Agenda Pushing in Email to Thwart Phishing

Hyundong Cho and **Genevieve Bartlett** and **Marjorie Freedman**
Information Sciences Institute
University of Southern California
{jcho, bartlett, mrf}@isi.edu

Abstract

In this work, we draw parallels in automatically responding to emails for combating social-engineering attacks and document-grounded response generation. We lay out the blueprint of our approach and illustrate our reasoning. E-mails are longer than dialogue utterances and often contain multiple intents. To respond to phishing emails, we need to make decisions similar to those for document-grounded responses—deciding what parts of long text to use and how to address each intent to generate a knowledgeable multi-component response that pushes scammers towards agendas. We propose Puppeteer as a promising solution to this end: a hybrid system that uses customizable probabilistic finite state transducers to orchestrate pushing agendas coupled with neural dialogue systems that generate responses to unexpected prompts. We emphasize the need for this system by highlighting each component's strengths and weaknesses and show how they complement each other.

1 Introduction

The Anti-Phishing Working Group observed a doubling of phishing attacks over 2020 with business e-mail compromise scams costing an average of $75,000$ per incident (APWG, 2021). Scammers use these attacks to reach a wide audience of victims and perform targeted attacks on high-value targets. Even when not fully successful, these attacks waste victims' time and resources.

To fight back against scammers, individuals—colloquially called *scambaiters*—have demonstrated that careful engagement with scammers can waste a scammer's time, thus reducing resources for new attacks. Engaging with scammers through dialogue in the form of email also opens up opportunities to push scammers towards actions beneficial for defense and attribution, such as getting scammers to visit a specialized honeypot or divulging information. This information can aid in identifying coordinated, large-scale attack campaigns and help with attack attribution. In this paper we introduce a framework for automating dialogue engagement with scammers and pushing agendas to get scammers to take actions.

Eliciting information from scammers and continuing an email sequence to waste their time presents challenges not addressed by existing dialogue systems. Specifically, this area of automated dialogue is challenging because: 1) email conversations are significantly different from chit-chat conversations: each turn is longer and thus usually contains more information that needs to be incorporated into the response and has multiple intents/requests in a single turn that should be addressed 2) the initial dialogue topics can range greatly and change quickly and a bot must respond appropriately to new topics, goals and questions from the scammer to appear human 3) there is a high cost associated with the scammer recognizing the dialogue is automated as any work put in for trust building is lost if the attacker suspects he/she is talking to a bot and 4) the scammer's agenda is independent of the bot's agenda– thus the bot needs to maintain awareness of its own goals without ignoring the competing goals of the scammer.

Using "canned" responses chosen by following a pre-written script, or performing deep-learning over expected conversation flows for eliciting information are reasonable approaches to address the challenges of keeping responses targeted, topical and persuasive without a lapse in coherency in dialogue. However, such approaches will not meet the second challenge of being robust enough to respond to open dialogue and unexpected scamming intents in a topical and directed manner.

In this paper, we introduce our approach to address all challenges with a modular hybrid dialogue system, Puppeteer. Puppeteer uses multiple Fi-

Proceedings of the 1st Workshop on Document-grounded Dialogue and Conversational Question Answering, pages 113–118

nite State Transducers (FSTs) to push and track multiple agendas in uncooperative dialogue and combines this with a neural dialogue system to keep conversation topics free-flowing and natural sounding while effectively incorporating information provided from the incoming email. We discuss our progress in building our approach and have released our framework for public use[1].

2 The Puppeteer Framework

Eliciting information from SE attackers introduces a niche but important problem space that requires a specialized dialogue system to address the distinct trade-offs and risks involved in engaging with scammers for the purpose of pushing the scammer into certain actions. In this section, we introduce our dialogue framework Puppeteer and discuss how our framework deals with open-ended dialogue, while inserting and tracking progress towards specific desired actions.

First, to carry out and track progress towards specific actions, Puppeteer uses probabilistic finite state transducers (FSTs). The FST approach enables a task-oriented framework for belief tracking and context-specific natural language understanding, which both keep the conversation moving towards specific goals and bolsters accurate interpretation of any extracted information.

Dialogue based on FSTs, however, can be inflexible and brittle in the face of open-ended conversations. An FST-based dialogue approach is not, on its own, appropriate for SMS, social media, and email conversations if the goal is to keep the conversation going without revealing the responder is a bot. To address this, the Puppeteer framework combines its FST approach with deep learning and neural generative approaches. Dialogue generated through the use of pre-trained models is folded in with responses prescribed by any active FSTs in a conversation. The goal in this hybrid approach is to "script" the persuasive dialogue designed to push agendas, while incorporating a more open-ended neural dialogue system to keep the scammer engaged. An illustrative example of this ensemble is shown in Figure 1.

Pushing Agendas with FSTs A Puppeteer agenda is defined by the states and transitions of an FST as well as the cues which indicate that a transition should be taken. The FST for an agenda captures the different pathways a conversation can go when requesting a specific action and responding to possible push-back against requests. At each turn in the conversation, the incoming message is evaluated for all cues in all active agenda FSTs. Additionally, the message is evaluated for a "non-event" for each agenda—the probability that the incoming message does not contain any cues for a particular agenda.

Each cue has a *cue detector* which recognizes when an indicator was found, and provides a confidence value for that decision. These confidence values are then combined with the non-event probability for an agenda and normalized. This normalization must support comparison between different cue detector confidence values and therefore is specific to the set of detectors used for an agenda. For each agenda's FST, Puppeteer tracks the probability distribution across all possible states in the FST as the conversation progresses, retiring agendas as they stall out or complete and adding new agendas based on policy rules dictating when and how to kick off agendas.

Determining when an agenda is complete is also based on thresholding. Ultimately, when the system reaches a high enough confidence the conversation has transitioned an agenda's FST to a terminus state, the agenda is considered complete. By default, Puppeteer does not use fixed thresholds for determining confidence for completion, but instead uses relative probabilities between states and configurable thresholds. This is because longer conversations tend to disperse total probability throughout all states over time. For agendas which are expected to complete over fewer turns, this default can be overridden.

We anticipate a wide range of agendas may be needed. The Puppeteer framework is written in Python and designed to be modular, enabling the easy addition of new agendas (backed by FSTs) and allowing for modular incorporation of nearly any natural language understanding approaches in cue detection. Additionally, defining response actions is extensible to enable differing approaches for response generation. To define a Puppeteer agenda, a user describes the state machine and any custom policy and thresholds in a YAML file. Default behaviors can be easily customized by overriding the appropriate *delegator mixin* class.

Currently, cue detectors are managed by Snips NLU (Coucke et al., 2018). For each transition cue, the user supplies a file of example sentences

[1] https://github.com/STEELISI/Puppeteer

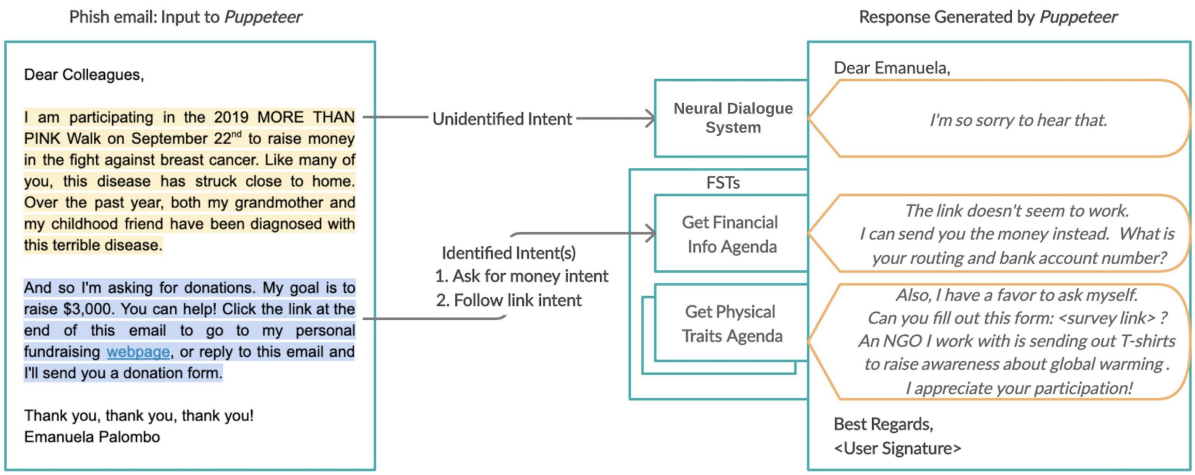

Figure 1: An example of a response generated by a neural dialogue system folded into the script indicatored by the FST to pursue an information collection agenda. Each component complements one another to generate an effective response for eliciting the attacker's information.

or phrases which indicate a transition should be taken, and optionally a file of examples for negative indicators. For example, if an cue detector is looking for text that someone lives in a location, a positive example would be "I live in New York" and a negative example would be "I want to visit New York". These negative examples help filter out false positives. These files are used to create a Snips engine which gives confidence scores on found intents in incoming messages. In practice, we have found most indicators need only 20–40 positive and negative example sentences each as cues are only employed in contexts likely to contain a small set of specific intents and need only to distinguish between "no intent" and the handful of intents in an active agenda. The framework is designed so Snips NLU can be replaced with another NLU approach. To do so, the user must supply a function to Puppeteer which takes in incoming message content and returns a confidence score a particular cue is found in the incoming messages.

Each agenda has a configurable number of associated actions with each state in its FST that can be kicked off any time the probability the conversation has reached policy thresholds for that state and threshold. The default action for all states is to pull a response from a template file, and users can provide additional functions and link these to states in their FST definition for an agenda. In use with our phishing defense system, most agendas have additional actions for states where the scammer has responded with information we pass to other functions of our phishing system such as the attribution module.

Neural Dialogue System: In our current implementation, the neural dialogue system can be chosen to be either a BERT-based question and answering system called Closed-Domain Question Answering (cdQA) or a fine-tuned GPT-2 model.

cdQA offers indirect functionality as a dialogue system by retrieving relevant segments of text to a given query. As its name suggests, it is actually closed-domain in the sense that it only retrieves answers from a given set of source documents, but the source documents can be expanded to accommodate a variety of domains.

Our GPT-2 model is SpolinBot, which can be used as a stand-alone dialogue system. SpolinBot is first fine-tuned with Personachat (Zhang et al., 2018) to adapt to the dialogue domain and then further tuned with SPOLIN to ground its response to the incoming email by learning how to incorporate the "Yes, and" principle of improvisational theatre (Cho and May, 2020). We use training details outlined by Wolf et al. (2019).

Importance of a Hybrid Approach The importance of correctly integrating the components becomes evident by observing the shortcomings of each component when used in isolation. Figure 2 demonstrates components in isolation. The FST approach is stilted in pushing an agenda as it is limited to responses for agendas deemed relevant to the conversation which does not directly address questions. The neural dialogue systems cannot push an agenda, but respond to the prompt.

In contrast, Figure 1 demonstrates the strengths of each component when they are ideally combined together to generate an effective response.

Putting them together: For each paragraph from the email other than the header and the signa-

SCAMMER: Are you interested in a job? Let's set up a meeting.

FST: Where are you based out of? What time works for you?

NDS-YESAND: Yes, I've been looking for one.

NDS-QNA: I can see what times work.

Figure 2: Examples that highlight the weaknesses of individual components. The FST approach is stilted in pushing an agenda as it does not address the question posed by the scammer. The neural dialogue system (NDS) fails to respond to specific tasks.

ture, Puppeteer currently consults the cdQA component for questions and the *yes-and bot* for non-question text and text which has no indicators for any agenda. As shown in Figure 1, the responses from the neural dialogue component and the Puppeteer agendas are naively appended in order of the parts of the email that they respond to. However, it may often be the case that some parts of the email do not necessarily need a response. Improving how and when components are called on for responses and how these responses are combined is an ongoing effort. So far, empirical results show our current combining approach does relatively well on short prompts, but this analysis is particularly challenging due to the lack of automatic evaluation metrics for neural dialogue systems and the large variance of resulting models based on different training data.

3 Related Work

Social engineering (SE) is the act of getting users to compromise information systems. Contrary to technical attacks directly on network and computer systems, SE attacks target humans with access to information and manipulate these target users to divulge confidential information (Krombholz et al., 2015). Phishing is a specific type of social engineering attack in which targets are contacted through digital channels such as e-mail, SMS or social media to lure individuals into providing sensitive data such as personally identifiable information, system log in credentials or organization details (Hong, 2012). Our work focuses on generating dialogue to engage such scammers over one or more of these digital, text-based channels.

Most research efforts addressing SE look at detection (e.g. Basnet et al. (2008); Chen et al. (2014); Singh et al. (2015)) and defending against such attacks by dropping or otherwise terminating such attacks (e.g. Chaudhry et al. (2016); Gragg (2003); Chandra et al. (2015)). An anti-phishing project by Netsafe[2] picks a curated personality and uses automated email responses to waste the attacker's time as much as possible, but its not open-sourced and little is known about how it works. Our system is similar to Netsafe's project in that it is focused on *actively engaging* scammers through automated dialogue, but Puppeteer also *pushes scammers towards actions* favorable for attribution and defense. We rely on separate detection methods to identify messages and senders the Puppeteer dialogue system should engage.

Only recently have research efforts looked at using automated text-based dialogue to respond to scammers. Li et al. (2019) leverage intent and semantic labels in non-collaborative dialogue corpora to distinguish on-task and off-task dialogue and therefore enhance human evaluation scores for engagement and coherence. We aim to achieve a similar objective with the additional goal of pushing a range of agendas and responding appropriately and topically over a broad range of open dialogue. Hobbyists and commercial developers also have looked at automatic responses to scammers. These efforts are interactive spoken-word approaches that detect silence in conversation and interject prerecorded non sequiturs to waste a scam caller's time (Oberhaus, 2018; TelTech, 2020). While one of the goals of our work is to waste scammer time, Puppeteer performs natural language understanding to engage scammers at a deeper level and push agendas with the ultimate goal of pushing scammers into actions which aid attribution.

Our hybrid system is inspired by a large body of existing work in dialogue systems. Hudson and Newell (1992) propose probabilistic FSTs for managing dialogue under uncertainty, while many dialogue systems incorporate FSTs for management functionality in spoken dialogue systems (Pietquin and Dutoit, 2003; Chu et al., 2005; Sontag, 2006; Hori et al., 2009). Recent interests in large pre-trained language models based on Transformers and open-domain question answering systems paved the way for our neural network approaches to be used as open-domain dialogue sys-

[2] https://rescam.org

tems, such as GPT-2 or DrQA (Vaswani et al., 2017; Devlin et al., 2019; Liu et al., 2019; Radford et al., 2018; Chen et al., 2017; Farias et al., 2019). The novelty of Puppeteer is in the combination of these two approaches to address the unique challenges of system-scammer dialogue.

4 Conclusion

In this paper we introduced email response generation for phishing as a challenging dialogue domain. Our approach draws on similarities with document-grounded response generation. As a first step to address the challenges of automating phishing response, we proposed Puppeteer and made it publicly available. Puppeteer's modular architecture makes it easy to augment or replace its components to tackle individual challenges. These components complement one another in generating suitable responses for engaging scammers and inserting agendas, but it remains an open problem to seamlessly combine response components into a composed email response.

This material is based on research sponsored by the AFRL and DARPA under agreement number FA8650-18-C-7878. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the AFRL, DARPA, or the U.S. Government.

References

APWG. 2021. Phishing activity trends report 4th quarter 2020.

Ram Basnet, Srinivas Mukkamala, and Andrew H Sung. 2008. Detection of phishing attacks: A machine learning approach. In *Soft Computing Applications in Industry*, pages 373–383. Springer.

J Vijaya Chandra, Narasimham Challa, and Sai Kiran Pasupuleti. 2015. Intelligence based defense system to protect from advanced persistent threat by means of social engineering on social cloud platform. *Indian Journal of Science and Technology*, 8(28):1.

Junaid Ahsenali Chaudhry, Shafique Ahmad Chaudhry, and Robert G Rittenhouse. 2016. Phishing attacks and defenses. *International Journal of Security and Its Applications*, 10(1):247–256.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*.

Yi-Shin Chen, Yi-Hsuan Yu, Huei-Sin Liu, and Pang-Chieh Wang. 2014. Detect phishing by checking content consistency. In *Proceedings of the 2014 IEEE 15th International Conference on Information Reuse and Integration (IEEE IRI 2014)*, pages 109–119. IEEE.

Hyundong Cho and Jonathan May. 2020. Grounding conversations with improvised dialogues. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2398–2413, Online. Association for Computational Linguistics.

Shiu-Wah Chu, Ian O'Neill, Philip Hanna, and Michael McTear. 2005. An approach to multi-strategy dialogue management. In *Ninth European Conference on Speech Communication and Technology*.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*, pages 12–16.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

André Farias, Félix Mikaeliean, Matyas Amrouche, Théo Nazon, and Olivier Sans. 2019. cdqa: Closed domain question answering. `https://github.com/cdqa-suite/cdQA`.

David Gragg. 2003. A multi-level defense against social engineering. *SANS Reading Room*, 13.

Jason Hong. 2012. The state of phishing attacks. *Commun. ACM*, 55(1):74–81.

Chiori Hori, Kiyonori Ohtake, Teruhisa Misu, Hideki Kashioka, and Satoshi Nakamura. 2009. Statistical dialog management applied to wfst-based dialog systems. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4793–4796. IEEE.

Scott E Hudson and Gary L Newell. 1992. Probabilistic state machines: dialog management for inputs with uncertainty. In *Proceedings of the 5th annual ACM symposium on User interface software and technology*, pages 199–208. ACM.

Katharina Krombholz, Heidelinde Hobel, Markus Huber, and Edgar Weippl. 2015. Advanced social engineering attacks. *Journal of Information Security and applications*, 22:113–122.

Yu Li, Kun Qian, Weiyan Shi, and Zhou Yu. 2019. End-to-end trainable non-collaborative dialog system. *arXiv preprint arXiv:1911.10742*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Daniel Oberhaus. 2018. The story of lenny, the internet's favorite telemarketing troll.

Olivier Pietquin and Thierry Dutoit. 2003. Aided design of finite-state dialogue management systems. In *2003 International Conference on Multimedia and Expo. ICME'03. Proceedings (Cat. No. 03TH8698)*, volume 3, pages III–545. IEEE.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*.

Priyanka Singh, Yogendra PS Maravi, and Sanjeev Sharma. 2015. Phishing websites detection through supervised learning networks. In *2015 International Conference on Computing and Communications Technologies (ICCCT)*, pages 61–65. IEEE.

Daniel Sonntag. 2006. Towards combining finite-state, ontologies, and data driven approaches to dialogue management for multimodal question answering. In *Proceedings of the 5th Slovenian First International Language Technology Conference (IS-LTC 2006)*.

TelTech. 2020. Robokiller, the app that stops spam calls forever.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. *arXiv preprint arXiv:1901.08149*.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213.