Modeling Coherency in Generated Emails by Leveraging Deep Neural Learners

Avisha Das and Rakesh Verma

University of Houston
Department of Computer Science
Houston, TX 77204, USA
adas5@uh.edu, rverma@uh.edu

Abstract. Advanced machine learning and natural language techniques have enabled attackers to launch sophisticated and targeted social engineering based attacks. To counter the active attacker issue, researchers have since resorted to proactive methods of detection. Email masquerading using targeted emails to fool the victim is an advanced attack method. However automatic text generation requires controlling the context and coherency of the generated content, which has been identified as an increasingly difficult problem. Our method leverages a hierarchical deep neural model which uses a learned representation of the sentences in the input document to generate structured written emails. We demonstrate the generation of short and targeted text messages using the proposed neural model. The global coherency of the synthesized text is evaluated using an independent study as well as multiple quantitative measures.

1 Introduction

The continuous growth in adversarial learning is a major threat to the field of computer security research. With the advancement in technology, the growing dependency on the Internet has made the common man vulnerable to serious cyber threats like phishing and pharming. Despite considerable research to counter such threats, staggering numbers of individuals and organizations fall prey to targeted social engineering attacks incurring huge financial losses.

Although attackers change their strategies, previous research [1] has shown that electronic mails (emails) are a popular form of attack vector. Emails can be embedded with a variety of malign elements [2] like poisoned URLs to malicious websites, malware attachments as well as executables, documents, image files, etc. Anti-Phishing Working Group (APWG) reports over 270,500[1] unique phishing email campaigns received in the 3rd quarter of 2018, rising from a total of around 233,600[2] unique reports identified in the 4th quarter of 2017. Phishing reports also reveal the consistent rise in phishing attacks targeted towards financial institutions like payment processing firms and banking sectors. The

---

[1] http://docs.apwg.org/reports/apwg_trends_report_q3_2018.pdf
[2] http://docs.apwg.org/reports/apwg_trends_report_q4_2017.pdf

statistics demonstrate how the threat continues getting serious and worse as attackers continue to devise more sophisticated (and maybe more effective) ways of scamming victims.

Innovative and unseen attack vectors can trick pre-trained classification techniques [3], thus placing the victim at risk. In *email masquerading* attacks, an attacker after compromising the email account of an individual can carefully construct a fraudulent email which he sends out to the contacts known to the compromised individual. This has serious implications, because the attacker has gained uninterrupted access to the inbox, outbox and other private details of the compromised person. Thus by exercising caution, the attacker can emulate the content and context of the emails written by the individual and can communicate with his contacts as a legitimate entity, successfully evading detection and causing harm to the victim.

However, construction of the perfect deceptive email requires fine-tuning and manual supervision. While a fake mail constructed manually by an attacker can guarantee a higher chance of success, the process is both time and labor intensive. In contrast, an automated text generator can be trained to synthesize targeted emails much faster and in bulk, thereby increasing the odds of a successful attack. However, the bottleneck in this case, lies in whether the system can generate high quality text, free from common malicious flags like misspellings, incorrect and abusive language, over-usage of action verbs, etc., which can be picked up by a classifier easily. Thus, proactive research in this area of deception based attacks using email masquerading techniques requires further sophisticated experimentation.

Advancement in the field of natural language processing has introduced newer and sophisticated algorithms which enable a machine to learn and generate high-quality textual content on a given context. Grammar based tools like the Dada Engine [4], N-gram language models [5] as well as deep neural learners [6] have been used to study and replicate natural language based attacks. The aim is to facilitate proactive research by predicting newer attacks and reinforce against such unseen yet impending threats.

At the hands of an attacker, language generation techniques can become dangerous tools for deception. With access to proper training data, deep learning neural networks are capable of generating textual content. This property has been leveraged by researchers for a wide variety of natural language generation tasks like tweets [7], poetry [8], [9]. While limited, proactive research has been pursued by using deep learners for generation of fake reviews [6], grammar based techniques [4] as well as simplistic deep networks [10] have been leveraged for email generation. Thus, we can assume that it is not long before phishers and even spammers resort to such techniques to generate newer kinds of malicious attack vectors.

Following a proactive mode of study, we identify the underlying implications of how an automated machine learning technique, here, deep learners can be leveraged to synthesize email bodies for the purpose of email masquerading attacks. We study the effectiveness and practicality of such systems by comparing

an hierarchical deep network with a baseline word prediction model. We also demonstrate the systems' performance using qualitative and quantitative methods. We summarize the following as our key contributions:

– We collect a list of most common signals that set apart a malicious email from its legitimate counterpart as mentioned in previous literature [11], [2], [12] on analyzing phishing emails. We focus on cues that are more prevalent in email bodies for our evaluation. This is necessary to observe the quality of the system generated emails.
– We leverage deep neural networks for generating targeted email bodies. While generation of coherent emails is challenging [10], we propose an hierarchical network which consists of two stages - an architecture which uses a word prediction model to generate probable candidate sentences which are then passed onto a sentence selection model, based on distributed vector representations of the email content, to select the best possible set of sentences. Such a two-staged architecture should be suitable for generating longer content while maintaining coherency.
– We compare the performance of the hierarchical system with a baseline word prediction based model by using multiple quantitative and qualitative metrics. Additionally, we analyze the effectiveness of our system by measuring the *syntactical correctness*, *coherency*, *fluency* and *legitimacy* of the fake emails by conducting a human evaluation.

2   Background

In this section, we provide and discuss a list of cues usually observed in spoofing emails that demarcate such attack vectors from their legitimate counterparts. Highlighting and studying such common signals helped us prepare, process and evaluate our training data as well as the generated emails. After studying common features in malicious emails, we define the goal of our proposed method. We then move on to the detailed description and demonstration of the baseline word prediction model as well as the hierarchical sentence selection model that we have used for our generation task.

2.1   Textual features in spoofing emails

Use of textual features, like presence of common action words, organization names, poisoned links to malicious webpages of financial institutions, grammatical errors, etc., is common in phishing email detection methods [11], [13], [14]. Moreover, researchers have widely studied spam, phishing, spear phishing emails to identify common signs that appear across malicious emails [2], [12]. However, since such signals are certain signs of malign intent an attacker would consciously avoid incorporating these words in a targeted email. Assuming this in mind, our generator should also learn to identify and eliminate overuse of such words.

Therefore, we curate a list of textual cues frequently used in spoofing emails after careful review of phishing email literature [2], [11], [13], [14], [12], [15]. The

list of these textual cues along with examples have been provided in Table 1. Researchers prefer to train their proposed detection methods on publicly available data. Since phishing emails are fairly rare, we base our evaluation on the largest publicly available dataset of malicious emails: Nazario Phishing Corpus [3]. We filter out Base-64 encoded HTML content in the emails and finally use 3,392 fairly clean emails with textual content (>10 words) to extract our spoofing cues.

Table 1: Common Spoofing Cues in Phishing Email body

Feature Types	Examples
Organization Names	(a) *Financial* like eBay, PayPal, Bank of America, Western Union (b) *Government* like Internal Revenue services, United Parcel Service (c) *Software* like Dell, Microsoft, Apple
Action Verbs and Urgency Adverbs	(a) *Action verbs* like click, follow, visit, go, update, apply, submit, confirm, cancel, dispute, enroll, login, answer, reply (b) *Adverbs implying urgency* like today, instantly, straightaway, straight, directly, once, urgently, desperately, immediately, soon, shortly, presently, before, ahead, front
Priciples of Persuasion	(a) *Authority* like an email from a bank asking the victim to update the password of his online account (b) *Social proof* denoted by Emails from the IT department of the target's institution (c) *Distraction* using emails where a target is tempted to click a link in order to receive a prize (d) *Reciprocation* appealing the victim to respond like resetting a password or paying a bill by clicking a link to a fraudulent website
Misspelled Words	*Typographical errors* like Paypl, Bnk Amrica, etc.
Presence of Links	URLs to malicious websites like https://www.maybank2u.com.my, etc.
Use of other languages	*Non-English words* like Aviso Importante de BBVA, societe, Transaktionen

While proposed machine learning systems can detect common cues, these detectors largely depend on historical data. To keep up with advanced reinforcement techniques, a phisher also resorts to employing sophisticated techniques for making their attacks more targeted to increase rate of success. Thus, for social engineering based attacks like email masquerading, spear phishing, or targeted phishing, an attacker may choose to avoid such easily identifiable red flags while generating fake emails. Therefore, in our proactive study, we also refrain from over usage of such spoofing cues in the synthesized emails.

2.2 Task Description

Masquerading as a legitimate entity and sending targeted emails to unknowing victims has become a serious threat to cybersecurity professionals. In our paper, we aim at providing a proactive paradigm to this issue. Given a large dataset of manually written emails, can an automated system learn to emulate the writing style of a human? Is the generated email content coherent and syntactic? Can an individual differentiate between a system generated email and a manually written one?

---

[3] https://monkey.org/~jose/phishing/

We propose a hierarchical model, compare it with a baseline word generation architecture. We study our system performance using multiple quantitative metrics along with qualitative evaluation corroborated by a study using human participants.

2.3 Architecture for Text Generation

Textual content can be considered as a sequence of words and characters placed together to convey meaningful information. In the realm of text generation, deep neural architectures have seen unprecedented success in emulating one's writing when trained on huge amounts of written textual content [6], [8], [16].

Recurrent Neural Networks (RNNs) are capable of retaining information learned from text sequences in "memory", helpful for learning sequential representations of words in the input text. Thus, the trained language model can subsequently generate samples similar in form and context to the input data. We leverage this ability of RNNs for our proactive protection scheme - generation of targeted emails suitable for spear-phishing or masquerading attacks. Moreover, Long Short Term Memory (LSTM) Networks, an improved version of RNN, have proved to be better at handling dependencies in longer sequences of text.

Our architectures use *words* as units for generation [9], [17] by leveraging LSTMs as building blocks for learning the language model. We propose an hierarchical model which merges sentence selection with iterative text generation to generate the best set of human readable samples. We compare the architecture, sampling and generation phases for these two architectures followed by an evaluation of their performance.

Training a Word Prediction Model. Our first model is a straightforward word-based language model built using RNNs [9]. In our implementation, we use Bidirectional LSTMs (Bi-LSTMs) [18] as the network to build the model. Figure 1 shows the overall model for word prediction. We describe the training and generation phases for this baseline generation architecture.

Training Phase. As mentioned before, we make use of Bidirectional LSTMs as our network for text generation model. The input to our text generation system is a sequence of $T-1$ one-hot vectors of words $(w_0, w_1, w_2, ..., w_{T-1})$. At every step t (starting with 0), we feed a word (w_t) into the hidden layer which predicts the next word w_{t+1}. Comparing the current output with desired output (next word in the input text), the model calculates the error in its prediction with the actual value, and accordingly updates the weights of hidden states in order to minimize the error. The perfect model can capture the best sequence of words that constitute the text sequence - in our case, the body of an email.

Generation Phase and Temperature Regulation. During the generation phase, we feed a sequence of starting words as $(W_{seed_0}, W_{seed_1}, W_{seed_2}, ..., W_{seed_N})$ into the trained Bi-LSTM model. This is used to start off the word generation system. Given a word (W_{seed_0}) as input to the model, it selects as output the word (W_1) most likely to occur after W_{seed_0} depending on the conditional probability,
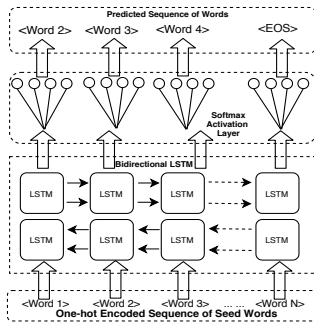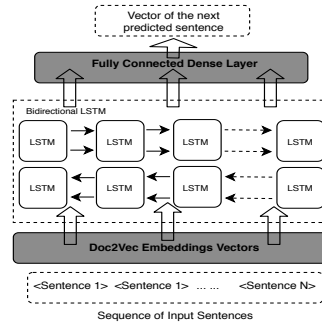
Fig. 1: Word generation model with Bi-LSTMs

Fig. 2: Sentence Selection model using Doc2Vec with Bi-LSTMs

$P(W_1|W_{seed_0})$. Thus, the input sequence of N seeds to the model (Figure 1) can generate a text body of $N + 1$ words.

The final layer of the model is a $softmax$ activation layer which is used for computing the distribution for the next word followed by subsequent sampling i.e., the $softmax$ layer computes the probability of the next word to be chosen. We use $temperature(\tau)$ as the hyper-parameter for selecting our word samples - regulating the parameter τ in Equation 1 encourages or controls the diversity of the generated text. The novelty or eccentricity of the RNN text generative model can be evaluated by varying the $temperature$ parameter between $0 < Temp. \leq 1.0$. While, lower values of τ generate relatively deterministic samples, higher values can make the process more stochastic.

Based on the input set of words, the model builds a probability distribution $P(W_{t+1}|W_{t' \leq t}) = softmax(W_t)$, here $softmax$ normalization with $temperature$ control (τ) is defined as:

$$P(softmax(W_t^j)) = \frac{e^{\frac{W_t^j}{\tau}}}{\sum_{j=1}^{n} e^{\frac{W_t^j}{\tau}}} \qquad (1)$$

Hierarchical Sentence Prediction Model. Controlling global coherency and structure in automated text generation is a non-trivial task. Using, a straightforward word prediction model makes it increasingly difficult to control the quality as well as coherency of the generated text. We propose an hierarchical model consisting of two stages - prediction of sentence candidates followed by sentence selection model. For the sentence selection model, we use Doc2Vec [19] embeddings to learn a vector representation of the email bodies. This is then used to generate the best vector representation of a new sentence given a previous set of candidate sentences. We describe the proposed model in this section.

Sentence Selection using Doc2Vec. Use of embeddings has been regarded as a favorable way to represent context in a piece of text, either in the form

of word phrases, sentences or paragraphs. Doc2Vec [19] can effectively learn the numeric representation of a paragraph or even a document, irrespective of its length. In this model, we use Doc2Vec to learn better representations of the sentences in a piece of text - for example, the body of an email. Figure 2 shows the stages of the proposed sentence selection model. The model starts with training a Doc2Vec model on the entire document. The goal is to predict the next sentence (here, its vector representation) given a sequence of sentences as starting points. As shown in Figure 2, using the trained Doc2Vec model, we vectorize a set of seed sentences and then feed the vectors to the bidirectional LSTM layer in order to learn the model for sentence prediction. This model is added as a second stage to the word generation model (Figure 1) for the final hierarchical structure.

Generation of Sentence Candidates. The first stage deals with the generation of a set of candidate sentences and/or phrases. The input to the whole model is a set of seed sentence sequences. The first stage of the hierarchical network is a trained word prediction model which takes as input the last N words from the input sequence of sentences. The model architecture is same as the one described in Section 2.3. The model iterates over the last N words to generate X candidates for sentence selection. The trained Doc2Vec model generates their vectors which are fed into the sentence selection model described above.

Generation of newer sentences. The seed sentences are fed into the trained Bi-LSTM-based model for sentence selection. This constitutes the second stage of the hierarchical architecture and selects the best sentence vector depending on the input sequences. The generated vector is then compared with the selected candidate sentences - output of stage 1 using a cosine similarity function. The most similar candidate is then produced as output of the hierarchical generation model.

Generation phase and Temperature Regulation. The parameters used and the purpose for temperature regulation during the generation phase using the hierarchical model is similar to the generation phase explained in Section 2.3. The selected best sentence from the generated candidate sentences or phrases are merged with the sequence of input seeds and fed into the model to generate newer sentences.

3   Data Collection and Setup

 A large amount of high quality data is required for text generation. This becomes crucial when automating the composition skill and pattern of an individual. Here, we describe the source and collection steps of the data used in the email body generation task.

3.1   Data Collection

Availability of high quality data is a major challenge in security research. Our purpose is to automatically generate the body of an email capable of deceiving the victim. To produce the best sample of deception, the trained generation
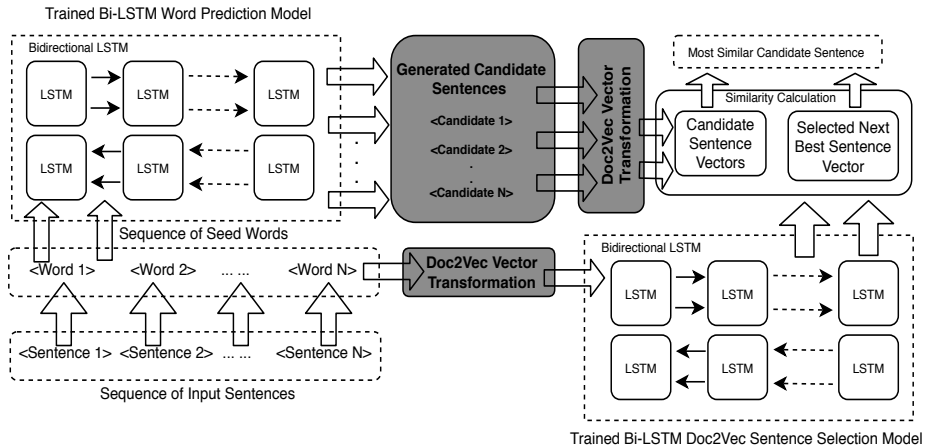
Fig. 3: Word prediction with sentence selection using Bidirectional LSTMs

model must synthesize an email body similar to an email composed by the individual whose writing style it is trying to emulate. Thus, for training our model we make use of the largest publicly available source of legitimate emails, the Enron Corpus [20]. We provide the statistics about our dataset in Table 2. We evaluate the average number of sentences, average vocabulary as well as average number of words in a typical email body. We summarize the step-wise data collection process along with the data preparation and pre-processing steps below.

Building the dataset of legitimate emails. We use the largest publicly available dataset of benign emails - Enron Corpus[4]. Since we aim at synthesizing the writing style of humans, it is necessary to make the process of email masquerading more effective. For building our training and evaluation dataset, we make use of 'legitimate' emails which we collect based on the following assumptions. We assume that the emails that an individual, within Enron, receives are legitimate i.e., they have been identified as benign by the mail server. Also, the emails that the individual sends out are also benign. Also, corpus consists of a large number of emails ranging from spam, deleted advertisements, etc., which is useless for training purposes. Hence, for our purpose, we use emails from two types of email directories - the Inbox (Received folder) and Outbox (Sent folder) of the individuals within Enron corpus.

We summarize the collection process in the following steps: (a) We use the collection of 517,401 emails from the publicly available Enron corpus; (b) We choose the emails that are composed of a minimum of 10 words and comprises of a minimum of one sentence; (c) We also calculate the average statistics of the dataset to determine the generic length, number of words, vocabulary size of the emails we plan to synthesize.

_____

[4] `https://www.cs.cmu.edu/~enron/enron_mail_20150507.tar.gz`

Table 2: Legitimate Data Statistics

Attributes	Values
Dataset size	517,401
Total Number of words	167,692,695
Average number of words per email	324
Total Vocabulary	2,028,671
Average Vocabulary per email	143
Average Number of words per sentence per email	25
Total Number of sentences	6,590,091
Average Number of sentences per email	13

3.2 Assumptions and Data Preprocessing

We assume a typical scenario where the attacker has compromised and gained access to the mail account, all email communication, personal details, email contact list, etc.

Our primary focus is on the automated generation of textual content of the body of an email. Hence, we do not account for the generation or evaluation of header information in an email. Moreover, we also do not consider the generation of email attachments, links or email addresses that maybe present in the body. The goal is to make the synthesized email content more generic to allow inclusion of personalized information like named entities, location names, etc., depending on the communication between the victim and the compromised individual. Also, from the architectural viewpoint, including named entities, personalized details, etc., is unnecessary during training phase and will eventually blow up the vocabulary of our word-based architecture, therefore confusing it. To avoid the inclusion of unnecessary word tokens and to generalize the context, we replace the named entities in the email bodies with the *ent* tags - this includes replacing person names as well as locations (if applicable). We use the Entity Recognizer Module implemented in SpaCy for Python 3.6 for entity tagging and replacement.

In most cases, the body of the Emails contain elements which are of little or no value to our training and evaluation process. We apply the following pre-processing steps to clean our data while making sure not to delete any important information:

- We remove all trailing spaces, newline characters, etc., from the text body
- Replacement of named entities (person, location, etc.) with *ent* tag
- We replace Email addresses in the body with *emailID* tag
- Replacement of URL links with the *link* tag
- Adding the *Start of Text* (<SOT>) and *End of Text* (<EOT>) tags to the email bodies to respectively mark the beginning and end of the content.
- Sanitization of non-ASCII characters from the text
- Removal of HTML text fragments and broken links from text body
- We also lowercase our textual content and remove special characters like #, $, %, , etc.

4 System Evaluation and Analysis

In this section, we describe our experimental setup for the training and generation phases of the baseline and proposed generation models. The evaluation setup for analyzing system performance based on the quantitative and qualitative nature of the generated content as well as their practical implications have also be described.

4.1 Experimental Setup

Implementation. The system was developed in Python 3.6 using Keras (Version 2.2.4) and TensorFlow (Version 1.11.0). In our experimental evaluation, the LSTM network consists of 128 hidden units, since we are using Bi-LSTMs, the total number of states is 256. We consider an unrolling size of 15, i.e., the network looks back up to a sequence of 15 words to predict the next probable word. Among the other hyperparameters, we consider a batch size of 50 with a learning rate 10^{-2}. The aforementioned combination of set of hyperparameters have been chosen based on our empirical evaluation as explained in this section. All our experiments were conducted on a server with 4 Tesla M10 GPUs using CUDA (Version 9.1.85) with a 3.20GHz Xeon CPU E5-2667 and 512 GB of memory.

Evaluation Setup. For our evaluation setup, we use a total dataset of 517,140 emails. We set apart 5% of the dataset for validation during training and the rest was used for training the model. The separate training and validation subsets are crucial for determining the performance of our prediction model on unrelated data. We train the proposed generative models each for 50 epochs with the validation of the model performance after every epoch, finally selecting and saving the model that achieves the best performance on validation data.

Evaluation Metrics. We evaluate the performance of the generative model along three dimensions: (a) a *quantitative* evaluation using a word-gram based measure of overlap and uniqueness between the synthesized and the targeted email bodies as well as model perplexity; (b) a simple *qualitative* evaluation where we inspect the samples generated by the generative word-based and hierarchical models by selecting some samples; and finally, (c) a *human* evaluation which consists of a survey with 6 participants providing feedback on the overall syntax, coherency, and fluency of the synthesized email bodies as compared to their legitimate counterparts.

Quantitative Evaluation. Measure of model *perplexity* is an established method to determine the ability of a language model. We compare the perplexity of the word-based model (Generator$_{word}$) with the proposed hierarchical sentence selection model (Generator$_{sentence}$). A lower perplexity ensures a more stable predictive ability. Model perplexity is defined as:

$$PPL = 2^{\frac{NLL}{T}} \tag{2}$$

Here, NLL refers to negative log likelihood of the model and T is the length of the training sequence for which the model perplexity is measured. A more detailed definition for a word-based generation model:

$$PPL = 2^{-\frac{1}{T}\sum_{t=0}^{T-1} log(P(W_{t+1}|W_0...W_t))} \tag{3}$$

During our second phase of quantitative evaluation, we use a novel metric to observe the semantic coherence across the generated samples. Coherency in an email should account for the adequacy in information between the sentences or even words in the email body [21]. Taking into account the mutual information between a word and its predecessors in the sentence to calculate semantic coherence, we can define the following measure:

$$Coherence = \frac{1}{T}\sum_t (\log p_{fwd}(w_t|w_{t-1}) + \log p_{bwd}(w_{t-1}|w_t)) \tag{4}$$

The mutual information between two words acts as a measure of how likely it is for the words to appear together in the context, thus a higher value of mutual information means better coherence [22]. We take into account both the forward as well as the backward probabilities of occurrence of the bigrams (w_t, w_{t-1}). To control the influence of the length of the generated text, the values are scaled by the total length (i.e., number of words) of the generated sentence/sequence.

Measuring perplexity can provide a false sense of language model's performance - while a low value denotes the model's ability to replicate the input text; it may emulate the characteristics of the input text too much. To measure the overlap between the generated and targeted email text, we consider the ratio of common word-N-grams between the ground-truth (or legitimate) email and its 'fake' counterpart. For evaluation purposes, we select $N = 3$, i.e., trigram overlap.

Table 3: Quantitative evaluation using generated emails

Model	Perplexity	Coherence	Trigram Overlap (%)
Generator$_{word}$	8.97	2.8	43.7
Generator$_{sentence}$	4.59	4.9	66.8

We describe the results of our evaluation on a dataset of 100 generated emails. We calculate our coherence measure based on the language model built using the Brown Corpus[5] available with NLTK package in Python. We use a list of top 1000 most common trigrams in the legitimate emails dataset for the calculation of our trigram overlap. We observe an increased measure of coherence as well as occurence of common trigrams in the emails generated by the proposed sentence based models.

---

[5] https://www.nltk.org/book/ch02.html

Qualitative Evaluation. The most simplistic form of qualitative evaluation is observing the nature of randomly selected samples from the generated text. We have defined common malicious cues in Section 2.1. Drawing motivation from Table 1, we inspect the generated email content for signs of obvious malign intent as well as other prominent features which can be used to evaluate our system performance.

Here, we include two samples each, generated using the word-based model and the hierarchical model at different temperatures for the purpose of our analysis. One noticeable feature of the emails generated by the systems is the number of sentences in the samples. While the more genuine looking samples selected from the outputs of the $\text{Generator}_{sentence}$ model tend to be longer i.e., consists of more than one sentence, the samples from the baseline model (Generator_{word}) are comparably more terse. Also, longer sequences from the proposed model generated at a higher τ tend to include more *ent* tags at random places within the text.

(A) Samples generated by Generator_{word}:

> **A1. Sampling at $\tau = 0.9$:** *ent* Could you give me a moment for why I would appreciate everyone's support . *ent* Best wishes, *ent*
> **A2. Sampling at $\tau = 0.5$:** Hi , I hope you are having a good *ent* interview. *ent*

(B) Samples generated using $\text{Generator}_{sentence}$:

> **B1. Sampling at $\tau = 0.5$:** Yes . We had *ent* them to set up a date . *ent* We are now in London on Monday for Christmas.
> **B2. Sampling at $\tau = 1.0$:** Hi ? *ent* What do you have the steps or makes it a minute. *ent* P.S. You can *ent* email *ent* with changes to a reports that *ent* has delivered several to the members with the office

Human Evaluation. While automated qualitative and quantitative evaluation methods seem believable, these measures just paint half the picture. One of the main contributions of proactive research in cybersecurity involve strengthening humans/individuals using the web and internet. Since humans are considered the weakest link in security research, we evaluate the efficacy of these fake emails by conducting a human evaluation study.

We provide our 6 participants with a survey of 24 email bodies. For each email body, the participants are required to rate the quality of the email content based on three attributes - *syntax*, *coherency* and *fluency*. The scores for each of these attributes are based on a Likert scale $\in$ [1,5]. We also ask each participant to identify whether the email body is legitimate or not - which requires a yes/no response. Of the 24 email bodies, 12 emails are genuine or manually written and the rest are fake or system generated emails. Of the 12 system generated emails, we consider 7 emails which are generated by our sentence selection model ($\text{Generated}_{sentence}$) and 5 emails generated by our baseline word

prediction model (Generated$_{word}$). We choose a balanced ratio to discern the effectiveness of our human participants in detecting fake emails from their genuine counterparts.

Table 4 demonstrates the results of the human evaluation setup. We report the scores for syntax, coherency and fluency, averaged across all participants, on the combined set of system generated emails - Generated$_{all}$. We also report the same on the subset of emails generated by each of the baseline and proposed models - Generator$_{word}$ and Generated$_{sentence}$ respectively, to compare the difference in system performance. We also report the scores on the combined set of system generated emails - Generated$_{all}$. We further compare the statistical significance between the results on the manually written emails (Truth in Table 4) with Generated$_{all}$ - syntax: 4.01 ($p - value = 0.1377$), coherency: 3.31 ($p - value < 10^{-5}$) and fluency: 3.19 ($p - value < 10^{-5}$). We observe that while the difference in syntax scores between the legitimate and generated email bodies are not statistically significant[6], the generated content is still behind in terms of coherency and fluency. However, in contrast, we observe that our human participants tend to have a low detection rate ($\approx$57%) when it comes to generated emails, as observed in Table 4, with the detection rate dropping to approximately 38% in case of emails generated by the proposed sentence selection model. We also include the mean and standard deviation in the number of words ($\overline{W}$, SD_W) and sentences ($\overline{S}$, SD_S) in the generated and the manually written emails used in our survey.

Table 4: Human evaluation results on the automated and true emails. Scores for Syntax (Syn), Coherency (Coh) and Fluency (Flu) vary between [1, 5]. Detection Rate has been reported as a percentage

Email Content	Scores			Detection Rate	$\overline{W}$	SD_W	$\overline{S}$	SD_S
	Syn	Coh	Flu					
Generated$_{word}$	3.63	2.8	2.6	83.33	12	2.88	1.2	0.45
Generated$_{sentence}$	4.29	3.67	3.62	38.09	13	6.21	2	0.81
Generated$_{all}$	4.01	3.31	3.19	56.94	12	5	2	0.78
Truth	4.42	4.56	4.47	75	24	14.8	3	0.94

5   Related Work

The need to counter the active attacker issue has given rise to proactive research methods. While there exists classical techniques for phishing email detection [13], [15], [23], state-of-the-art malicious email detectors fail to detect a sophisticated or targeted attack. Thus, phishing and social engineering attacks

---

[6] $p - value$ calculated using the Unpaired Two Samples Wilcoxon test or Mann Whitney test

have on the rise. Riding on the wave of proactive research, many researchers have delved deeper into the realm of attack generation for different attack vectors but none as serious as emails. While the use of fully automated methods for text generation has been considered, there have not been much investigation into modeling structure-aware emails.

5.1   Natural Language Generation

Use of deep neural networks have enabled in building fully (or partially, with feature engineering,) automated generation of natural language. From the perspective of written text, a substantially trained deep network is capable of emulating the writing style of an individual. This property has been leveraged in natural language research by making deep learners write a wide variety of text Shakespearean Sonnets [9], poetry [8], [24], answer generation [25].

While the use of grammar [4], templates [5], [26] and statistical language based models (e.g. N-grams [27]) are popular, Recurrent Neural Networks (RNNs) have been shown to be a more suitable choice owing to their ability to learn dependencies across the textual context [28]. Long-Short Term Memory (LSTM) networks are more suitable for longer text sequences. However, while a fully automated system seems lucrative - controlling the coherency, topic and structure of the generated text can be quite challenging.

For this purpose, sequence-to-sequence learning using RNNs, is popular for text generation as shown in [29], [30], [31], [32]. However, simple encoder decoder architectures have trouble assigning importance to certain words or context in the text - [33], [29] use attention based models for preserving coherence in the generated text. Other techniques have been used for text generation like deep learning using Markov Models [34], modeling variational auto-encoders [35], generative adversarial networks [36]. However, the literature in text generation have engineered changes at the architecture level. Such an architecture which implements coherency based on characters or words are prone to break as the length of the generated text increases. The architecture proposed in this paper attempts at fixing this issue at a more global level - by implementing coherence in each sentence and then selecting the best sentence to be included in the generated text.

5.2   Attack Generation

Growth in the field of proactive research has become more prominent in an effort to counter active attackers. Phishing is a largely unsolved cyber threat, worsened more by the proliferation in spear-phishing attacks. The ability of the perpetrators to deceive an individual by behaving as a legitimate entity can be automated for widespread social engineering attacks as studied in [37] and [38]. Researchers in [4], [10], [39], look at 'weaponinizing' advanced machine learning techniques to launch sophisticated yet automated targeted attacks. While [4] uses a grammar-based approach for synthetic email generation; Das et. al. [10] uses a more automated deep neural network for email generation, which suffers

from incongruity in the generated context as shown by their evaluation. Other studies in an adversarial setting, which leverage natural language techniques, have been pursued in spreading malicious Twitter messages [40], generating malicious URLs [41], generation of fake reviews [6] as well as text messages [42]. Thus, automated means of synthesizing sophisticated attack vectors reduces the manual labor and provide phishers an opportunity to launch targeted attacks on a much larger scale and magnitude. This in turn increases the chances of succeeding in an attack.

5.3 Emails as Attack Vectors

Emails are the most common and preferred method for social engineering attacks. [2] describes the *modus operandi* and the structure of a common phishing email. Researchers have also delved deeper into the attributes and underlying psychological features that cause phishing and social engineering attacks to be successful in [12], [43], [39]. Techniques for automatic generation of synthetic emails have been discussed in [4], [5], but introducing attributes of deception into legitimate emails is a non-trivial task [10], [44], [37].

6 Discussion and Conclusions

We revisit *two major error trends* observed in the evaluation of our word and character based generation models. *First,* repetitions of tags and words in the generated text body. A sample sentence generated by the word-based language model - "The corres *ent ent ent* Also *ent , ent* I we can operating a gift to ensure, are that extent will is a links are not *ent*" - demonstrates such behavior. While, we hypothesized such a behavior at larger text lengths, the brittleness in a model which uses characters or words as units for text generation can be observed for shorter text sequence generation as well. We believe, that quality of the input text along with the *temperature* (τ) parameter used for sample generation play an important role in such aberrant behavior of the predictive model.

While the RNN model generated text which had 'some' malicious intent in them - the examples shown above are just a few steps from being coherent and congruous. We designed an RNN based text generation system for generating targeted attack emails which is a challenging task in itself and a novel approach to the best of our knowledge. The examples generated however suffer from random strings and grammatical errors. We identify a few areas of improvement for the proposed system - reduction of repetitive content as well as inclusion of more legitimate and phishing examples for analysis and model training. We would also like to experiment with addition of topics and tags like 'bank account', 'paypal', 'password renewal', etc. which may help generate more specific emails. It would be interesting to see how a generative RNN handles topic based email generation problem.

Acknowledgements

References

1. Caputo, D.D., Pfleeger, S.L., Freeman, J.D., Johnson, M.E.: Going spear phishing: Exploring embedded training and awareness. IEEE Security & Privacy **12** (2014) 28–38
2. Drake, C.E., Oliver, J.J., Koontz, E.J.: Anatomy of a phishing email. In: CEAS. (2004)
3. Sommer, R., Paxson, V.: Outside the closed world: On using machine learning for network intrusion detection. In: Security and Privacy (SP), 2010 IEEE Symposium on, IEEE (2010) 305–316
4. Baki, S., Verma, R., Mukherjee, A., Gnawali, O.: Scaling and effectiveness of email masquerade attacks: Exploiting natural language generation. In: Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, ACM (2017) 469–482
5. Chen, Y.N., Rudnicky, A.I.: Two-stage stochastic email synthesizer. In: INLG. (2014) 99–102
6. Yao, Y., Viswanath, B., Cryan, J., Zheng, H., Zhao, B.Y.: Automated crowdturfing attacks and defenses in online review systems. arXiv preprint arXiv:1708.08151 (2017)
7. Sidhaye, P., Cheung, J.C.K.: (Indicative tweet generation: An extractive summarization problem?)
8. Ghazvininejad, M., Shi, X., Choi, Y., Knight, K.: (Generating topical poetry.)
9. Xie, S., Rastogi, R., Chang, M.: (Deep poetry: Word-level and character-level language models for shakespearean sonnet generation)
10. Das, A., Verma, R.: Automated email generation for targeted attacks using natural language. In Lefever, E., Desmet, B., Pauw, G.D., eds.: Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), Paris, France, European Language Resources Association (ELRA) (2018)
11. Verma, R., Shashidhar, N., Hossain, N.: Detecting phishing emails the natural language way. Computer Security–ESORICS 2012 (2012) 824–841
12. Ferreira, A., Chilro, R.: What to phish in a subject? In: International Conference on Financial Cryptography and Data Security, Springer (2017) 597–609
13. Verma, R., Shashidhar, N., Hossain, N.: Two-pronged phish snagging. In: Availability, Reliability and Security (ARES), 2012 Seventh International Conference on, IEEE (2012) 174–179
14. Verma, R., Hossain, N.: Semantic feature selection for text with application to phishing email detection. In: International Conference on Information Security and Cryptology, Springer (2013) 455–468
15. Chandrasekaran, M., Narayanan, K., Upadhyaya, S.: Phishing email detection based on structural properties. In: NYS Cyber Security Conference. Volume 3. (2006)

16. Sutskever, I., Martens, J., Hinton, G.E.: Generating text with recurrent neural networks. In: Proceedings of the 28th International Conference on Machine Learning (ICML-11). (2011) 1017–1024

17. Henderson, M., Thomson, B., Young, S.: Word-based dialog state tracking with recurrent neural networks. In: Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL). (2014) 292–299

18. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional lstm and other neural network architectures. Neural networks : the official journal of the International Neural Network Society **18 5-6** (2005) 602–10

19. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International conference on machine learning. (2014) 1188–1196

20. Enron Corpus: Enron Email Dataset. `https://www.cs.cmu.edu/~enron/enron_mail_20150507.tar.gz` (2015) Online; accessed 13 February 2018.

21. Li, J., Monroe, W., Ritter, A., Galley, M., Gao, J., Jurafsky, D.: Deep reinforcement learning for dialogue generation. arXiv preprint arXiv:1606.01541 (2016)

22. Guo, Z.L.: Using mutual information to identify new features for text documents of various domains. In: Proceedings of the 17th Pacific Asia Conference on Language, Information and Computation. (2003) 372–379

23. Basnet, R.B., Mukkamala, S., Sung, A.H.: Detection of phishing attacks: A machine learning approach. Soft Computing Applications in Industry **226** (2008) 373–383

24. Yi, X., Sun, M., Li, R., Yang, Z.: Chinese poetry generation with a working memory model. arXiv preprint arXiv:1809.04306 (2018)

25. Liu, C., He, S., Liu, K., Zhao, J.: Curriculum learning for natural answer generation. In: IJCAI. (2018) 4223–4229

26. Chen, Y.N., Rudnicky, A.I.: Two-stage stochastic natural language generation for email synthesis by modeling sender style and topic structure. In: INLG. (2014) 152–156

27. Goodfellow, I., Bengio, Y., Courville, A., Bengio, Y.: Deep learning. Volume 1. MIT press Cambridge (2016)

28. Graves, A.: Generating sequences with recurrent neural networks. arXiv preprint arXiv:1308.0850 (2013)

29. Feng, X., Liu, M., Liu, J., Qin, B., Sun, Y., Liu, T.: Topic-to-essay generation with neural networks. In: IJCAI. (2018) 4078–4084

30. Zhang, H., Lan, Y., Guo, J., Xu, J., Cheng, X.: Reinforcing coherence for sequence to sequence model in dialogue generation. In: IJCAI. (2018) 4567–4573

31. Qian, Q., Huang, M., Zhao, H., Xu, J., Zhu, X.: Assigning personality/profile to a chatting machine for coherent conversation generation. In: IJCAI. (2018) 4279–4285

32. Jagfeld, G., Jenne, S., Vu, N.T.: Sequence-to-sequence models for data-to-text natural language generation: Word-vs. character-based processing and output diversity. arXiv preprint arXiv:1810.04864 (2018)

33. Kiddon, C., Zettlemoyer, L., Choi, Y.: Globally coherent text generation with neural checklist models. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. (2016) 329–339

34. Wiseman, S., Shieber, S.M., Rush, A.M.: Learning neural templates for text generation. arXiv preprint arXiv:1808.10122 (2018)

35. Roberts, A., Engel, J., Raffel, C., Hawthorne, C., Eck, D.: A hierarchical latent vector model for learning long-term structure in music. arXiv preprint arXiv:1803.05428 (2018)

36. Press, O., Bar, A., Bogin, B., Berant, J., Wolf, L.: Language generation with recurrent generative adversarial networks without pre-training. arXiv preprint arXiv:1706.01399 (2017)
37. Nohlberg, M., Kowalski, S.: The cycle of deception: a model of social engineering attacks, defenses and victims. In: Second International Symposium on Human Aspects of Information Security and Assurance (HAISA 2008), Plymouth, UK, 8-9 July 2008, University of Plymouth (2008) 1–11
38. Huber, M., Kowalski, S., Nohlberg, M., Tjoa, S.: Towards automating social engineering using social networking sites. In: Computational Science and Engineering, 2009. CSE'09. International Conference on. Volume 3., IEEE (2009) 117–124
39. Giaretta, A., Dragoni, N.: Community targeted spam: A middle ground between general spam and spear phishing. arXiv preprint arXiv:1708.07342 (2017)
40. Seymour, J., Tully, P.: Weaponizing data science for social engineering: Automated e2e spear phishing on twitter. Black Hat USA (2016) 37
41. Bahnsen, A.C., Torroledo, I., Camacho, D., Villegas, S.: Deepphish: Simulating malicious ai. In: 2018 APWG Symposium on Electronic Crime Research (eCrime). (2018) 1–8
42. Shropshire, J.: Natural language processing as a weapon. In: Proceedings of the 13th Pre-ICIS Workshop on Information Security and Privacy. Volume 1. (2018)
43. Sheng, S., Holbrook, M., Kumaraguru, P., Cranor, L.F., Downs, J.: Who falls for phish?: a demographic analysis of phishing susceptibility and effectiveness of interventions. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ACM (2010) 373–382
44. Carvalho, V.R.: Modeling intention in email. Springer (2008)