EECS 598 Project Final Report
Due Dec 09 by 11:59pm

Do June Min (dojmin@umich.edu), Spencer Vagg (spencerv@umich.edu)

December 9, 2020

1 Introduction

Privacy Policies can be very lengthy and confusing documents for the average user to sift through. If users have a question about a company's privacy policy, they must both find the relevant sections in the policy document for that question and then decipher what those segments mean. This can cause users to be misinformed about what is happening with their information when they use a product or service. In this work, we present a Question Answering framework that will take in any questions that a user may have related to a company or organization's privacy policy and then provide an abstractive answer in response to the question. Our system is a two-stage framework consisting of a retrieval module for extracting relevant segments from the policy document given a user query, and a generation module which takes the retrieved segments as input and outputs an answer to the user query. Our dataset lacks annotatation for supervised training, and we use self-critical reinforcement learning to train our generation system. We illustrate our approach's difference to other approaches in Table 1.

2 Related Work

There are several works that study how natural language processing can be used to help users read and understand privacy policies. Harkous et al use word embeddings and convolutional neural networks to build Polisis, a framework for annotating privacy policies with labels and categories aimed at enhancing users' understanding [1]. They also use the framework to create Pribot, a chatbot for answering user queries about a given privacy policy. It is notable that although Pribot allows free-form user queries, the returned answers are always chosen from the privacy policy segments. Similarly, Ravichander et al proposes a model for QA on privacy policies, also returning policy segments as answers to user queries [2]. Recently, Ahmad et al released a dataset and a baseline model for span-extraction QA [3]. Ahmad et al's dataset is based on the OPP-115 corpus, a collection of website privacy policies, compiled by Wilson et al [4]. Although Ahmad et al's approach moves beyond returning whole policy sentences as answers to user queries, we note that their approach is still extractive, since only observed spans in the policy can be returned as answer. On the other hand, our module is abstractive by design, and has the capacity to generate unseen yet relevant and informative text in respose to user queries.

User Query: Is my information shared with others?			
	Sentence Extraction (Ravichander et al)	Span Extraction (Ahmad et al)	Generative (Our model)
Gold Truth/Ideal Output	Sometimes we send offers to selected groups of Amazon.com customers on behalf of other businesses. When we do this, **we do not give that business your name and address**. If you do not want to receive such offers, [Omitted]	**we do not give that business your name and address**	We do not share your information with other parties.

Table 1: Comparison of different approaches to Question Answering on Privacy Policy documents on a PolicyQA dataset example. Boldface indicates relevant span in the sentence-level extraction and the span-level extraction.

3 Methodology

3.1 Dataset and Data Processing

We use PrivacyQA by Ravichander et al to train our evidence extraction and answer generation module. [2]. PrivacyQA contains a collection of 1,750 user-generated queries for privacy policies of mobile phone apps, paired with annotations by legal experts. In total there are 35 different policies that are included in the dataset, with the data pre-split into a train and test set. The train set consists of 27 privacy policies and 1,350 questions and the test set consists of 8 new, unseen privacy policies and 400 questions. Each segment of the privacy policy is compared to each question and deemed as a relevant or irrelevant segment for answering that question. Each of these relevant/irrelevant labels were looked over by multiple people.

Our data preprocessing step is relatively simple as PrivacyQA is a pre-compiled dataset. The dataset consists of csv files which contain each query, policy segment, and relevance score. These files are already split into train and testing splits for us, with no overlap in policies between the two files. We read in the csv files and used huggingface's pretrained BertTokenizer and XLNet Tokenizer for the extraction module and a GPT2Tokenizer for the generation module.

3.2 Models

We adopt a two stage approach to generate answers to user queries. In the first stage, a user query is fed to an evidence extraction/retreival model that returns a list of relevant segments. Then, the query and the set of retrieved evidence segments are fed to the generator module that outputs an answer to the query, based on the retrieved evidence.

3.2.1 Stage 1: Evidence Extraction Module

For our retrieval model, we tried two approaches: (1) binary classification with single policy sentences and (2) binary classification with multiple policy sentences using joint sentence prediction.

We originally used bert-base-uncased as the underlying model for both of these approaches. Having found that the single policy BERT outperformed the multiple policy BERT, we decided to try and increase our performance on this single policy task by also testing with a xlnet-base-cased model. This model is larger than the BERT model and outperforms it on many tasks, so we thought it would be a good model to try and increase performance. We are using pretrained transformer networks, since (masked) language model training is thought to be beneficial for performances in many downstream tasks.

Regarding the choice of single sentence vs joint sentence prediction, the single sentence framework is the default design, and is also the one suggested in the PrivacyQA paper as a baseline. It frames evidence retrieval as a binary classification between query and a single sentence. The motivation behind joint sentence prediction was to incorporate context into the retrieval step, since the single prediction ignores the context from other sentences in the document. The idea was to set a context size for n sentences, and use the transformer model to jointly predict the relevance score for each of the sentences.

3.2.2 Stage 2: Answer Generation Module

Our generator module is inspired by the Summary Loop. It consists of the generator model (gpt-2), fluency model, and the scoring components for self-critical reinforcement learning, but we replaced the original coverage model with our own variation. We adopt the unsupervised learning approach of the Summary Loop framework to train our generator model.

Unlike abstractive summarization, which involves two texts (source and summary), our abstractive QA framework involves three text inputs: query, evidence, and answer. Thus, we want our answer-check model to be able to reliably evaluate how the generated answer responds to the question, given the information from the evidence. To this end, we operationally conceive of an answer as an abstractive summary of the evidence emphasizing information which is relevant to the question. Following the coverage model from the Summary Loop, we want to mask the tokens from the evidence sentences so that the "important" tokens are masked. However, in our case the importance is decided by its relevance to the question. Thus, we propose a two-step procedure to mask the evidence text. First, we train a masked language model with a masked Query concatenated with an unmasked evidence text as input. The query mask can be chosen by an unsupervised keyphrase extraction algorithm, such as tf-idf. The intuition is that the network will be able to learn how to attend to the evidence text for recovering important tokens in the question, and we can use its attention weights as a proxy for which evidence tokens are relevant for a given query. Then, we will use a separate masked language model which takes the concatenation of an unmasked query, a masked evidence text (masked from the step 1's network), and the generated answer.

We additionally introduce two additional scoring components to encourage the generator to include direct responses to user query in its output. For instance, a direct response to a "Does the company - " type question would be "Yes/No". To this end, we utilize two pretrained transformer-based models to implement the answer content scorer and the answer form scorer. The former measures the semantic similarity between the first segment (first 30 tokens) of the generated output and the retrieved text, by measuring the cosine similarity between their sentence embeddings. The sentence embeddings are computed using the sentence transformer model [5]. On the other hand,

the answer form scorer's object is to score how compatible the user query and the first segment of the generated answer are, as a question and answer pair. We use a BERT-based model trained on span-extraction QA datasets such as, SQuAD, RACE, and etc (https://huggingface.co/iarfmoose/bert-base-cased-qa-evaluator). We include further details about the training process in the appendix.

4 Experiments

Given that we have two different models that we are evaluating here, we will need various different metrics to test their aptitude for the task. For the first model, which identifies whether the given parts of the policy correspond to the answer, we will use precision, recall, and f-score metrics. These metrics are standard when it comes to binary prediction questions. For the second model, we evaluate the summarized answers using simple, human annotations and compare them across our model and the baseline models.

4.1 Results

4.1.1 Evidence Extraction Module

Our single-prediction BERT retrieval model is designed similarly to the PrivacyQA paper's baseline, and achieves similar results (precision = 44.76, recall = 40.19, F1 = 37.58, Note: here each reported metric is the maximum value from all six annotator's reference). Interestingly, our joint prediction model with context augmentation (context size = 5) performs worse than the single prediction model. One guess to why this might be happening is each sentence in policy documents are carefully crafted by legal experts and already contains enough information, and thus the added task of learning to utilize contextual information degrades the resulting performance. We also found our XLNet results to be surprising, as they were also lower than the single BERT model (precision = 41.84, recall = 35.78, F1 = 34.68). We think this may be due to hyperparameter tuning, but more work would have to be done to look into it.

4.1.2 Answer Generation Module

Since our dataset lacks annotation of ground truth answers for each user query-evidence pair, the authors of this paper conducted a human evalutaion on a set of 48 examples. Each author independently annotated the examples with the following three criteria, each of which have binary 0,1 labels:

- Answeredness: Does the generated output directly answer the user query? Score 1 if and only if the first sentence of the answer is in response to the question.
- Readability: Is the generated output readable and grammatical? Score 0 if any part of the answer is ungrammatical or incoherent.
- Faithfulness: Does the answer only contain facts/information that is consistent with the retrieved evidence? Score 0 if facts or statements not included or inconsistent with the evidence is in the answer.

Also, we compute Rouge-l to see how much overlap exists between the generated answer and the retrieved evidence. For baseline, we consider two t5-based summarizers. The T5 model is known

as a powerful model able to adapt to many different problems, so we thought this would be a good comparison against our model. [6]. One model sees both the question and evidence in the input (T5 Q + E), and the other baseline is treated as an evidence summarizer, only having the retrieved evidence in its input (T5 E).

Criteria/Model	Our Model	T5 Q + E	T5 E	Cohen's Kappa
Answeredness	**0.34**	0.32	0.31	0.52
Readability	0.63	**0.84**	0.82	0.22
Faithfulness	**0.96**	0.82	0.86	0.48
Rouge-l	0.18	**0.08**	0.09	-

Table 2: Human Evaluation on a set of 48 Question-Evidence Pairs. Q stands for Query, and E for Evidence.

From the table, we see that our model performs best for answeredness and faithfulness metrics. However, we note that still the performance on answeredness is quite low, indicating that training the generator model to produce direct responses is difficult. Moreover, we note that our model has converged to a mode of mostly copying segments from the evidence, as shown in our model's high level of faithfulness and Rouge-l score.

5 Conclusion and Future Work

Abstractive QA is a relatively new yet promising and rapidly developing area of research. In this project, we implemented an abstractive QA system for answering user queries for privacy policy documents for mobile applications. The biggest obstacle in developing the generation module was the lack of annotated data, both for training of the parameters and the testing and validation of our models. A promising line of further work is to investigate how transfer learning from other Question Answering datasets can be leveraged to combat this problem. Moreover, we believe that systemic grid search of scoring parameters (importance weights) and in-domain training of component models will lead to better performance of the final generator model.

References

[1] Hamza Harkous, Kassem Fawaz, Rémi Lebret, Florian Schaub, Kang G. Shin, and Karl Aberer. Polisis: Automated analysis and presentation of privacy policies using deep learning, 2018.

[2] Abhilasha Ravichander, Alan W Black, Shomir Wilson, Thomas Norton, and Norman Sadeh. Question answering for privacy policies: Combining computational and legal perspectives, 2019.

[3] Wasi Uddin Ahmad, Jianfeng Chi, Yuan Tian, and Kai-Wei Chang. Policyqa: A reading comprehension dataset for privacy policies, 2020.

[4] Shomir Wilson, F. Schaub, A. A. Dara, Frederick Liu, Sushain Cherivirala, P. Leon, M. S. Andersen, Sebastian Zimmeck, Kanthashree Mysore Sathyendra, N. Russell, T. Norton, E. Hovy, J. Reidenberg, and N. Sadeh. The creation and analysis of a website privacy policy corpus. In *ACL*, 2016.

[5] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019.

[6] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2020.

A Appendix

Here, we provide information on the unperceived training of our generator module and two sample outputs, each showcasing desirable property or failure mode of our trained model.

A.1 Generator Module Training

For self-critical reinforcement training of the generator model, we use the base GPT-2 model and initialize it with the provided weight. The reward function is computed as the sum of the scores from the 4 main scoring functions (coverage, answer form, answer content, and fluency), plus the "guardrails" defined in the Summary Loop framework. We use the following weights to compute total score:

- Coverage/Answer Check: 10.0
- Answer Content: 5.0
- Answer Form: 5.0
- Fluency: 2.0
- Length Penalty: 2.0
- Repetition Penalty: 2.0
- Pattern Penalty: 5.0

The training progress for the combined total score and the four main scorers are shown in Figures 1-5. Due to constraints in resource usage, we ran the training loop for approximately 36 hours before termination. We note that although the answer form and the coverage scores increase over time, there remains significant noise and fluctuation in the scores.

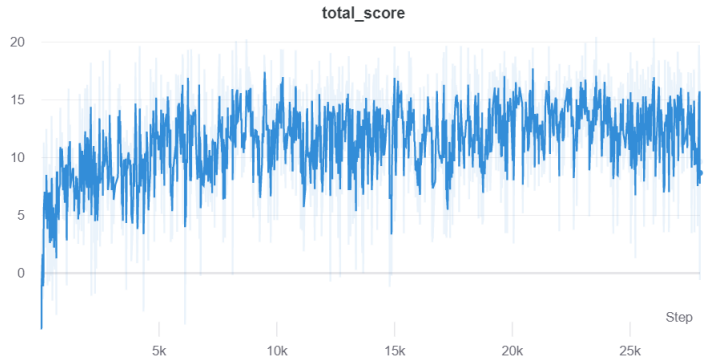

Figure 1: Total Score

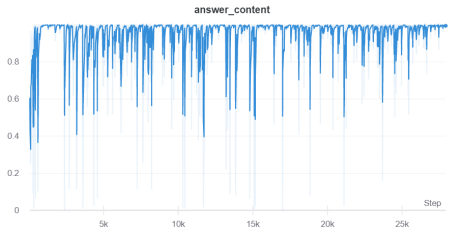

Figure 2: Answer Content Score

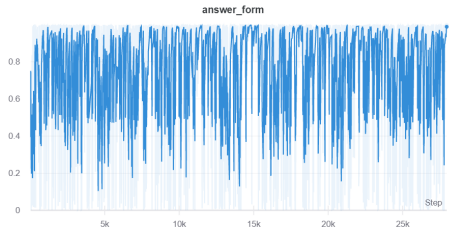

Figure 3: Answer Form Score

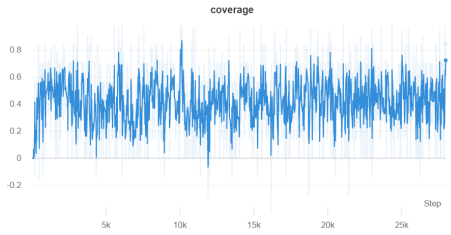

Figure 4: Coverage/Answer Check Score

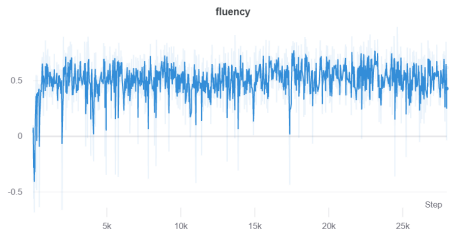

Figure 5: Fluency Score

A.2 Example of Coverage Output

Below is an example of how our model covers evidence given a query. It identifies relevant words/stretches such as send such information, data, and address, but it still misses some tokens such as location.

Query: Does it store my gps data?

Evidence: We automatically collect some information from your computer or Device when you visit TripAdvisor. For example, we will collect session data, including your IP address, Web browser software, and referring website. In addition, we may collect information about your location if you have instructed your Device to send such information to the Application via the privacy settings on that Device, or if you have uploaded photos tagged with location information. We may use the location information collected from your Device or photos to enhance your use of the Application by providing you with relevant content and contextual advertising. For example, we may use your location to show you reviews of hotels or restaurants near you when you are traveling. You can change the privacy settings of your Device at any time, in order to turn off the functionality to share location information with the Application and/or the functionality to tag your photos with location information. Please note that turning off location sharing may affect certain features of our App.

Masked Tokens: data, address, software, website, send such information, information

A.3 Generator Samples

In the following examples, the provided evidence segments are gold truth annotations, rather than Stage 1-module extracted segments.

Example with Good Model Output

Query: does viber log messages?
Evidence: Before you start sharing good vibes, its important you understand that by using our Services, you allow us to collect, use, disclose, and retain your personal information and other information but we will never read or listen to the content you are sharing privately. First of all, we want you to be assured that we do not read or listen to the content of your messages and/or calls made privately via Viber and we do not store those messages once they have been delivered to their destination (which on average takes less than one second). [Omitted]
Our Model: You allow us to collect, use, disclose, and retain our personal information and other information but we will never read or listen to the content you are sharing privately. First of all we want you to trust that we do not read or listen to
T5 Q + E: viber does not read or listen to the content of private messages and calls. If
T5 E: if you don't receive a message within 2 weeks, it will be deleted from

Example with Bad Model Output

Query: how long will you have my information for?
Evidence: You can access, update and delete your Personal Information you provided to us, as described below, by managing this information through your online account or sending us an email at privacy@groupon.com. You can update or remove Financial Account Information [Omitted]
Our Model: You can access, update and delete your Personal Information You provided to us, as described below, by managing this information through your online account or sending us an email at privacy@groupon.com. You can update or remove Financial Information as
T5 Q + E: if you want to deactivate your Groupon account, please contact us.
T5 E: groupon will retain your personal information for as long as your account is active. we will