brew-tracking-down-fake-amazon-raves. (2009).

[38] Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring networks of substitutable and complementary products. In *Proc. of KDD.*

[39] William Melicher, Blase Ur, Sean M. Segreti, Saranga Komanduri, Lujo Bauer, Nicolas Christin, and Lorrie Faith Cranor. 2016. Fast, lean and accurate: Modeling password guessability using neural networks. In *Proc. of Usenix Security.*

[40] Miguel Helft 2010. Charges settled over fake reviews on iTunes. http://www.nytimes.com/2010/08/27/technology/27ftc.html. (2010).

[41] Rada Mihalcea and Carlo Strapparava. 2009. The lie detector: Explorations in the automatic recognition of deceptive language. In *Proc. of ACL-IJCNLP.*

[42] Tomáš Mikolov. 2012. Statistical language models based on neural networks. *PhD Thesis, Brno University of Technology* (2012).

[43] Tomáš Mikolov, Anoop Deoras, Stefan Kombrink, Lukáš Burget, and Jan Černockỳ. 2011. Empirical evaluation and combination of advanced language modeling techniques. In *Proc. of Interspeech.*

[44] George A. Miller. 1995. WordNet: A lexical database for English. *Commun. ACM* 38, 11 (1995), 39–41.

[45] Arash Molavi Kakhki, Chloe Kliman-Silver, and Alan Mislove. 2013. Iolaus: Securing online content rating systems. In *Proc. of WWW.*

[46] Marti Motoyama, Damon McCoy, Kirill Levchenko, Stefan Savage, and Geoffrey M. Voelker. 2011. Dirty jobs: The role of freelance labor in web service abuse. In *Proc. of SEC.*

[47] Arjun Mukherjee, Abhinav Kumar, Bing Liu, Junhui Wang, Meichun Hsu, Malu Castellanos, and Riddhiman Ghosh. 2013. Spotting opinion spammers using behavioral footprints. In *Proc. of KDD.*

[48] Arjun Mukherjee, Vivek Venkataraman, Bing Liu, and Natalie S Glance. 2013. What yelp fake review filter might be doing?. In *Proc. of ICWSM.*

[49] Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T. Hancock. 2011. Finding deceptive opinion spam by any stretch of the imagination. In *Proc. of NAACL-HLT.*

[50] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *Proc. of ICML.*

[51] Ted Pedersen, Siddharth Patwardhan, and Jason Michelizzi. 2004. WordNet::Similarity: Measuring the relatedness of concepts. In *Proc. of HLT-NAACL (Demonstration Paper).*

[52] James W. Pennebaker, Cindy K. Chung, Molly Ireland, Amy Gonzales, and Roger J. Booth. 2007. The development and psychometric properties of LIWC2007. *Technical Report* (2007).

[53] Jakub Piskorski, Marcin Sydow, and Dawid Weiss. 2008. Exploring linguistic features for web spam detection: a preliminary study. In *Proc. of AIRWeb Workshop.*

[54] Posting positive reviews 2017. http://postingpositivereviews.blogspot.com. (2017).

[55] Mahmudur Rahman, Bogdan Carbunar, Jaime Ballesteros, and Duen Horng Polo Chau. 2015. To catch a fake: Curbing deceptive yelp ratings and venues. *Statistical Analysis and Data Mining: The ASA Data Science Journal* 8, 3 (2015), 147–161.

[56] Shebuti Rayana and Leman Akoglu. 2015. Collective opinion spam detection: Bridging review networks and metadata. In *Proc. of KDD.*

[57] Ehud Reiter, Robert Dale, and Zhiwei Feng. 2000. *Building natural language generation systems.* Vol. 33. MIT Press.

[58] Ehud Reiter, Somayajulu Sripada, Jim Hunter, Jin Yu, and Ian Davy. 2005. Choosing words in computer-generated weather forecasts. *Artificial Intelligence* 167, 1-2 (2005), 137–169.

[59] Reviews that stick website order form 2017. http://www.formstack.com/forms/?1653778-I3QqcHV4xC. (2017).

[60] Chen Rui, Yang Jing, Hu Rong-gui, and Huang Shu-guang. 2013. A novel LSTM-RNN decoding algorithm in CAPTCHA recognition. In *Proc.of IMCCC.*

[61] Sabrina Tavernise 2016. As fake news spreads lies, more readers shrug at the truth. https://www.nytimes.com/2016/12/06/us/fake-news-partisan-republican-democrat.html. (2016).

[62] Joshua Saxe and Konstantin Berlin. 2015. Deep neural network based malware detection using two dimensional binary program features. In *Proc. of MALWARE.*

[63] Saul Schleimer, Daniel S. Wilkerson, and Alex Aiken. 2003. Winnowing: local algorithms for document fingerprinting. In *Proc. of SIGMOD.*

[64] Iulian V. Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. 2016. Hierarchical neural network generative models for movie dialogues. In *Proc. of AAAI.*

[65] Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural responding machine for short-text conversation. In *Proc. of ACL-IJCNLP.*

[66] Eui Chul Richard Shin, Dawn Song, and Reza Moazzezi. 2015. Recognizing functions in binaries with neural networks. In *Proc. of Usenix Security.*

[67] Ilya Sutskever, James Martens, and Geoffrey Hinton. 2011. Generating text with recurrent neural networks. In *Proc. of ICML.*

[68] The 20 newsgroups data set 2008. http://qwone.com/~jason/20Newsgroups/. (2008).

[69] Tom Slee 2014. In praise of fake reviews. https://thenewinquiry.com/essays/in-praise-of-fake-reviews. (2014).

[70] Ross Turner, Somayajulu Sripada, and Ehud Reiter. 2010. Generating approximate geographic descriptions. In *Proc. of EMNLP.*

[71] Bimal Viswanath, Muhammad A. Bashir, Muhammad B. Zafar, Simon Bouget, Saikat Guha, Krishna P. Gummadi, Aniket Kate, and Alan Mislove. 2015. Strength in numbers: Robust tamper detection in crowd computations. In *Proc. of COSN.*

[72] Andreas Vlachos and Sebastian Riedel. 2014. Fact Checking: Task definition and dataset construction. In *Proc. of ACL.*

[73] Gang Wang, Tianyi Wang, Haitao Zheng, and Ben Y. Zhao. 2014. Man vs. machine: Practical adversarial detection of malicious crowdsourcing workers. In *Proc. of Usenix Security.*

[74] Gang Wang, Christo Wilson, Xiaohan Zhao, Yibo Zhu, Manish Mohanlal, Haitao Zheng, and Ben Y. Zhao. 2012. Serf and turf: crowdturfing for fun and profit. In *Proc. of WWW.*

[75] Guan Wang, Sihong Xie, Bing Liu, and Philip S. Yu. 2011. Review graph based online store review spammer detection. In *Proc. of ICDM*.
[76] William Yang Wang. 2017. "Liar, liar pants on fire": A new benchmark dataset for fake news detection. In *Proc. of ACL*.
[77] Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. In *Proc. of EMNLP*.
[78] Why Yelp has a review filter 2009. https://www.yelpblog.com/2009/10/why-yelp-has-a-review-filter. (2009).
[79] Yelp dataset challenge 2017. https://www.yelp.com/dataset_challenge. (2017).
[80] Yelp's review filter explained 2010. https://www.yelpblog.com/2010/03/yelp-review-filter-explained. (2010).
[81] Zhenlong Yuan, Yongqiang Lu, Zhaoguo Wang, and Yibo Xue. 2014. Droid-sec: deep learning in Android malware detection. In *Proc. of SIGCOMM (Demonstration Paper)*.
[82] Qing Zhang, David Y. Wang, and Geoffrey M. Voelker. 2014. DSpin: Detecting automatically spun content on the web. In *Proc. of NDSS*.