Wide-Ranging Review Manipulation Attacks: Model, Empirical Study, and Countermeasures

Parisa Kaghazgaran, Majid Alfifi, James Caverlee
Texas A&M University
College Station, Texas
{kaghazgaran,alfifi,caverlee}@tamu.edu

ABSTRACT

User reviews have become a cornerstone of how we make decisions. However, this user-based feedback is susceptible to manipulation as recent research has shown the feasibility of automatically generating fake reviews. Previous investigations, however, have focused on generative fake review approaches that are (i) domain dependent and not extendable to other domains without replicating the whole process from scratch; and (ii) character-level based known to generate reviews of poor quality that are easily detectable by anti-spam detectors and by end users. In this work, we propose and evaluate a new class of attacks on online review platforms based on neural language models at word-level granularity in an inductive transfer-learning framework wherein a universal model is refined to handle domain shift, leading to potentially wide-ranging attacks on review systems. Through extensive evaluation, we show that such model-generated reviews can bypass powerful anti-spam detectors and fool end users. Paired with this troubling attack vector, we propose a new defense mechanism that exploits the distributed representation of these reviews to detect model-generated reviews. We conclude that despite the success of neural models in generating realistic reviews, our proposed RNN-based discriminator can combat this type of attack effectively (∼90% accuracy).

CCS CONCEPTS

• **Security and privacy** → *Social aspects of security and privacy*; • **Computing methodologies** → *Natural language processing*.

ACM Reference Format:
Parisa Kaghazgaran, Majid Alfifi, James Caverlee. 2019. Wide-Ranging Review Manipulation Attacks: Model, Empirical Study, and Countermeasures. In *The 28th ACM International Conference on Information and Knowledge Management (CIKM'19), November 3–7, 2019, Beijing, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3357384.3358034

1 INTRODUCTION

User reviews have become a cornerstone of how we make purchase decisions – from what products to buy (e.g., Amazon), restaurants to patronize (e.g., Yelp), and apps to install (e.g., the App Store). However, allowing users to share their opinions has a dark side as these reviews are vulnerable to manipulation, casting doubt on the reliability of many online reviews [22, 25].

In one direction, large crowd-based manipulation campaigns can spread fake reviews in these systems [11, 17–19, 23, 47]. In these campaigns, a crowd of workers are paid in exchange for positive reviews, showcasing the potential of marshalling large human workforces to undermine the trustworthiness of online reviews. Since these reviews are written by humans and paymasters typically require them to write realistic reviews – e.g., reviews often have to meet a minimum length and contain positive but not skeptical comments – they often go undetected by modern detection algorithms that focus on review content. Still, crowd campaigns may leave manipulation traces, e.g., as reviews arrive synchronized in time [19] or forming a dense community over the large and mostly sparse co-review graph [18, 47], which can be helpful in their detection.

Recent advances in deep learning have shown the possibility of *automating* review manipulation in which these crowd workers can be replaced by neural models. In seminal work, Yao et al. [52] presented the current state-of-the-art approach to generate fake reviews. Such an approach can help manipulators overcome the limitations introduced by crowd-based campaigns creating new challenges for defense mechanisms, including:

- *Scalability:* since automated approaches do not rely on paying workers, large-scale attacks can be launched.
- *Increased deception:* since automated approaches can potentially obfuscate signals left by crowd campaigns (e.g., by varying the rate of posting fake reviews) that are helpful for detection, difficult-to-detect attacks can be executed.

However, generating high quality reviews that are readable by humans is a challenge on its own. In a nutshell, the work of Yao et. al., [52] leverages neural language models to generate synthetic reviews. The language model is trained over restaurant reviews from Yelp at a character-level granularity. Once trained, the model generates reviews character by character. We refer to this model as *CharLSTM*. CharLSTM generates domain dependent reviews meaning the whole training process needs to be replicated from scratch to generate reviews for any other arbitrary domain (e.g., Mobile accessories on Amazon or Apps on App Store and so on). The proposed approach is not fully automated as it applies a post-processing step (known as customization) which replaces some generated words with more suitable ones. Also, character-level language models need to capture longer dependencies and learn spelling in addition to syntax and semantics, so they are likely to become grammatically more error-prone.

In this paper, we identify a new class of attacks that are capable of generating fake reviews at (i) word-level granularity and

CIKM '19, November 3–7, 2019, Beijing, China

ACM ISBN 978-1-4503-6976-3/19/11…$15.00
https://doi.org/10.1145/3357384.3358034

that are (ii) transferable across different domains. First, the main advantage of character-level over word-level language modeling is its significantly smaller vocabulary. However, online reviews tend to be short, and centered around a limited range of topics determined by the review domain. For example, the quality of food and service in restaurants on Yelp or the value of a stay reviewed on Airbnb. Therefore, users typically adopt a limited vocabulary (~30k) compared to a general language domain (~300k) [31], meaning that word-level models could potentially generate high-quality reviews that avoid traditional traps like misspelling errors. Second, by leveraging recent advances in transfer learning in NLP tasks [13], we can develop models that can easily target new domains for which we have only limited training data, leading to potentially wide-ranging attacks on review systems. The main idea is to develop a universal model from a large collection of reviews (say from Yelp) to capture the general properties of the language used in online reviews and to comprehend the commonly used linguistic patterns. The intuition is that users use similar language to write reviews on different domains, e.g., *I enjoyed the food* and *I enjoyed using this App* differ only in domain-dependent vocabularies while the surrounding words express similar semantics. Hence, we can transfer the knowledge obtained during training of the universal model to conduct learning for any desired *target domain* efficiently with possibly smaller review samples as it only needs to adapt to the idiosyncrasies of the target review domain. We consider mobile accessories reviews on Amazon and mobile App reviews on App store as the target domains.

Based on this model, we conduct a comprehensive empirical study of its effectiveness in generating high-quality fake reviews. We find that machine learning spam detectors cannot distinguish synthetic reviews from real reviews. Furthermore, through a user study (N=300 for each of three domains, i.e., N=900 in total) we find that human examiners perceive synthetic reviews as real ones, meaning that new neural generative models of fake reviews pose serious risks to the validity of online review platforms. We compare model-generated reviews with fake reviews written by crowd workers and find that human examiners cannot distinguish between the two. We further compare our approach with Yao et al. [52]. Paired with this troubling attack vector, we propose a new defense mechanism that exploits the distributed representation of these reviews to distinguish between real and model-generated ones. We propose an RNN-based discriminator that can uncover automated fake reviews with high accuracy. Concretely, our main contributions are:

(1) We introduce a new class of attacks on online review platforms using *transfer learning* to automate review generation across different domains (Section 2).
(2) We perform a comprehensive empirical study on the robustness of synthetic reviews against traditional spam detectors and human examiners. We show that our proposed framework beats the baselines and competes with crowd written fake reviews (Section 4).
(3) We propose a new defense mechanism that leverages the distributed representation of the reviews to detect synthetic reviews with high accuracy (Section 5).

2 THE PROPOSED FRAMEWORK: DIOR

In this section, we present the design of our proposed framework DIOR for Domain Independent Online Review generation (Figure 1). DIOR is inspired by recent advances in neural language models [13, 30, 31]. DIOR comprises two steps: (i) building the universal model; and (ii) refining to the target domains. The main assumption is to transfer the knowledge from a source domain with large training samples to a target domain efficiently with possibly smaller number of samples. Therefore, we pick Yelp as the source model where significant amount of its reviews are available as described in Table 1. We also show empirically how much target training samples are sufficient to build the transferred model.

2.1 Background Information

We begin with a quick refresher on the basics of language models based on recurrent neural networks (RNNs) [2, 32, 33] before diving in to the design of DIOR. Readers familiar with this topic can jump ahead to Section 2.2. RNNs build a "memory" cell [10] to maintain the information about what it has seen from the input sequence so far and transfer the information to the next time step. An RNN cell is composed of a set of high dimensional weights learned during the training stage to capture the dependency among the words in the training samples.

Training stage. At each time step t, the network takes in the current word w_t along with the current hidden state h_t, that encodes the sequence till the time step t, and outputs a distribution over the vocabulary for the next word. The output distribution essentially describes the probability of observing each word w' in the vocabulary given the sequence w(<=t) ($P(w'|w_1, \ldots, w_t)$). The output is then compared with the desired output w_{t+1} that is the next word in the training sequence through cross-entropy loss defined as follows:

$$J(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{j=1}^{|V|} y_{t,j} \log \hat{y}_{t,j} \tag{1}$$

where y is one-hot vector wherein the index corresponding to the desired output word w_{t+1} is set to one. $\hat{y}$ is the model probability vector activated by the softmax function that is interpreted as the probability distribution over the words. Both y and $\hat{y}$ are V-dimensional vectors, where V is the size of vocabulary and T denotes the length of the sequence. The network parameters are then updated over multiple iterations to minimize the loss value.

Generating stage. This is an auto-regressive approach, where the trained language model can be used to generate a sequence of words. It begins by taking in the initial hidden state h_0 and word w_0. At each time step t, it takes in the word predicted at $t-1$ along with the hidden state h_{t-1} and predicts the distribution for the next word w_t and also updates the hidden state to h_t. By feeding w_t back to the model, it produces another probability distribution to predict the next word.

Diversity Control. To control the diversity of predicted words, a hyper-parameter called temperature τ is used in the generating stage by scaling the output vector o before applying the softmax function. In other words, the logits in the output vector are divided by τ. The softmax function over o produces probability value $\hat{y}_k$:

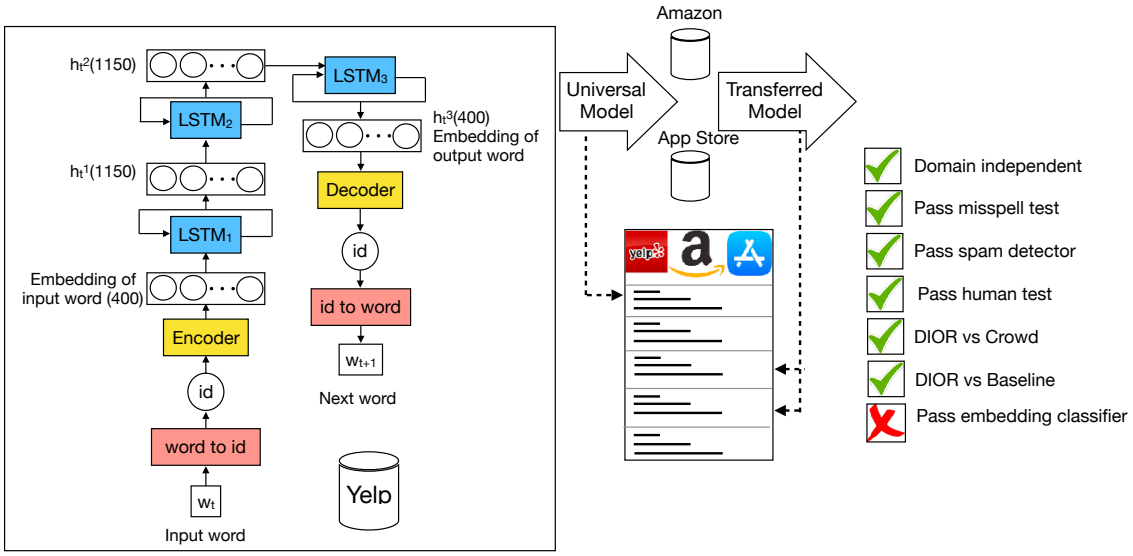


Figure 1: DIOR framework to generate domain independent fake reviews and its advantages.

$$\hat{y}^k = \frac{e^{o^k/\tau}}{\sum_{j=1}^{V} e^{o^j/\tau}} \tag{2}$$

where o^k represents the element of the output vector corresponding to the word at index k in the vocabulary. When τ is set to be 1, the softmax is directly computed on the logits. Lower temperatures result in more conservative predictions since it is less likely to pick from unlikely word candidates. The next word is selected based on a multinomial distribution over $\hat{y}$ probability vector.

2.2 Universal Model

Given this background, we turn now to the design of DIOR. We first start by building a universal model over a large collection of user reviews. We use a collection of Yelp reviews. R^Y indicates the concatenation of the all reviews in the training set. Since the language model aims to predict the next word at each time step, the labels are just like the inputs but shifted by one word. For example, if the input sequence is "we ate at this restaurant" then the language model is trained to predict "ate at this restaurant" sequentially given the first word (we). Hence, labels are defined as $L^Y = R^Y[1:]$.

Then, we break down R^Y into sequences with respect to the number of the time steps (T) in the language model. For faster gradient decent update, the resulting sequences are divided into mini batches with respect to the batch size (bs). Therefore, for a given mini batch b, the input to the learning algorithm would be matrix X_b^Y of shape $T \times bs$ where values represent the reviews tokens. The label matrix y_b^Y is built similarly over labels L^Y.

We now describe the architecture of the DIOR at time step t which can be generalized for any time step. It consists of three layers: an Encoder, 3-layer stacked-LSTMs, and Decoder. The encoder layer uses a trainable matrix (W_e^Y) to learn the embedding representation of the input tokens. The matrix shape is defined by vocabulary size V and embedding size em. The embedding for words are obtained by multiplying their one-hot representation with the W_e^Y

$$\xi(X_{b,t}^Y) = W_e^Y \times X_{b,t}^Y$$

From these embedding representations, we employ a stacked-LSTM (3 layers) to model dependency between words.

$$h_t^1 = LSTM(\xi(X_{b,t}^Y), h_{t-1}^1)$$

$$h_t^2 = LSTM(h_t^1, h_{t-1}^2)$$

$$h_t^3 = LSTM(h_t^2, h_{t-1}^3)$$

The Decoder layer is a linear transformation which decodes the output of the last LSTM layer (h_t^3). However, we use the "Weight Tying" technique that allows sharing of weights between encoder and decoder layers to reduce the number of learned parameters [14]. Therefore, h_t^3 is decoded using the transpose of the embedding matrix (W_e^Y) and the result vector is activated by the softmax function to produce the probability distribution over the words in the vocabulary. It should be noted that the hidden size of the last LSTM layer is set to be equal to the embedding size em.

$$\hat{y} = softmax(h_t^3 \times (W_e^Y)^\intercal)$$

The output $\hat{y}$ is then compared with desired labels $y_{b,t}^Y$ through cross entropy loss as defined in Equation 1 and model parameters are updated accordingly during multiple iterations.

In addition to this architecture, Howard et. al., [13] introduce two regularization techniques that capture the dependency between words in the language more effectively. These techniques are centered around fine-tuning the learning rate η i.e., a hyper-parameter that controls how much to update the model parameters (weights) with respect to the loss gradient.

The first technique called *Discriminative Fine-tuning* suggests tuning each layer with different learning rates instead of using a single learning rate through all layers of the model. The intuition is

that different layers capture different features [53], so they should be tuned differently. Considering the update process at time step t:

$$\theta_t^Y = \theta_{t-1}^Y - \eta \cdot \nabla_{\theta^Y} J(\theta^Y)$$

where θ^Y represents the model parameters, and $\nabla_{\theta^Y} J(\theta^Y)$ is the gradient with respect to the cost function. With this technique, the update process at each layer l would be:

$$l(\theta_t^Y) = l(\theta_{t-1}^Y) - \eta^l \cdot \nabla_{l(\theta^Y)} J(l(\theta^Y))$$

The second technique called *Slanted triangular learning rates (STLR)* suggests that not only we do need to consider different learning rates for different layers but also that we need to change the learning rate through the training iterations rather than using a fixed learning rate. The intuition is that varying the learning rate helps the model converges efficiently. Through this technique, the learning rate first linearly increases and then linearly decays.

After training, the set of learned model parameters θ^Y are used to generate reviews for our universal model. Empirically, we will study Yelp as our source model in the following.

2.3 Transferred Model

Given this universal model, we now turn to transferring this model to new domains for which we have only limited training data. In this way, a single learned model can potentially be used to launch attacks against a host of other review systems. For ease of presentation, we assume in this section that the target domain is Amazon and indicate the corresponding notations with Am superscript e.g., θ^{Am} denotes the target model parameters. In practice, the target domain could be any domain for which some reviews can be sampled.

A straightforward approach for transfer learning is to focus on the model's first encoder layer and initialize the weights with pre-trained word-embeddings [38, 39] or embedding matrix W_e^Y learned for Yelp vocabularies in our case. However, this approach still trains the target language model from scratch and treats pre-trained word embeddings as fixed parameters.

A recently introduced approach [13] proposes to transfer the knowledge from *all the layers* to benefit from source model to the fullest. Hence, we use the same architecture comprising an encoder, 3-layer LSTMs, and decoder and initialize the refined model parameters θ^{Am} with the parameters of the universal model θ^Y.

Initialization. The parameters of the Yelp generative model (θ^Y) are reused as the starting point for the Amazon generative model. However, the vocabularies are not the same in the two domains, so the universal embedding matrix W_e^Y with the shape of $|V^Y| \times |em|$ cannot be used directly because the size of the transferred model's embedding matrix W_e^{Am} needs to be $|V^{Am}| \times |em|$. The average over embedding of the common words in the two domains is used to initialize the unseen words e.g., Amazon domain dependent words.

$$m = Avg(w_e^i \forall w^i \in V^Y \wedge w^i \in V^{Am})$$

$$W_e^{Am} = ||w_e^i \forall w^i \in V^Y \wedge w^i \in V^{Am}|| \ldots ||m \forall w^i \in V^{Am} \wedge w^i \notin V^Y||$$

After initialization, the target model is trained using the same set of techniques introduced in Section 2.2. Together, this DIOR model design promises the potential of generating high-quality reviews at word-level granularity in an inductive transfer-learning framework.

3 EXPERIMENTAL DESIGN

Before turning to our empirical study of the proposed DIOR framework, we describe our experimental design. We consider Yelp and Amazon/App Store reviews as the source and target domains respectively. Each review contains the text of the review, and the rating score in the range of one to five stars. We present the results based on reviews with positive sentiment, so we keep only the reviews with 5-star ratings. We split the datasets into training and validation sets with ratio of 90% and 10%. Four disjoint datasets are used for generating and evaluating synthetic reviews. Table 1 summarize the statistics on these datasets.

Yelp. We use the Yelp Challenge Dataset (round 11) released in January 2018.[1] This dataset contains ~5m reviews targeting ~174k businesses. We extract reviews corresponding to restaurants and find 318,392 five-star reviews containing 31,514,567 total words and 35,394 unique words, a sufficiently large dataset to serve as our transfer learning source task.

Amazon. This dataset introduced in [28] includes Amazon product reviews across a variety of categories. In our evaluation, we focus on the cell-phone accessories category and extract 108,664 five-star reviews with 10,633,295 total words and 20,731 unique words.

App Store. This dataset contains reviews about mobile applications with 231,199 five-star reviews and 12,131,926 words and 24,614 unique words introduced in [23].

Model-Generated Dataset. This dataset contains the reviews generated by the proposed language model. The synthetic reviews for Yelp are generated directly from the universal model. The Amazon and App store generated reviews are results of transferring the domain from Yelp to the corresponding domain.

Reproducibility. We are interested in a general model that performs robustly across different domains. Therefore, we use the same set of hyper-parameters as reported in [13], which have shown good success at efficient convergence. The language model has an embedding size of 400, 3 layers, and hidden size of 1,150. It applies Adam optimizer with $\beta_1 = 0.7$ and $\beta_2 = 0.99$. The batch size is set to 64 and the base learning rate for fine-tuning is set to 0.004. The number of epochs are tuned on the validation set.

Review Generation. To generate reviews, the initial word represents the beginning of the reviews which we define by a special token $< sor >$. The generation process continues until the model predicts the end of the review identified by a special token $< eor >$ or the sequence length becomes equal to the median length of the reviews in the corresponding dataset. We generate reviews at temperatures $[0.2, 0.4, 0.6, 0.8, 1.0]$. Intuitively, generation at low temperatures reinforces the difference between the occurrence probability of the words and reduces the chance of words with lower probability to be predicted. At low temperatures, the model tends to generate sequences commonly seen in the training data, so it generates repetitive patterns. By increasing the temperature, rarer words become visible to the predictor at the cost of grammar mistakes and incoherency. We evaluate quality of the generated reviews at different temperatures and recognize the optimal temperature

[1] https://www.yelp.com/dataset

Table 1: Summary of Review Datasets

	# Reviews	# Tokens	Vocabulary size	Median review length	Average review length	Training reviews	Validation reviews
Yelp	318,392	31,514,567	35,394	69	100	286,552	31,840
Amazon	108,664	10,633,295	20,731	49	97	97,797	10,867
App store	231,199	12,131,926	24,614	41	53	208,079	23,120

Table 2: Example of the synthetic five-star reviews targeting restaurants on Yelp, mobile accessories on Amazon, and mobile Apps on the App store at different temperatures.

Temperature	**Generated Reviews (Yelp)**
0.2	I love this place ! I 've been here several times and I 've never been disappointed . The food is always fresh and delicious . The service is always friendly and attentive . I 've been here several times and have never been disappointed .
0.4	I 've been to this location twice now and both times I 've been very impressed . I 've tried their specialty pizzas and they 're all really good . The only problem is that they 're not open on sundays . They 're not open on sundays .
0.6	I have been coming to this place for years and have always had great food and service . They have a great lunch buffet . They have a great selection of food for the price . They do have a lot of seating and I would recommend reservations .
0.8	I 've eaten here about 8 times . I 've been introduced to this place . Its always busy and their food is consistently great . I LOVE their food , hence the name . It is so clean , the staff is so friendly , and the food is great . I especially like the chicken pad thai , volcano roll , and the yellow curry .
1.0	this is strictly to go . Love , love , love the food ! we usually usually get brisket (oh my) , sandwich (pastrami , or pork , just so good) and now these are my two favorites . It 's great . This is gone (according to our waitress) .
Temperature	**Generated Reviews (Amazon)**
0.2	I have been using this case for a few weeks now and I love it ! I have had this case for about a month now and it is still holding up great ! I have dropped my phone a few times and the case has protected it perfectly ! I would recommend this case to anyone !
0.4	after reading the reviews I read the reviews and decided to give it a shot . I am very pleased with the results . The quality is great , it fits perfectly and I do n't have any problems with it . It 's a great value for the price .
0.6	this is a nice case . It 's a little difficult to remove , but that 's to be expected . The case is slightly thicker than a regular screen protector , but that is to be expected . It 's a great phone case and I highly recommend it .
0.8	the case works great ! it has a soft rubber insert that goes over the hard shell . The hard plastic shell has a soft inner shell and the hard case is hard plastic . It is very sticky and has not fallen out or dropped or fallen apart .
1.0	I love having different cases for my phone but this one is well worth the money ! it fits the phone great ! feels great not slick at all ! also the quality of the screen protector was very nice ! would definitely recommend this .
Temperature	**Generated Reviews (App store)**
0.2	I love this game so much ! it 's so fun and addicting ! I love the fact that you can play with friends and family !
0.4	this app is great ! I have been using it for years and it has always been reliable and reliable . I have been using it for a long time now and it is always reliable .
0.6	this app is great for learning the basics of math ! I love that it has a different function that can help you learn the words that you understand . I wish all apps were this simple .
0.8	this app is a great tool for discovering new things : being able to search for films and putting reviews on particular items as well as having a way to download stories from the app .
1.0	well , this game is pretty fan rank alive and the battle system is really hard , but it does n't require super flick . It breaks my overall playability . My only issue is it 's too short , but they 're always adding new levels / things to it .

value. Samples of generated reviews are shown in Table 2. It is notable that generated reviews at temperature 0.2 tend to repeat themselves.

4 EMPIRICAL STUDY

In this section, we perform a comprehensive empirical study to evaluate the quality of the reviews generated by DIOR across the three domains. *First*, we investigate if synthetic reviews are distinguishable from real reviews based on their linguistic features by developing a spam detector. *Second,* we conduct a user study to understand how susceptible human readers are to synthetic reviews. *Third*, we evaluate the model-generated reviews against fake reviews written by crowd-based review manipulators. *Fourth*, we compare DIOR with state-of-the-art work [52]. Finally, we investigate the impact of transfer learning. Overall, we aim to show advanced language models make it easier to manipulate review platforms.

Table 3: Performance of spam detector. From temperature 0.8 synthetic and real reviews become indistinguishable

	Yelp					Amazon					App store				
Temperature	**0.2**	**0.4**	**0.6**	**0.8**	**1.0**	**0.2**	**0.4**	**0.6**	**0.8**	**1.0**	**0.2**	**0.4**	**0.6**	**0.8**	**1.0**
Accuracy (%)	92	81	73	64	55	92	82	75	61	56	91	77	69	62	54
Precision (%)	93	82	74	65	56	93	83	76	64	57	92	80	71	62	55
Recall (%)	93	82	74	65	55	92	82	75	61	56	91	78	70	62	54
F1 score (%)	93	82	74	65	54	91	82	75	60	54	91	77	69	62	53

4.1 Spam Detector

Here we develop a supervised machine-learning scheme to evaluate if model-generated reviews carry different linguistic patterns from real reviews. Text classification has been widely used to detect opinion spam on the web [15, 36, 40]. We describe four groups of linguistic features consisting of 12 features in total, following the approach proposed in [52] to classify the reviews.

Similarity feature (1): Measures the inter-sentence similarity within a review. It computes the cosine-similarity between each pair of sentences based on their unigram tokens and considers the maximum value as the similarity feature [7].

Structural features (2): Captures the structural aspects of a review including the average sentences length measured by their number of words and the average word length measured by their number of characters [42].

Syntactic features (5): Defines linguistics properties of a review based on parts-of-speech (POS) tagging process. It includes percentage of nouns, verbs, adjectives, adverbs, and pronouns [42].

Semantic features (4): Captures the sentiment and subjectivity of a review including percentage of positive, negative, subjective, and objective words. We use SentiWordNet library [3] to extract this type of feature.

For each domain and at each of the temperatures, we sample 10k model-generated reviews and 10k real reviews, for a total of 20k reviews. We split this data into training and testing sets with a ratio of 70% and 30% respectively. We train an SVM classifier with rbf kernel and c=1 (obtained from [0.001, 0.01, 0.1, 1, 10] set through grid-search). After training over all 12 linguistic features, we test the performance of the classifier over the testing data. Results are averages of 10 runs. It should be noted that we tried different classifiers such as Logistic Regression and Linear SVM and observe similar results. Hence, we report the results obtained from SVM as representative standard machine learning classifier.

Our evaluation metrics are average of precision, recall and F1-score over both classes of reviews and overall accuracy. Table 3 reports the performance of the classifier at different temperatures across different domains. We observe that text classification shows high detection performance at lower temperatures. For example, at temperature 0.2 it classifies reviews with 0.92, 0.92, and 0.91 accuracy in Yelp, Amazon, and the App store domains respectively. We can relate this trend to the fact that reviews generated at lower temperatures tend to repeat themselves (Table 2), which makes features like inter-sentence similarity an informative feature to distinguish two class of reviews.

However, we are more interested to evaluate the performance at temperature 0.8 as we find this to be an optimal temperature in a qualitative analysis by human readers (Section 4.2). We observe that model-generated reviews can circumvent the linguistic-based test when an appropriate temperature is selected. Low evaluation metrics at temperature 0.8 – e.g., 0.64, 0.61 and 0.62 accuracy across Yelp, Amazon, and the App store respectively – indicate synthetic reviews do not resemble spam behavior.

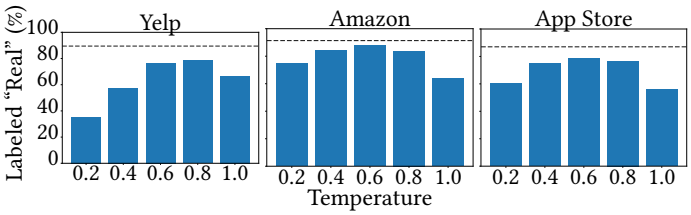


Figure 2: Majority of the synthetic reviews at temperatures 0.6 and 0.8 are recognized as real. Dot lines indicate percentage of real reviews that were labeled as real by humans.

Finding 1: The linguistic-based spam detector may not distinguish synthetic reviews from real reviews.

4.2 User Study

Regardless of the ability of machine-generated reviews to bypass computational-based detectors, the real test is to evaluate their quality by human readers. For this purpose, we set up a crowdsourcing user study to examine whether these model-generated reviews are convincing to human readers. We post surveys on Amazon Mechanical Turk (AMT) to assess reviews. Each survey includes a guideline, and a set of reviews. The guideline highlights two main points: 1) Turkers are tasked to mark each review as either real or fake. And, for those recognized as fake, they are asked to provide their reasoning; 2) the guidelines shows a sample of real reviews to the users emphasizing that online reviews are not necessarily well-structured pieces of text. This prevents many real reviews from being marked as fake due to their informal languages.

For each domain, we design 100 surveys each with 10 reviews out of which 5 are real reviews and 5 are synthetic reviews, each generated at one of [0.2, 0.4, 0.6, 0.8, 1.0] temperatures. Each unique survey is assigned to 3 workers (3 HITs [2] per task), giving us a total of 300 surveys. To ensure the quality of responses, we insert a trivial question into each survey, which asks the Turker to check if a mathematical equation is False or True. It mitigates the risk of blindly answering surveys in our evaluation. Furthermore, we only accept surveys where the Turker dwelled on the survey for at least 5 minutes. We also restrict our tasks to those who are located in the United States to guarantee English literacy.

[2]Human Intelligence Task

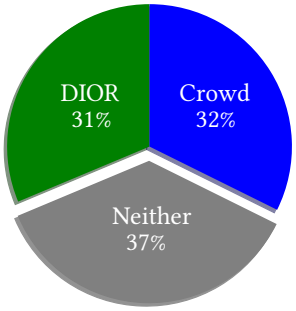


Figure 3: Users perceive DIOR generated reviews as reliable as crowd written fake reviews.

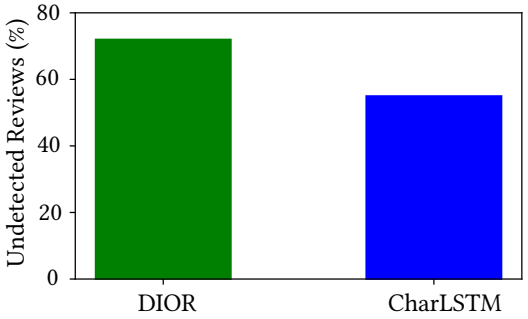


Figure 4: Comparison with baseline.

Figure 2 demonstrates the performance of synthetic reviews against human judgment at various temperatures. The key point is that such reviews remain quite robust and many of them are recognized as real. Under the best configuration in Yelp, i.e., temperatures 0.6 and 0.8 the percentage of reviews flagged as real is 76.42% and 78.42% respectively. At the same time 89.52% of real reviews are labeled correctly showing a 10% error while they are supposed to pass the human test perfectly. We can relate this to the fact that user-written reviews could be also error-prone on any basis making it a challenge to humans to distinguish between these two types of reviews. In addition, similar to algorithmic evaluation, the DIOR performance improves with the increase in the temperature.

By examining Turkers' reasoning while they flag a review as synthetic, we find that many reviews at low temperatures e.g., 0.2 and 0.4 are identified as fake due to repetition and they sound robotic even though they are grammatically correct. On the other hand, the performance of such reviews improves at higher temperatures.

However, we can observe a downfall of the accuracy at temperature 1.0 as examiners believe the reviews may not be coherent and contain a nonsensical argument. We can recognize the best performance occurs at temperature 0.8 considering that reviews at this temperature perform well against the ML-based classifier as well. A similar pattern is observable in both other domains.

Finding 2: Reviews generated at temperature 0.8 can fool human readers and go undetected.
Finding 3: Human readers are more sensitive to repetition errors than they are to small grammar mistakes.

Hence, in the following studies (Sections 4.3, 4.4 and 4.5), we focus on reviews generated at temperature 0.8 as it is demonstrated to be the optimal threshold for generating reliable reviews understood by humans and at the same time pass the algorithmic detectors. Also, we follow the same guidelines as described in this section to ensure quality of responses.

4.3 DIOR versus Crowd Manipulators

One of the most prominent ongoing attacks on review platforms is to spread fake reviews through organized manipulation campaigns [4, 49]. In these campaigns, a crowd of workers are paid in exchange of positive reviews. Here we are eager to evaluate how end users perceive the fake reviews generated by our approach compared with those written by crowd manipulators. This study answers the important question of whether neural language models are capable enough to take the place of the crowd campaigns?

We first prepare a ground-truth of crowd written fake reviews. [18] introduces a dataset of fake review writers on Amazon who are actively engaged in manipulation tasks. However, the focus of this work is to detect manipulators and not to identify individual reviews as fake or real. We label reviews as fake using the notion of *self-plagiarism* [19] in which a manipulators simply duplicate a single review on two different products. From this pool of fake reviews, we filter out reviews about mobile accessories for the fair comparison with our Amazon synthetic reviews.

Second, we design 20 surveys each with five pair of reviews each generated by either crowd manipulators or DIOR framework. Each unique survey is assigned to 3 Turkers (3 HITs per task), giving us a total of 300 ($20 \times 5 \times 3$) pairs of reviews. They are tasked to select which of the two reviews sound fake to them or none if they find them to be equally reliable.

As Figure 3 shows, both crowd and model generated reviews are equally likely to be detected by human evaluators as fake. For example, 32% and 31% of the answers found crowd written fake reviews and reviews generated by DIOR as suspicious respectively while 37% of the answers found both type of the reviews equally reliable.

Finding 4: Users find reviews generated by DIOR as reliable as fake reviews written by manipulation campaigns.

4.4 DIOR versus the state-of-the-art Model

We turn to compare the effectiveness of our proposed DIOR model to the state-of-the-art work [52]. In summary, CharLSTM is a two-layer character-based LSTM trained over Yelp challenge dataset and generates restaurant reviews. However, the implementation code or a sample of fake reviews generated by this model is not publicly available. Hence, we replicate the model as closely as we could based on the configurations reported in the paper (Section 3.2 Training Process).

Now we conduct a user-study and design 20 surveys each with 20 reviews out of which 12 are real reviews, 4 are fake reviews generated by CharLSTM and 4 are fake reviews generated by DIOR. Each unique survey is assigned to 3 Turkers (3 HITs per task), giving us a total of 60 surveys and 1200 ($20 \times 20 \times 3$) reviews. Each participant is tasked to label four reviews as fake.

On average, the detection rate (recall) is 28% and 45% for reviews generated from DIOR and CharLSTM respectively. Figure 4 shows

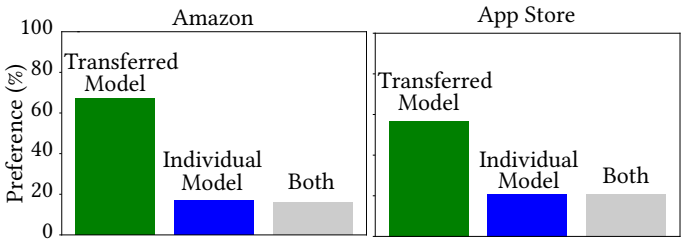


Figure 5: Users find transferred models more convincing than individual models

that DIOR generated reviews go undetected with higher probability (72% versus 55%).

Finding 5: DIOR beats the state-of-the-art model in generating effective reviews.

4.5 Transferred versus Individual Models

In the following two studies, we investigate the impact of transfer learning from two different aspects. First, how it improves the quality of synthetic reviews compared to individual models. Second, how much data is needed to refine the universal model.

Hence, we evaluate the quality of reviews generated from the transferred model against reviews generated from a language model trained from scratch – what we refer to as the individual model for our target domains. To do this, we set up a pair-wise comparison based on a survey-based user study to see how natural reviews sound to human readers.

For each target domain, we design five surveys each with five pair of reviews. Each unique survey is assigned to 10 Turkers, giving us a total of 50 surveys and 250 pairs of reviews. The task is to select which of the two reviews sound more natural or both if they find them to be equally natural. To ensure that users evaluate reviews based on only the language of the review, we pair reviews with a similar topic for each comparison. For example, both reviews talk about fitness applications in the App domain or headsets in the Amazon domain. As Figure 5 shows, for the App store, human readers found 57% of the reviews generated from the transferred model to sound natural as opposed to only 21% of reviews generated by individual model. 21% of responses found both reviews to sound equally natural. We find similar results for Amazon.

Finding 6: Using transfer learning not only facilitate the domain shift but also improves the performance significantly.

4.6 Training Size

It is critical to understand how much data is needed to refine the universal model. In particular, when the target domain's dataset is not sufficiently large we are eager to know how many reviews are required to converge the model. We plot the validation loss versus training size for Amazon and App transferred models in Figure 6. We gradually increase the training size starting with 25k reviews. According to this figure, we need approximately 100k and 75k training reviews for App and Amazon domains respectively to converge the loss values and achieve a relatively stable performance. In the Amazon domain, the model converges with smaller set of reviews and we can relate this trend to the size of the vocabulary.

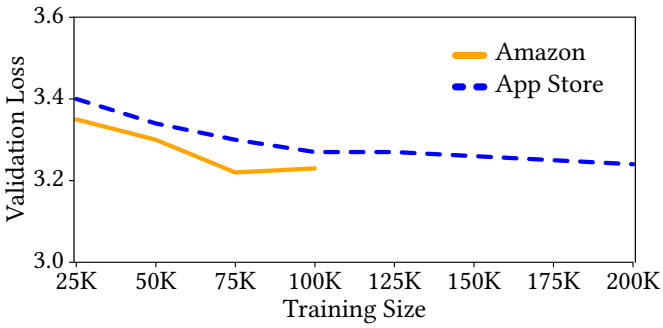


Figure 6: The transferred models need reasonably low number of samples to reach stable performance.

According to Table 1, there are about 20k unique words in Amazon (less than that of App store with 24k unique words) which helps the model to adapt to the domain more efficiently.

Finding 7: The transferred models need reasonably low number of samples compared to universal model to reach stable performance.

In summary, we showed that automating generation of fake reviews can expose serious threats to review platforms through seven different findings. It should be noted that our first two findings are in line with those reported in [52]. However, despite the success of DIOR in generating realistic reviews, in the next section, we aim to combat this threat by proposing a new defense mechanism.

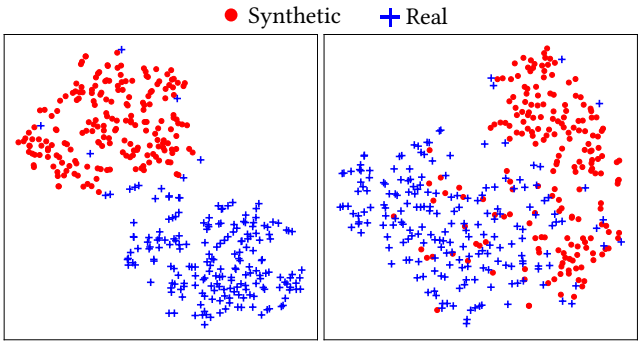
5 PROPOSED DISCRIMINATOR

Although model-generated reviews successfully pass the computational detectors and human tests, the question is whether RNN-based language models manage to model the real review distribution? To answer this question, we propose a discriminator capable of distinguishing synthetic reviews from real ones using the underlying distributed representation of review words learned during the training process of the language model.

The key insight is that language models predict the next word conditioned on previously seen words while humans are not restricted by this requirement when they write online reviews. Figure 7 visualizes the review embeddings over 400 Yelp review samples in two dimensions using t-SNE [26], with markers corresponding to synthetic and real reviews. Note that the review embedding is computed by composing their word embeddings. We adapt a straightforward approach that represents a review by taking the average over its word embeddings [34]. As we can see from Figure 7, synthetic reviews tend to cluster together in the embedding space in particular at low temperatures.

This motivates us to explore the manner of words appearing in a sequence in synthetic and real reviews. Our discriminator learns a classifier $M(R, \theta)$ that classifies a review R as real or synthetic given its terms and model parameters θ. A bipartite training sample consists of (i) embedding representation of review terms $\{\xi(w_1), ..., \xi(w_{|R|})\}$ extracted from embedding matrix W_e (Section 2); and (ii) its corresponding label y i.e., 1 or 0 indicating synthetic and real reviews respectively. For a given batch of training samples b, the loss function based on cross-entropy is defined as follows:

$$J(b;\theta) = \frac{1}{|b|}\sum_{i=1}^{|b|} y_i \log(M(R_i,\theta)) + (1-y_i)\log(1-M(R_i,\theta))$$



(a) Temperature 0.4 (b) Temperature 0.8

Figure 7: Generated reviews tend to cluster in the embedding space. Figure best viewed in color. (Yelp)

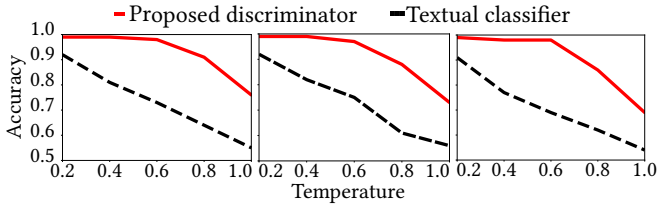
Where $M(R_i, \theta)$ and y_i indicate the predicted and actual labels of review R_i respectively.

Network Architecture. We opt for a straightforward LSTM network which is composed of: LSTM and linear layers. The LSTM is fed distributed representations of review terms (term embedding for short) and the output of the last hidden state is fed to the linear layer where it is activated by the softmax function and outputs the classification result.

Experimental Setup. For each domain and at each temperature, we sample 10k model-generated reviews and 10k real reviews from the training dataset, in total 20k reviews. We split the data into training, validation, and testing datasets with the ratio of 80%, 10%, and 10% respectively.

Reproducibility. To evaluate our key insight, we avoid setting up a highly tuned network and choose the hyper-parameters from singleton sets. We set the number of LSTM hidden states with respect to the median of review length in each domain (Table 1). We set input size, hidden size, learning rate, dropout rate, batch size, hidden layers, and optimizer to 400, 1150, 0.001, 0.1, 16, 2, and Adam respectively. We only tune the number of epochs based on the performance of the model on the validation dataset. That is, as accuracy on the validation set decreases the training process would stop to prevent the model from over-fitting.

Results. Figure 8 shows the detection performance across different domains at different temperatures. Due to the balanced dataset (equal representation of two classes) other evaluation metrics like recall, precision and f1-score remain similar to accuracy, so we only report the accuracy. The discriminator achieves significantly high accuracy at low temperatures [0.2, 0.4, 0.6] that is 0.99, 0.99 and 0.97 on Yelp and a similar pattern is observable over other domains. However, we are more interested to evaluate its performance at temperature 0.8 that is demonstrated to be the optimal temperature in our qualitative analysis. At this temperature, the discriminator is able to identify model-generated reviews with 0.92, 0.89 and 0.86 accuracy across different domains while the corresponding values obtained from the baseline textual classifier (spam detector) is 0.64, 0.62 and 0.61. These promising results motivate us to consider more complex architectures for the discriminator in our ongoing work to detect model-generated reviews at temperature 0.8 more accurately in order to minimize the impact of automated fake reviews.



(a) Yelp (b) Amazon (c) App Store

Figure 8: The proposed discriminator detects synthetic reviews with high accuracy and performs significantly better than textual classifier.

Finding 8: Model-generated reviews are detectable in the embedding space with high accuracy.

6 RELATED WORK

Text Generation. Natural language generation techniques place structured data into well-designed templates [27, 43]. These systems require rules and consistent format. Probabilistic approaches like N-gram models generate text by looking back only a few steps in the sequence [51]. While N-gram models exhibit limitation against long text sequences, RNN-based models perform based on complex "memory" gating which maintain longer term dependency [9, 30, 48]. The application of RNN-based text generators on different domains like chatbot [50], conversational systems [46], email auto-responses [20], movie dialogues [45] and image captioning [21] has shown successful results. In another direction, Generative Adversarial Networks (GANs) are explored to fill in the blanks in sentences [5]. They were originally designed to produce differentiable values that have direct application in computer vision, so generating discrete text is a challenge for them. On the other hand, researchers have shown that properly regularized RNN models can achieve state-of-the-art performance in generating text [29].

Review Generation. Character-level RNNs to capture the sentiment and meaning in product reviews has been proposed in [24]. Byte-level RNN to generate product reviews for sentiment classification is proposed in [41]. An N-gram based review generator is preliminary examined in an adversarial setting with the purpose of generating fake reviews [12] without considering any countermeasures. In another effort, [16, 52] propose a character-level RNN to generate fake reviews for the specific domain of Yelp. [16] uses sequence-to-sequence neural models to incorporate contextual information such as location into the model.

Transfer Learning. Similar to our work, a few studies also investigate the impact of transfer learning on NLP tasks. This includes work on sentiment classification [13], multilingual language modeling [6], machine translation [44] and question and answering [35]. To the best of our knowledge this is the first to explore the power of transfer learning in generating domain-independent online reviews.

Detection of Machine Generated Content. Statistical approaches like measuring TF-IDF, examining Zipf's law on the target text are proposed to detect machine-generated text with the focus on textual features [1, 37]. [16] uses N-gram features to classify reviews as real or synthetic. Recent advances in GANs that produce synthetic images suggest mimicking the underlying distribution of the image where the discriminator is responsible to distinguish between

the distribution of true and generated samples [8]. In this paper, inspired by the idea of the discriminator in GANs, we propose to distinguish between real and synthetic reviews based on distributed representation of their constituent tokens. It should be noted that the method proposed in [52] to defend against model-generated reviews, examines the character distribution of synthetic and real reviews as the language model is trained at character level granularity. Therefore, the direct application of this approach on word-level generators is ruled out.

7 CONCLUSION AND FUTURE WORK

We proposed and evaluated a wide-ranging class of attacks on online review platforms based on neural language models at word-level granularity using transfer-learning. The unique attribute of our work is being domain independent and can target any arbitrary review domain even with small available review samples. The main intuition is to: (i) develop a universal model to learn general linguistic patterns in review domain and transfer this knowledge to the domain-specific language; (ii) generate high quality reviews which are competitive with real reviews and can pass the quality test by both computational-based detectors and human evaluators; (iii) demonstrate that synthetic reviews do not completely mimic the true distribution of real reviews, so this is a powerful signal to detect automated fake reviews. Our results on discriminating generated reviews are promising. In our ongoing work, we aim to study the performance of other neural network architectures like CNN in modeling synthetic review distributions and to develop a more powerful discriminator.

Acknowledgement. This work was supported in part by NSF grant SaTC-1816497. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsors.

REFERENCES

[1] Roee Aharoni and et al. 2014. Automatic detection of machine translated text and translation quality estimation. In *ACL*.
[2] Ebru Arisoy, Tara N Sainath, Brian Kingsbury, and Bhuvana Ramabhadran. 2012. Deep neural network language models. In *NAACL-HLT*. ACL.
[3] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. Sentiwordnet 3.0: an enhanced lexical resource for sentiment analysis and opinion mining.. In *Lrec*.
[4] Amir Fayazi, , and et al. 2015. Uncovering crowdsourced manipulation of online reviews. In *SIGIR*.
[5] William Fedus and et al. 2018. Maskgan: Better text generation via filling in the _. *ICLR* (2018).
[6] Daniela Gerz and et al. 2018. On the Relation between Linguistic Typology and (Limitations of) Multilingual Language Modeling. In *EMNLP*.
[7] Wael H Gomaa and Aly A Fahmy. 2013. A survey of text similarity approaches. *International Journal of Computer Applications* (2013).
[8] Ian Goodfellow and et al. 2014. Generative adversarial nets. In *NIPS*.
[9] Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv* (2013).
[10] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* (1997).
[11] Bryan Hooi and et al. 2016. Birdnest: Bayesian inference for ratings-fraud detection. In *ICDM*.
[12] Dirk Hovy. 2016. The enemy in your own camp: How well can we detect statistically-generated fake reviews–An adversarial study. In *ACL*.
[13] Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *ACL*.
[14] Hakan Inan and et al. 2016. Tying word vectors and word classifiers: A loss framework for language modeling. *arXiv* (2016).
[15] Nitin Jindal and Bing Liu. 2008. Opinion spam and analysis. In *WSDM*.
[16] Mika Juuti and et al. 2018. Stay on-topic: Generating context-specific fake restaurant reviews. In *ESORICS*.
[17] Parisa Kaghazgaran and et al. 2017. Behavioral analysis of review fraud: Linking malicious crowdsourcing to amazon and beyond. In *ICWSM*.
[18] Parisa Kaghazgaran and et al. 2018. Combating crowdsourced review manipulators: A neighborhood-based approach. In *WSDM*.
[19] Parisa Kaghazgaran and et al. 2019. TOmCAT: Target-Oriented Crowd Review ATtacks and Countermeasures. In *ICWSM*.
[20] Anjuli Kannan and et al. 2016. Smart reply: Automated response suggestion for email. In *KDD*.
[21] Andrej Karpathy and Li Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *CVPR*.
[22] Srijan Kumar and Neil Shah. 2018. False information on web and social media: A survey. *Social Media Analytics: Advances and Applications* (2018).
[23] Shanshan Li and et al. 2017. Crowdsourced App Review Manipulation. In *SIGIR*.
[24] Zachary C Lipton and et al. 2015. Capturing meaning in product reviews with character-level generative text models. *arXiv* (2015).
[25] Michael Luca and Georgios Zervas. 2016. Fake it till you make it: Reputation, competition, and Yelp review fraud. *Management Science* (2016).
[26] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* (2008).
[27] Mitul Makadia. 2018. What Are the Advantage of Natural Language Generation and Its Impact on Business Intelligence?, https://www.marutitech.com/advantages-of-natural-language-generation/, Last Access: 01/28/2019. (2018).
[28] Julian McAuley and et al. 2015. Image-based recommendations on styles and substitutes. In *SIGIR*.
[29] Gábor Melis and et al. 2017. On the state of the art of evaluation in neural language models. *ICLR* (2017).
[30] Stephen Merity and et al. 2017. Regularizing and optimizing LSTM language models. *arXiv* (2017).
[31] Stephen Merity and et al. 2018. An Analysis of Neural Language Modeling at Multiple Scales. *arXiv* (2018).
[32] Tomáš Mikolov and et al. [n.d.]. Empirical evaluation and combination of advanced language modeling techniques.
[33] Tomáš Mikolov and et al. 2010. Recurrent neural network based language model. In *ISCA*.
[34] Tomas Mikolov and et al. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*.
[35] Sewon Min and et al. 2017. Question answering through transfer learning from large fine-grained supervision data. *arXiv* (2017).
[36] Arjun Mukherjee and et al. 2013. What yelp fake review filter might be doing?. In *ICWSM*.
[37] Hoang-Quoc Nguyen-Son and et al. 2017. Identifying computer-generated text using statistical analysis. In *APSIPA ASC*.
[38] Matthew Peters and et al. 2017. Semi-supervised sequence tagging with bidirectional language models. *arXiv* (2017).
[39] Matthew E et al. Peters. 2018. Deep contextualized word representations. *arXiv* (2018).
[40] Jakub Piskorski and et al. 2008. Exploring linguistic features for web spam detection: a preliminary study. In *AIRWeb*.
[41] Alec Radford and et al. 2017. Learning to generate reviews and discovering sentiment. *arXiv* (2017).
[42] Shebuti Rayana and Leman Akoglu. 2015. Collective opinion spam detection: Bridging review networks and metadata. In *KDD*.
[43] Ehud Reiter and et al. 2005. Choosing words in computer-generated weather forecasts. *Artificial Intelligence* (2005).
[44] Rico Sennrich and et al. 2015. Improving neural machine translation models with monolingual data. *ACL* (2015).
[45] Iulian Vlad Serban and et al. 2015. Hierarchical neural network generative models for movie dialogues. *CoRR, abs/1507.04808* (2015).
[46] Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural responding machine for short-text conversation. *arXiv* (2015).
[47] Kijung Shin, , and et al. 2017. D-cube: Dense-block detection in terabyte-scale tensors. In *WSDM*.
[48] Ilya Sutskever and et al. 2011. Generating text with recurrent neural networks. In *ICML*.
[49] Gang Wang, , and et al. 2012. Serf and turf: crowdturfing for fun and profit. In *WWW*.
[50] Anbang Xu and et al. 2017. A new chatbot for customer service on social media. In *CHI*.
[51] Arun Kumar Yadav and Samir Kumar Borgohain. 2014. Sentence generation from a bag of words using N-gram model. In *ICACCCT*.
[52] Yuanshun Yao and et al. 2017. Automated crowdturfing attacks and defenses in online review systems. In *CCS*.
[53] Jason Yosinski and et al. 2014. How transferable are features in deep neural networks?. In *Advances in neural information processing systems*.