Hark: A Deep Learning System for Navigating Privacy Feedback at Scale

Hamza Harkous†‡, Sai Teja Peddinti†‡, Rishabh Khandelwal†‡*, Animesh Srivastava‡, and Nina Taft‡
†Google, ‡University of Wisconsin-Madison

Abstract—Integrating user feedback is one of the pillars for building successful products. However, this feedback is generally collected in an unstructured free-text form, which is challenging to understand at scale. This is particularly demanding in the privacy domain due to the nuances associated with the concept and the limited existing solutions. In this work, we present Hark, a system for discovering and summarizing privacy-related feedback at scale. Hark automates the entire process of summarizing privacy feedback, starting from unstructured text and resulting in a hierarchy of high-level privacy themes and fine-grained issues within each theme, along with representative reviews for each issue. At the core of Hark is a set of new deep learning models trained on different tasks, such as privacy feedback classification, privacy issues generation, and high-level theme creation. We illustrate Hark’s efficacy on a corpus of 626M Google Play reviews. Out of this corpus, our privacy feedback classifier extracts 6M privacy-related reviews (with an AUC-ROC of 0.92). With three annotation studies, we show that Hark’s generated issues are of high accuracy and coverage and that the theme titles are of high quality. We illustrate Hark’s capabilities by presenting high-level insights from 1.3M Android apps.

I. INTRODUCTION

Recently, application stores, such as the Android Play Store and the iOS App Store, have added features to improve the developer-to-user communication of apps’ privacy practices [1, 4]. These include mechanisms to clearly state what data is collected/shared and the purposes behind that. However, few advances have been made towards user-to-developer communication of privacy-related concerns. Users mostly communicate their views and needs via app reviews. Extracting, processing, and understanding privacy-related reviews remains a highly underutilized opportunity despite initial assessments showing that, when such reviews are uncovered, developers take concrete steps to update their apps [34].

For a system that would help developers sift through privacy reviews in meaningful ways, we posit that there are three main requirements:

- **topical diversity**: It should have a high coverage of the various aspects in the privacy domain, regardless of the way they are linguistically expressed.
- **glanceability**: It should allow developers to understand the gist of the topics discussed without having to read all reviews.
- **navigability**: It should enable developers to have a high level understanding with the ability to dive deep into the issues.

Previous attempts at analyzing privacy reviews [5, 7, 32, 34] have not built classifiers with **topical diversity** as a goal. They primarily relied on keyword-based sampling of training data, thus restricting the privacy issues users discuss to a set of predefined wordings. Moreover, these approaches did not go further beyond the classification step. Hence, they fail at creating a structure out of the reviews. Even when considering the broader work on analyzing app reviews [11, 18, 36], we notice that these fall short at providing glanceable summaries of the topics users raise. They are often restricted to extracting verbatim keywords or phrases from users’ reviews [11]. The ultimate result achieved there is a set of clustered reviews, without an explainable common theme for each cluster. This results in a lot of manual work for navigating through reviews by reading them, finding issues users discuss, and understanding the high-level themes summarizing users’ privacy feedback.

Despite these shortcomings, previous works have shown that, when developers are made aware of privacy reviews, they do carry out related updates [34]. Similar results where observed when nudging developers to reduce unnecessary permissions [38]. Such actions are further motivated by the correlation between low ratings and negative privacy reviews [5].

In this work, we present Hark, a system for end-to-end retrieval and analysis of privacy-related feedback, which is designed to satisfy the above requirements. Hark leverages state of the art techniques in Natural Language Processing (NLP) for rethinking how privacy reviews are presented to developers at multiple levels of abstraction.

To satisfy the **topical diversity** requirement, we developed Hark’s privacy feedback classifier, by leveraging the Natural Language Inference (NLI) task [30] to ensure that our training data has a high coverage of the privacy concepts defined in widely-used privacy taxonomies [43, 50]. Our principled approach to designing this classifier results in an AUC-ROC of 0.92, significantly outperforming baseline models.

To cover the **glanceability** requirement, Hark includes an issue generation model that takes the privacy reviews output by the classifier and assigns meaningful, fine-grained issues to each review. Unlike the traditional review analysis literature [10, 21], our model is not restricted to predetermined topics. It takes an abstractive labeling approach, generating issue tags that distill long informal reviews (even those containing rants) into simple easy to grasp issues (e.g., Unwanted Password Sharing or Personal Address Deletion). These issues are dynamically
generated, covering both commonly occurring issues as well as newly emerging ones. Through two annotation studies with 600 test reviews, we show how our model generates issues whose accuracy reaches 96% (28% higher than the baseline) and whose coverage reaches 93% (50% higher than the baseline) when 5 out of 7 annotators agree.

Next, Hark’s outputs are designed with navigability as a main goal. Towards that, Hark includes a theme creation component, which takes the issues across all reviews and groups them in clusters. These represent high-level themes, each containing a set of related fine-grained issues. Importantly, Hark includes a generative model that assigns a succinct title for each theme (e.g., Sharing Concerns or Data Deletion). This eliminates the manual work required to interpret clusters. Through an annotation study with 600 groups of issues, we show that our model produces titles which are judged to be of high quality in 92% of the cases where 5 out of 7 annotators agree (20% higher than the baseline).

To further facilitate navigating this hierarchy, Hark includes a classifier that dissects issues and themes across 28 emotions, such as joy, anger, annoyance and confusion. We show how this way of navigating the hierarchy can provide new insights into the topics users discuss. Finally, Hark’s feedback quality scoring model allows ranking the representative quotes per issue. This allows developers to understand an issue in more detail, and in the user’s voice, without having to read numerous reviews. Overall, Hark enables developers to explore this feedback from a high level perspective (themes), and then drill down into successively more details (fine-grained issues and then high quality example reviews annotated with emotions).

To illustrate Hark’s capabilities, we apply it to a large dataset of 626M million publicly visible reviews covering 1.3 million apps. Our classifiers extract over 6M privacy-related reviews from that set. We further illustrate Hark’s ability to satisfy the above requirements by providing an example analysis.

We scope this paper around building the underlying framework and methodology for understanding privacy feedback at scale. We leave the use of Hark for conducting deep studies into the identified privacy issues for a future work.

II. BACKGROUND AND RELATED WORK

In this section, we introduce the necessary background from natural language processing that we build on for the coming sections. We also discuss related works around analyzing user feedback, with a particular focus on the privacy domain.

A. Advances in Unsupervised Pretraining

Our system involves two main types of tasks: text classification and text generation. Classification tasks allow assigning one or more predefined tags to a given input. Examples include email spam classification or language detection. Generation tasks output free-form text for a given input. Examples include abstractive document summarization or machine translation.

The field of Natural Language Processing (NLP) has seen significant improvements on these tasks, bolstered by advances in unsupervised pretraining and transfer learning. By pretraining a model on massive corpora, the model can develop general-purpose knowledge that can be transferred to downstream tasks. This has been shown to be highly effective by models such as BERT [13], which was pretrained on predicting the masked word in a sequence (a.k.a masked language modeling) and GPT-2 [39], which was pretrained on predicting the next word in a sequence (a.k.a causal language modeling). Once pretrained, the model can be finetuned on a downstream task. This is commonly done by adding certain layers to the base architecture and training for additional steps on the desired objective (e.g., a classification loss).

Until recently, the best approaches for classification and generation tasks have been realized with specialized architectures for each. For instance, an encoder-only architecture such as BERT coupled with a dense output layer has been a standard recipe for classification tasks [44]. For generation tasks, encoder-decoder architectures (such as the ones used in BART [26] or PEGASUS [57]) have demonstrated strong state-of-the-art performance.

B. T5 Unified Architecture

A recent emerging paradigm in NLP has been the introduction of unified architectures that can achieve strong performance on both classification and generation tasks [16, 40]. The T5 model by Raffel et al. [40] has been one of the leading performers on language understanding benchmarks such as SuperGLUE [48] and has been matching or exceeding the performance of specialized architectures on generation tasks [19]. T5’s unified architecture is based on casting problems into the text-to-text paradigm and training an encoder-decoder model on a text generation objective. The input to the model encoder is a sequence of text tokens. In the case of multiple inputs, these are simply concatenated into one sequence of tokens. For instance, given a paraphrase detection task, such as that in the Microsoft Research Paraphrase Corpus (MRPC) [15], the input would be: “mrpc sentence1: I found it expensive. sentence2: I found it not so cheap.” The model’s decoder output would be another sequence of tokens. For the same paraphrase detection (classification) task, that output text would be either “equivalent” or “not-equivalent”. T5 comes in a variety of sizes ranging from T5-small (60M parameters) to T5-11B (11B parameters).

C. Analysis of App Reviews

NLP has been proposed to mine and extract useful content from app reviews for a variety of purposes. We dissect these along eight dimensions in Table I. For classification models, we notice that there are two shortcomings of previous approaches: (1) on the modeling side and (2) on the data selection side. First, despite the aforementioned research highlighting the performance leaps brought by models relying on pretraining, the vast majority of recent works still develop classifiers based on traditional NLP approaches, such as SVM and Logistic regression [5, 32, 34]. Second, particularly when it comes to efforts tackling privacy and security, data selection has so far been relying on keyword or regex-based approaches [5, 7, 32, 33, 34].
These choices are intercorrelated as it is easy to achieve high performance using traditional models on tests sets created with such sampling methods. One core contribution of Hark is a new approach for dataset construction that aims to cover two major privacy taxonomies without being keyword-restricted, by leveraging the Natural Language Inference (NLI) [30] task. On such a diverse dataset, the limitations of traditional classification models become apparent, making the case for integrating state of the art models, such as T5 [40]. For general review analysis, some studies went beyond classification. In [11], the authors showed state-of-the-art results on the task of identifying software requirements from app reviews. However, that task is limited since requirements are simply phrases extracted verbatim from the review and do not repeat elsewhere (e.g., “I don’t understand why I should allow you to my cam or calls to (multiple) issues: “Unnecessary Camera Access” and “Unnecessary Calls Access”.

Next, these issues are aggregated across the whole corpus and are grouped into themes based on their semantic similarity (Section VII). Each group of issues and the associated feedback constitute a theme. The most frequent issues in the theme are used to generate a theme title automatically via an abstractive theme summarization model. For instance, a theme with the top issues “Cannot Access Activity Controls”, “Turn Off Activity History”, and “Turn on Activity History” would get the title “Activity Management”.

By generating this hierarchy of high-level themes and fine-grained issues, Hark enables developers to navigate the privacy-related feedback at multiple levels of abstraction. To enrich the navigation experience, Hark includes an emotion classification model with 28 categories (Section VIII), thus providing a valuable way for filtering issues and themes by the level of anger, joy, confusion, etc. Hark further attaches to each fine-grained issue a set of high quality quotes (Section VIII), allowing developers to dig deeper into representative feedback behind the issues of interest. By combining the issues, themes, emotions, top quotes, and feedback metadata (timestamp, star rating, etc.), Hark unlocks this user-to-developer channel, equipping developers with the material to perform a variety of trend analyses and to track their progress on a variety of metrics. We provide illustrative examples of these in Section IX.

IV. MODELING APPROACH

In Hark, we use T5-based models (cf. Section II-B) in our various generation and classification tasks, adding the necessary optimizations to tailor them to the domain at hand. Figure 2 illustrates our modeling approach for the 5 tasks we described in Section III. Essentially, we cast each task as a text-to-text one, and separately finetune a T5 model for it. We will shed light on each of these 5 tasks and the training data in the respective sections.

When adapting T5 models to our tasks, the text is initially tokenized (i.e., broken into tokens) using the T5 SentencePiece tokenizer that breaks each review into a sequence of subwords, thus minimizing the effect of out-of-vocabulary words [25]. We then finetune the T5 models using the maximum likelihood training objective, with teacher forcing [54].

At inference time, i.e., when we want to run the model on new data, the text is decoded one token at a time. In a “greedy” decoding setup, the token with the maximum log-likelihood (referred to as logit) is selected at each step. In the special case of classification tasks, we are also interested in the scores of the various classes. We compute these scores...
by feeding the input text to the model encoder and each of the target classes’ tokens to the model decoder. Given the logits of these classes, we apply a Softmax function to obtain a set of normalized scores that sum up to 1. This method for approximating the classification probabilities in text-to-text models has been shown to be effective by Nogueira et al. [35].

V. PRIVACY FEEDBACK CLASSIFIER

We now describe the first stage of the Hark pipeline, namely the privacy feedback classifier, which distinguishes reviews related to privacy from those which are not.

A. HARK REVIEWS CORPUS

During August 2021, we collected a large corpus of app reviews from Google’s Play store, which we use in the rest of this paper. For each review, we collected its content, the submission time, its star rating, the package name of the corresponding app, and the app’s Play store category information. We limit our corpus to English-only reviews as identified by the CLD3 language identification library (github.com/google/cld3). Our review dataset contains a total of 626M reviews from 1.3M apps published across all of the Play store app categories.

Ethical considerations: App reviews are already public, and users who submit reviews are aware of this. Nevertheless, we took several steps to ensure user privacy and avoid user identification. First, no user information is stored during the reviews gathering process. Second, we only included apps that had at least 10K installs and at least 1000 reviews. Third, we will not release the raw reviews data.

B. CREATING TRAINING DATA

A core challenge we faced in constructing the training data for this classifier is that only a small subset of app reviews relate to privacy. Mukherjee et al. [32] have estimated privacy reviews to be around 0.5% of all reviews while Nguyen et al. [34] estimated both security and privacy reviews to constitute 0.12% of all reviews. Regardless of the methodologies employed (we address their limitations below) and the accuracy of these estimates, this order of magnitude indicates that uniformly sampling reviews and labeling them as privacy vs. not-privacy is highly inefficient and would consume tremendous labeling resources.

1) Creating NLI-Annotated Corpus for Manual Labeling

We need to extract a seed corpus with a significant presence of privacy-related reviews from the full corpus. This would allow us to sample candidate data that undergoes manual labeling before using it to train the privacy classifier.

Similar needs have arisen in previous works targeting review analysis, in the context of security or privacy reviews [32, 34, 45]. The common approach these works followed was to search the full corpus using a limited set of seed keywords compiled for the target domain (e.g., privacy, permissions, personal info, etc.). Then the resulting data is annotated to train a machine learning model. This approach has clear limitations in terms of the topical diversity of privacy issues. Essentially, the domain of collected texts will be limited to the well-known privacy issues that the keywords represent. The model trained on the annotated version of these texts is also highly prone to overfit on the presence of these keywords (or their absence). Hence, a model can appear to have a high performance on such datasets while suffering when tested in the wild. In this work, we take a more principled approach at constructing this seed corpus, which is designed to ensure a high diversity of the various privacy topics, without being keyword-driven.

In order to achieve such diversity, we rely on two commonly used and complementary taxonomies developed for privacy: the taxonomy of privacy violations by Solove [43] and the taxonomy of privacy enhancing technologies by Wang and Kohsa [50]. We extracted most of the concepts from these taxonomies, excluding those outside the scope of this work, such as “security”. The full list of these concepts is in Table II.

Now that we have a set of high-level concepts that cover a
wide range of issues in the privacy domain, we want to identify sample reviews that discuss each topic. Our approach leverages the task of Natural Language Inference (NLI), which is the problem of deciding whether a natural language hypothesis can reasonably be inferred from a given premise [30]. An NLI model has to determine whether a hypothesis is true (i.e., entailment), false (i.e., contradiction), or undetermined (i.e., neutral) given a premise. For example, take a premise saying “This app does not offer any visibility controls to hide your information.” A hypothesis that says “app data is publicly accessible” would receive an entailment label. A hypothesis that says “app data is kept private” would receive a contradiction label. A hypothesis that says “app has a good interface” would receive a neutral label.

Our idea is to leverage NLI models in order to find reviews discussing certain privacy concepts. The premises in our context would be the app reviews. The hypotheses would be manually constructed based on the privacy concepts we selected earlier. For each concept, we came up with one or more hypotheses. For example, for the “blackmailing” concept, we created the hypothesis “A data blackmailing issue is discussed.” We also included 7 additional hypotheses covering generic mentions of privacy issues or positive privacy features. In total, we ended up with 35 hypotheses (see Table II). We chose a model trained on MultiNLI, which is a multi-genre dataset of 433K sentence pairs covering a variety of domains [53]. This helps handling the general breadth of topics raised in app reviews. We use the publicly-available Vanilla T5-11B model checkpoint, which is readily finetuned on the MultiNLI dataset (as part of the GLUE mixture of tasks [47]). We run the NLI model on a dataset of 9M reviews, randomly sampled from the full dataset of 626M reviews. With 35 hypotheses, this amounts to a total of 35×9M=315M inference operations. We refer to these 9M reviews and the entailment probabilities assigned per hypothesis as the NLI-Annotated Corpus. One major advantage of this method is that it eliminates the reliance on keywords. The premises corresponding to the hypotheses can have a high linguistic variability. For instance, both of the following reviews receive an entailment label for the hypothesis “Personal data disclosure is discussed.”:

- “this game will NOT open unless you agree to them sharing your information to advertisers” (P(entailment)=0.89)
- “and doesn’t ask for access to unneeded personal data permissions. Well done developers 5Stars” (P(entailment)=0.75)

Notice that the first review has no words in common with the hypothesis. Neither review mentions disclosure, and one of them explains a problem while the other has a positive sentiment.

![Privacy classifier construction stages](image)

**Fig. 3: High level overview of the privacy classifier construction stages.**

---

### Table II: Privacy Concepts and Associated Hypotheses

<table>
<thead>
<tr>
<th>Privacy Concept</th>
<th>Hypotheses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concepts from Solove’s Taxonomy</td>
<td></td>
</tr>
<tr>
<td>Surveillance</td>
<td>The user is facing a data surveillance issue.</td>
</tr>
<tr>
<td>Interrogation</td>
<td>The user is forced to provide information.</td>
</tr>
<tr>
<td>Aggregation</td>
<td>Personal user information is collected from other sources.</td>
</tr>
<tr>
<td>Insecurity</td>
<td>The user is concerned about protecting their personal data.</td>
</tr>
<tr>
<td>Identification</td>
<td>A data anonymity topic is discussed.</td>
</tr>
<tr>
<td>Secondary Use</td>
<td>The user is concerned about the purposes of personal data access.</td>
</tr>
<tr>
<td>Exclusion</td>
<td>The user wants to correct their personal information.</td>
</tr>
<tr>
<td>Breach of Confidentiality</td>
<td>A breach of data confidentiality is discussed.</td>
</tr>
<tr>
<td>Disclosure</td>
<td>Personal data disclosure is discussed.</td>
</tr>
<tr>
<td>Exposure</td>
<td>The app exposes a private aspect of the user life.</td>
</tr>
<tr>
<td>Increased Accessibility</td>
<td>User’s data has been made accessible to public.</td>
</tr>
<tr>
<td>Blackmail</td>
<td>A data blackmailing issue is discussed.</td>
</tr>
<tr>
<td>Appropriation</td>
<td>User data is being exploited for other purposes.</td>
</tr>
<tr>
<td>Distortion</td>
<td>False data is presented about the user.</td>
</tr>
<tr>
<td>Intrusion</td>
<td>Unwanted intrusion to personal info is discussed.</td>
</tr>
<tr>
<td>Decisional Interference</td>
<td>Intrusion by the government to the user’s life is discussed.</td>
</tr>
<tr>
<td>Concepts from Wang and Kobsa’s Taxonomy</td>
<td></td>
</tr>
<tr>
<td>Notice/Awareness</td>
<td>Opting out from personal data collection is discussed.</td>
</tr>
<tr>
<td>Data Minimization</td>
<td>More access than needed is required.</td>
</tr>
<tr>
<td>Purpose Specification</td>
<td>The reason for data access is not provided.</td>
</tr>
<tr>
<td>Collection Limitation</td>
<td>Too much personal data is collected.</td>
</tr>
<tr>
<td>Use Limitation</td>
<td>The data is being used for unexpected purposes.</td>
</tr>
<tr>
<td>Onward Transfer</td>
<td>Data sharing with third parties is discussed.</td>
</tr>
<tr>
<td>Choice/Consent</td>
<td>User choice for personal data collection is discussed.</td>
</tr>
<tr>
<td>Concepts from Wang and Kobsa’s Taxonomy</td>
<td></td>
</tr>
<tr>
<td>User</td>
<td>User did not allow access to their personal data.</td>
</tr>
<tr>
<td>(H_i) &amp; (P_{E}(i)) &amp; (P_{T}(i)) &amp; (P_{M}(i)) &amp; (P_{R}(i)) &amp; (P_{A}(i)) &amp; (P_{F}(i)) &amp; (P_{C}(i)) &amp; (P_{D}(i)) &amp; (P_{E}(i)) &amp; (P_{T}(i)) &amp; (P_{M}(i)) &amp; (P_{R}(i)) &amp; (P_{A}(i)) &amp; (P_{F}(i)) &amp; (P_{C}(i)) &amp; (P_{D}(i)) &amp; (P_{E}(i)) &amp; (P_{T}(i))</td>
<td></td>
</tr>
</tbody>
</table>

### 2) Creating Manually Labeled Training Data

We use the the NLI-Annotated Corpus to sample diverse data for manual labeling. Given the 9M reviews, let \(N_{E}(i, t)\) be the number of hypotheses receiving an entailment score above a threshold \(t\) for review \(i\). We apply the following heuristics:

- We designate a review \(i\) as maybe-not-privacy if \(N_{E}(i, 0.4) = 0\).
- We designate a review as maybe-privacy if \(N_{E}(i, 0.8) >= 1\) or \(N_{E}(i, 0.7) >= 3\) or \(N_{E}(i, 0.6) >= 5\) or \(N_{E}(i, 0.5) >= 7\).

The intuition is that the more hypotheses a review satisfies, the more likely it is to be within the privacy domain. The rest
of reviews that satisfy neither of these heuristics are considered as undetermined and are not used further. This is in order to leave a safe margin between these heuristics.

Notice that our few hypotheses per concept are not meant to completely cover the underlying concepts. They are designed to produce a diverse sample of candidate data for manual labeling. Since we sample data from both true and false matches on the hypotheses, we also capture some parts of the concepts not readily included in our hypotheses.

From the reviews annotated by the heuristics, we randomly sampled 3,254 reviews, ensuring nearly equal representation across: (1) maybe-privacy vs. maybe-not-privacy labels, (2) four different review world length buckets, and (3) app categories. We get these sampled reviews manually annotated to create a high quality privacy training dataset.

In order to mitigate the effect of individual perceptions of what constitute privacy [52], we created labeling instructions (available at github.com/google/hark), that explained the task, and provided definitions for privacy and not-privacy labels. We ensured to clarify some tricky cases (e.g., around security, scam, spam, etc.) by offering several examples. We recruited annotators from our company’s internal crowdsourcing platform that contracts with third-party vendors to source thousands of annotators across the world for labeling the reviews as privacy or not-privacy. Our annotator pool is composed of college-educated individuals with a nearly balanced gender distribution and more younger population (~50% are in the age range of 25-34, with less than 5% above 55 years). These annotators are paid per hour based on local market conditions at a rate set by our employer. Each review was then labeled by 5 annotators, and a total of 1,332 annotators labeled the 3,254 reviews. Krippendorff’s alpha [24] for inter-annotator agreement was 0.455. While this agreement value might seem low, it is within an acceptable range for cases using crowdsourcing for evaluating latent constructs (privacy in our case) [28]. Of the 3,254 reviews manually annotated, 99.4% of maybe-not-privacy were labeled as not-privacy and 64.3% of maybe-privacy were labeled as privacy by the annotators. This indicates that the NLI approach results in almost no false negatives but contributes some false positives. Hence, it is necessary to couple it with a manual annotation step to generate high quality training data. In Appendix B, we break down the data distribution across the various privacy concepts we sampled from.

C. Model Training

From the 3,254 labeled examples, we extracted a balanced test set of 300 examples (split equally between the privacy and not-privacy labels). From the remaining data, we take 200 items (82 of them are privacy) as the validation set and the remaining 2,754 reviews (1030 of them are privacy) as the training set. Next, we trained a T5-11B model on this training data (parameters in Appendix A). In Figure 3, we summarize the various steps we described for building Hark’s privacy feedback classifier.

D. Classifier Performance

We use the 300 examples test set to compare the performance of our privacy classifier with four baseline classifiers. These classifiers are varied across the dataset and the model architecture dimensions. In addition to our training data, referred to as Hark Data, we consider the dataset by Nema et al. [33] at ICSE 2022 (referred to as ICSE Data). That dataset is built based on regex patterns developed to cover a privacy taxonomy. Hence, we compare our model (T5-11B Hark Data) to:

• T5-11B - ICSE Data: T5-11B model trained on ICSE Data.
• SVM - Hark Data: SVM Classifier based on bag of words (using 3-5 character n-grams), reproducing the one used in Nguyen et al. [34].
• RoBERTa-Large - Hark Data, a 24-layer deep learning model, achieving strong results on various classification tasks [29].
• RoBERTa-Large - ICSE Data: variant trained on ICSE Data.

Figure 4 shows the Receiver Operating Characteristic (ROC) curves and the corresponding AUC-ROC values for our model and the four baselines. We observe that the T5-11B model trained on Hark Data obtained 0.17 higher AUC compared to the same model trained on ICSE Data (0.92 vs 0.75). We also independently tested T5-11B - Hark Data on the ICSE test set and found that it matches the best reported ensemble model performance (AUC=0.98) by Nema et al. [33]. This illustrates that Hark’s method leveraging NLI for training set sampling enables generalization to other test sets while regex-based sampling of training data fails in that regard. Another observation we see is that models such as the SVM model used by Nguyen et al. [34] fail to learn the nuances of our syntactically and semantically diverse dataset (AUC=0.73 on Hark’s test set), despite getting a reported AUC-ROC of 0.93 on a keyword-sampled test set in [34]. Using RoBERTa-Large with Hark Data improves the AUC by 0.13, and using the T5-11B results in 0.19 absolute increase in the AUC. This shows the power of using larger models that benefit from transfer learning. In Appendix E, we provide qualitative examples, illustrating our classifier’s superior performance vs. the baselines.

VI. Issue Generation

Having developed the privacy classifier module, which allows us to extract privacy-related reviews, we now describe Hark’s
issue generation model which aims to surface the fine-grained topics that users discuss.

A. Problem Formulation

Given a user review, the goal is to generate one or more issues summarizing the main topics that the user is discussing. We use the term issue in the generic sense (i.e., it can denote both negative and positive experiences).

One approach to generate these issues is to enumerate all the possible topics users might discuss (e.g., “Unnecessary Permissions”, “Data Deletion”, etc.), construct training examples for each of them, and build a classification model to tag new examples with these labels. This approach has two main limitations. First, creating training examples for each label requires a significant effort. That is why previous works on reviews’ analysis have used limited taxonomies (e.g., 12 fine-grained classes were used by Ciurumelea et al. [10]). To cover all possible issues, these classes tend to be too broad.

Second, the topics mentioned in the reviews evolve over time (a phenomenon called concept drift [51]). Hence, a classification approach falls short in detecting the emerging issues.

Another approach is to extract important words in the reviews and rely on these words conveying the issues [11, 22]. However, that would result in a set of dispersed, out-of-context quotes that do not necessarily convey the actual issues users discuss.

Hence, in this work, we take an abstractive labeling approach that combines the generalization power of abstractive models (similar to the ones used in summarization) with the familiar style of issue labels. Our goal is to obtain issues with the following features:

- concise: Issues are typically 2-4 words, allowing developers to glance through a large set with a minimal effort.
- consistently worded: When users raise the same topic in different reviews, the issue would be worded in an almost identical manner.
- fine-grained: Issues highlight the actual topics users discuss rather than high-level concepts, such as “bugs” or “feature requests”.

We aim to achieve this goal by: (i) authoring a new dataset with a concise and consistent style of issues for the given reviews; (ii) training a generative model, based on T5 (Section II-B), in a way that leads it to behave like an abstractive summarization model rather than a classification model.

B. Training Dataset Creation

We wanted to sample a diverse set of reviews for our dataset. A natural starting point is to re-use the NLI-Annotated Corpus we created in Section V-B1 as that allows us to cover a variety of privacy concepts. Hence, we ran the T5-11B Hark privacy classifier from Section V on that corpus to keep the reviews tagged as privacy. Then, we sampled 1,060 reviews from that corpus while ensuring diversity across (1) the covered hypotheses, (2) the reviews’ length, and (3) app categories.

Two of the authors then annotated the reviews with the set of issues they contain. For instance, the review “It shows up on locked screen and u can see who wrote what and who wrote it...” was tagged with the issue “Lock Screen Visibility”. The annotation was performed in two stages. Author A did a first pass on a quarter of the reviews, following the conciseness and consistency guidelines. Then author B provided feedback on the issues created in that round, and the two adjusted the wording as necessary. That way, author B was exposed to the style of A, allowing them to mimic that style when creating issues. The two authors continued labeling the rest of the reviews and held a final round of feedback at the end, adjusting the issues as necessary. Notice that the outputs of these annotations are free-form issues. Hence, there was no need for more than one annotator per review for the training data creation (we do that for the evaluation later). Across these reviews, the annotators produced 1,851 issues. Of these, 1,123 were unique.

C. Issue Generation Model Training

As explained in Section IV, we will be using T5 as the main model across the various tasks in this work. We continue to use the largest available T5 version (T5-11B) as it has been shown to have the best performance on the generative tasks compared to other model sizes [40]. Despite our attempt at diversifying the data, there are certain issues that are very prevalent in the case of app reviews. For example, the issues “Account Hacking”, “Excessive Permissions”, and “Unneeded Contacts Access” occurred 54, 29, and 23 times respectively in the annotated data. We empirically observed that allowing such frequent issues in the training data would lead the T5 model to over-generate them at inference time. Hence, it would behave like a classification model, often restricting itself to the frequent issues observed at training time. To mitigate that, we imposed a limit that an issue can occur a maximum of 2 times across the whole training data. That way, we nudge the model to learn the task of originating issues for reviews rather than assigning from a common set of issues it has been exposed to. All the additional annotated reviews that are above that limit are moved to the validation data. We ended up with 613 training examples and 447 validation examples. Next, we trained the T5-11B model on this data (parameters in Appendix A).

D. Evaluation Setup

In order to show the efficacy of the issue generation component in Hark and to justify the major design decisions, we evaluate the following models:

1) Hark Issue Gen: T5-11B issue generation model.
2) T5 Wikihow: T5-11B model trained on an existing public dataset for abstractive summarization. We chose to train this model on the wikihow/sep dataset [23], where the task is generating section titles for sections on the website wikihow.com. This was the closest publicly available dataset to the task at hand.
3) RE-BERT: a RE-BERT model [11], which is a state of the art extractive model for identifying software requirements from app reviews. This model extracts the most relevant words/phrases from the text as the requirements as opposed to the previous abstractive models that are not bound by selecting from the input review.
We followed the same approach we used to sample training data in Section VI-B to create a diverse test set for evaluation. To enable us to compare the various models, we filtered the newly sampled data to only keep the reviews where the models produced different sets of issues. We ended up with 600 reviews in the evaluation set. Our strategy is to compare the above models based on two metrics:

- **Accuracy**: an issue-level metric indicating how precise each issue is in capturing the intent of the review.
- **Coverage**: indicates how comprehensive a set of issues is in capturing the main topics mentioned in the review.

It is a well-accepted convention in the Natural Language Generation literature that human evaluation is the best method for evaluating the outputs of generative models [42] as compared to automated metrics (such as BLEU [37] or ROUGE [27] which correlate the model-generated output with manually-created outputs). Hence, we designed two human evaluation studies, one for each metric.

We note one distinguishing aspect of the task at hand, which is the highly subjective nature of the evaluation, where agreement among raters is not expected to be high. Previous works [2, 3] have studied this extensively, showing the limitations of using traditional agreement metrics, such as Krippendorff’s alpha [24], as those are primarily designed for objective tasks. That was not restricted to crowdsourcing, but also to expert annotators. Hence, in our evaluation, we tackle that by reporting metrics at varying levels of agreement, showing how the different systems fare. Previous works have used the minimum agreement level as a way to filter crowdworkers annotations [55].

1) **Accuracy Evaluation**

In this first study, we display the review alongside one issue from each evaluated model. We ask the annotators to label each issue with one of the following choices:

- **Topic_Discussed**: The topic is discussed in the review.
- **Not_A_TOPIC**: Contains keywords present in the review, but is not a topic.
- **Unrelated**: Unrelated to the review.

We chose to select one issue per model for this experiment since accuracy is an issue-level metric. Our labeling instructions explained the task (available at github.com/google/hark). We also used a similar pool of annotators to that described in Section V-B2 for the accuracy task. We shuffle the models’ order per review to avoid any positional bias. Each review was annotated by 7 raters, and a total of 267 raters were involved.

We measure the accuracy as the percentage of reviews where **Topic_Discussed** was the most frequently chosen label (by 3 or more annotators out of 7). As described above, we report in the upper part of Figure 5 the accuracy for each value of $N$, where $N \in [3, 7]$ is the minimum number of annotators that chose the label. In the bottom part, we show how many reviews are still considered per model if we impose that minimum agreement level. This part will be used to judge whether the former graph is representative. Hence, we show the total number of reviews satisfying that level and not only the ones with the choice being **Topic_Discussed**.

2) **Coverage Evaluation**

In the case of RE-BERT, the accuracy decreases from 28% with $N = 3$ to 19% to $N = 5$. It increases back to 30% when $N = 7$. However, at $N = 7$, the sample of reviews considered is too small to be representative (only 10 reviews). With the T5 Wikihow model, we notice a different trend, where the accuracy increases from 56% at $N = 3$ to 79% at $N = 5$. This indicates that abstractive models like T5 Wikihow, even if not customized to the domain at hand, are better suited for generating the topics in the reviews compared to extractive models that select phrases from the text. Our Hark Issue Gen model’s accuracy, which is customized to the reviews domain, shows the full power of this approach. Its accuracy increased from a minimum of 83% at $N = 3$ to reach 96% at $N = 5$. Even at $N = 5$, around 57% of the reviews are still being considered with Hark Issue Gen (vs. 38% and 39% for RE-BERT and T5 Wikihow respectively). This indicates that our system results in (1) annotators agreeing more often on its outcomes and (2) the agreement being primarily on the **Topic_Discussed** choice. We measured the statistical significance of the differences between each two models at the different agreement levels using McNemar’s test, with Bonferroni correction for multiple comparisons [31]. The null hypothesis was that the marginal probability for the binarized outcome (**Topic_Discussed** or not) is the same for each pair of models. The differences between Hark Issue Gen and the other models were significant ($p < 0.05$) for $N \in [3, 5]$ vs. T5
Wikihow and for \( N \in [3, 6] \) vs. RE-BERT.

We take the case of \( N = 5 \) and plot it in Figure 6 as a suitable spot where we have statistically significant differences, a high level of agreement, and a considerable number of reviews. We can observe that the RE-BERT model is perceived to produce keywords that are not a topic in 74% of the cases. This occurred in only 4% of the cases with Hark Issue Gen. We also see that T5 Wikihow has a higher level of Unrelated issues (18%) compared to RE-BERT (6%), which is expected given that it is an out-of-domain abstractive model. Our Hark Issue Gen model, in contrast, does not have this issue and produces Unrelated outputs in only 0.3% of the cases at \( N = 5 \).

2) Coverage Evaluation

In the second study evaluating the coverage metric, we display the review alongside the full set of issues produced by each evaluated model (as compared to a single issue per model in the accuracy evaluation). We ask the annotators to label each set of issues with one of the following choices:

- **Topics_Covered**: Label set covers the main topics mentioned in the review.
- **Topics_Not_Covered**: Label set contains keywords from the review, but does not capture any main topics.
- **Unrelated**: Label set is not related to any main topics in the review.

Our labeling instructions are available at github.com/google/hark. We also used here a pool of annotators similar to that described in Section V-B2. We evaluate the coverage for the same set of 600 reviews sampled for the accuracy evaluation. We also shuffle the models’ order per review so as to avoid any positional bias. Each review was annotated by 7 raters, and a total of 272 raters were involved.

We measure the coverage as the percentage of reviews where **Topics_Covered** was the most frequently chosen label (by 3 or more annotators). We show in the upper part of Figure 7 the coverage for various values of \( N \), which is the minimum number of annotators that chose **Topics_Covered** as the label. The bottom part of the figure shows how many reviews are still considered for value of \( N \in [3, 7] \) (regardless of the choice agreed upon).

We can observe similar trends to the case of accuracy evaluation. Notably, the RE-BERT model performs the worst with the coverage consistently decreasing from 18% at \( N = 3 \) to 7% at \( N = 5 \) (with 45% of the reviews considered). We see the complete opposite trend with Hark Issue Gen, where the coverage evolves from 83% at \( N = 3 \) to 93% at \( N = 5 \) (with 60% of the reviews still considered). This indicates that, as more annotators agree, they tend to agree on Hark Issue Gen producing high coverage outputs. The differences between Hark Issue Gen and the other models are significant for \( N \in [3, 6] \) \( (p < 0.05 \) with McNemar’s test and Bonferroni correction). The null hypothesis was that the marginal probability for the binarized outcome (**Topics_Covered** or not) is the same for each pair of models. It is worth noting that both RE-BERT and Hark Issue Gen, by design, produce multiple issue candidates from the review. RE-BERT generates 4.7 candidates on average while Hark Issue Gen generates 2.1 on average. Hence, they are comparable in that regard. The T5 Wikihow model, on the other hand, is not trained to do so. Hence, its perceived coverage at \( N = 3 \) (42%) was much lower than its accuracy (56%). These observations indicate that Hark Issue Gen strikes a good balance by producing the minimal set of issues that are enough to achieve high coverage.

This conclusion is further solidified when plotting the case of \( N = 5 \) in Figure 8. That figure also shows that Hark Issue Gen avoids Unrelated outputs (unlike the other abstractive model - T5 Wikihow) and that it produces issues that cover the main topics in the review. In Appendix E, we further show qualitative examples of Hark Issue Gen’s outputs compared to the baselines.

VII. Theme Creation

After having explained how we generate issues for individual reviews, we now move from analyzing a single review to analyzing a body of reviews. The core outcome of this section is showcasing how to organize a large set of fine-grained issues under high-level themes, providing developers with a bird’s-eye view of the issues users are discussing. We proceed in 2 stages: issues grouping and theme title creation (see Figure 9).

A. Issue Grouping

After obtaining the issues, we want to group these issues into themes. To achieve that, we use the **Leader Algorithm** for clustering [20]. Given a set of items in a certain order,
this algorithm produces clusters composed of items which are within a maximum distance $d_{max}$ from the cluster leader. It has several interesting properties. First, it requires a single pass over the data, which makes it very fast. Second, it is order-dependent, which is a desired property in our case as we want the high-frequency issues to act as cluster leaders. That is why we order the input issues based on their descending frequency order. As a distance metric within the clustering algorithm, we use the cosine distance between the embedding vectors of each two issues. We compute these embedding vectors based on the Transformer-Based Universal Sentence Encoder [8], which is trained on general text similarity tasks. The outcome of this stage is a set of issues acting as leaders of clusters. Each cluster practically corresponds to a high-level theme that we want to relay to the developer. We empirically found that $d_{max} = 0.9$ is a suitable threshold for the grouping step.

B. Theme Title Creation

Although clustering has been used before in the context of reviews analysis [14, 36] (albeit not applied to abstractive fine-grained issues), a key limiting aspect about it is that it produces a long list of groups without meaningful, representative titles. Hark eliminates that limitation by adding a generative model capable of taking the most frequent issues on a closely related topic and combining them into a high-level theme. We take a similar approach to the issue generation problem in Section VI. The main difference is that, here, we are summarizing issues into themes instead of summarizing reviews into issues. Hence, we create a theme generation dataset and train a generative model on that data.

To create a candidate dataset of issues to summarize, we started from a subset of 200K reviews tagged as privacy by our privacy classifier from Section V. We applied the various steps in the Hark pipeline, namely issue generation and issue grouping. We only considered clusters with more than two issues, and we chose a maximum of 10 issues per cluster (keeping the most frequent issues). We chose 570 sets of issues for manual annotation. In total, these contained 2,171 issues (i.e., an average of 3.8 issues per set). Then one of the authors went through each set of issues and created a title. For instance, the set of issues: “Unable to Record Calls, Unable to Call, Unable to Receive Calls, Unable to Hear Calls, Unable to Record Caller Voice” received the title “Call Management Issues”. Since this is an open-text generation task, we did not need to have multiple titles per set of issues (we have multiple annotators though during evaluation).

Next, we split the manually annotated data into 80% training data and 20% validation data. Similar to what we did in Section VI-C, we also use the T5-11B model for this generative task (parameters in Appendix A).

C. Evaluation

1) Baseline

To illustrate the advantages of our approach, we wanted to compare against a strong baseline. We are not aware of any publicly available dataset that is close enough to the domain at hand. Hence, our go-to baseline is GPT-J 6B [49], a causal language model (cf. Section II-A) that was shown to have strong zero-shot performance on a variety of NLP tasks. The idea is to do model priming [6], leveraging the model’s ability to auto-complete text, when provided with enough context, as a way to generate theme titles. As the model input, we provide a text stating 4 examples of issue sets with the expected titles. The last sentence of the input has a new set of issues for which we want to generate a title. We run GPT-J 6B on this combined text, and we expect it to auto-complete with the generated title. This approach performed decently well in our testing. For example, it generated the title “Feature Requests” for the issues “Asking for Feature, Asking for Rating, Requesting Messaging, Premium Feature Required, Asking for Visibility”.

2) Evaluation Data

We created the evaluation data in a similar fashion to the training data construction, by starting from 1.5M examples and going through the Hark pipeline. We ensured that there are no issue set in the evaluation dataset that has more than 50% overlap with any issue set in the training data. We sampled 600 issue sets from this dataset, and we conducted a human evaluation to assess the quality of the generated titles.

3) Study Results

We created a study where the annotators were given a set of issues as well as titles generated by our model (referred to as Theme-Gen) and by the baseline GPT-J in a randomized order. The instructions, which we provide at github.com/google/hark, required the user to annotate each title with one of the following:

- Title Covers: Title covers the vast majority of the labels.
- Title Misses: Title misses the vast majority of the labels.
- Unrelated: Title is unrelated or misrepresents the labels.

As this evaluation task is also asking a subjective question, we follow a similar methodology to that used for evaluating issue generation accuracy and coverage in Section VI-D. We measure the title quality as the percentage of cases where Title_Covers was the most frequently chosen label (by three annotators or more out of seven).

The top part of Figure 10 shows the quality of the two models with respect to the minimum number of annotators who agreed on Title_Covers being the choice. Below it, we show the total number of reviews considered after imposing a minimum agreement level of $N \in [3, 7]$ (regardless of the choice agreed upon). This is to understand how representative the numbers in the top chart are. The quality of the titles generated with our Theme-Gen model goes from around 83% at $N = 3$ and $N = 4$ to 92% at $N = 5$. At $N = 5$, 360 (i.e., 60%) of the reviews are still being considered. The GPT-J model, in contrast, has a much lower quality of 60% (at $N = 3$) and reaches 72% at $N = 5$. We also observe that the number of reviews where the annotators agree on the decision is 49%
higher with Theme-Gen compared to GPT-J at \( N = 5 \). Hence, Theme-Gen results in significantly better quality and higher annotator agreement. The differences between Theme-Gen and the GPT-J are significant (\( p < 0.05 \) with McNemar’s test with Bonferroni correction) for \( N \in [3, 5] \). The null hypothesis was that the marginal probability for the binarized outcome (Theme-Covers or not) is the same for each pair of models.

In Figure 11, we take the case of \( N = 5 \) and showcase the percentage of reviews with each of the three choices. Notice that Theme-Gen has no cases where 5 or more annotators perceived the title as Unrelated while this was the case in 6% of the titles produced by the GPT-J baseline. Overall, these results solidify the case for using a generative model like Theme-Gen, which is finetuned on an in-domain dataset. In Appendix E, we further show qualitative examples of Theme-Gen’s outputs compared to the baseline.

**VIII. Improving Navigability**

As described in Section III, by building the hierarchy of high-level themes and fine-grained issues, we enable the developers to have an easy way to track privacy issues in their applications. In order to further improve the navigability of this hierarchy, we introduce two additional models in this section for classifying emotions in the reviews and for classifying high vs. low quality feedback. In both cases, we rely on leveraging existing public datasets and training new models on them. We will further illustrate how these models fit within the bigger system in the next section.

**A. Emotions Model**

*Training Data:* Hark’s emotions classifier builds on the GoEmotions dataset, introduced by Demszky et al. [12]. This is the largest manually annotated dataset of 58k English Reddit comments, labeled for 28 emotion categories.

*Model Training:* We continue to use the T5-11B model for this dataset too (parameters in Appendix A). Since there can be multiple emotions associated with each text in the training data, we chose to train the model on generating a comma-separated list of classes. For example, the input to the model would be “emotion classifier: My two favorite things, The Office and The Show, combined in one reference. Life is good.”. The output would be “admiration, approval”. We used the original training/validation/test datasets from the authors [12].

*Evaluation:* On the test set, our model achieves a 0.54 macro-averaged F1-score across the 28 emotions. This adds 8% in absolute macro-averaged F1 score on top of the existing BERT-based state-of-art model developed by the dataset authors [12]. We report the per-emotion metrics in Appendix C.

**B. Feedback Quality Model**

Next, we describe Hark’s model for assessing review’s quality, which is designed to automatically provide representative quotes for each issue or theme. To achieve that, we needed examples of both high and low quality reviews.

*High Quality Reviews:* For high quality reviews, we collected reviews that have been found to be helpful by other users. This is measured by the number of upvotes displayed next to the review on Google’s Play store. We use an existing publicly available dataset of Play reviews [41] containing such metadata. From that dataset, we extracted 1,090 reviews that have 5 or more upvotes while ensuring diversity across the reviews’ star ratings (on a scale of 1 to 5 stars). On average, the selected reviews received 27.2 upvotes.

*Low Quality Reviews:* We cannot assume that reviews with a low number of upvotes are low quality since such reviews can be simply recent or not viewed by enough users. Hence, we used the AR-Miner dataset by Chen et al. [9], which contains informative and non-informative reviews, manually annotated by humans. Non-informative reviews are those reflecting pure emotional expression or those that are too general or unclear. We selected 1,090 non-informative reviews while ensuring diversity across the star ratings they are associated with. We opted to not use the informative reviews from AR-Miner for the positive examples as we wanted a stronger signal of quality.

*Model Training:* We split the 2,180 examples into 80% training and 20% testing data and trained a T5-11B model on a classification task using the two output labels high and low (parameters in Appendix A). On the testing data, the model had a performance of 99% AUC-ROC. Despite training the model on a classification task, we use the probability of the high label in Hark as a proxy for ranking the quotes per issue/theme.

**IX. Qualitative Analysis of a Large Scale Dataset**

After introducing the various components of Hark, we wanted to showcase how Hark can satisfy the three requirements discussed in Section I: topical diversity, glanceability, and navigability. To achieve this, we ran the full Hark pipeline over the set of 626M reviews in our dataset. Although we illustrate these concepts over a dataset of 1.3M apps, similar
analysis can be performed at the level of a single app (or a single developer’s apps), offering similar types of insights².

Figure 12 showcases an example of the hierarchy that Hark produces. At the top level, a Mosaic plot shows the top 10 identified themes (the width of each bar indicates relative sizes). For each theme, we also show the prevalence of dominant emotions on the vertical axis. For ease of representation, we consolidated the 28 emotions Hark generates into 8 emotions based on Ekman’s emotions taxonomy [17] (using the same grouping criteria done by Demsky et al. [12] and adding the neutral emotion). For instance, the “Unneeded Access” theme has a volume of 546K reviews, 42% of which are associated with Anger and 23% with Confusion.

The diversity across these 10 themes gives a glimpse of Hark’s ability to cover a rich set of privacy topics, ranging from “Excessive Permissions” to “Content hiding”. Across the whole set of reviews, Hark generated over 300 high-level themes that had at least 1000 reviews. Of these, the smallest theme covered about 15 fine-grained issues whereas the largest one covered over 1000 fine-grained issues.

²Our company’s policy does not allow publishing individual apps’ analyses.

The emotions dimension provides an important tool for navigability. Unlike previous works that focused on negative privacy issues [5, 32, 34], our approach uncovered a lot of content associated with positive emotions. An example is the “Content Hiding” theme, where we saw that users are pleased with privacy controls that enable functionalities such as hiding videos and locking photos. The emotions filter also provides developers with a new way to prioritize what to tackle first - as they could select issues with a much higher anger representation over those with the highest volume.

The second Mosaic plot in Figure 12 allows us to zoom into the “Spying Concerns” theme (for example) and look at its top 10 fine-grained issues. This showcases how Hark turned 77k reviews in this theme into an easily-glanceable set of fine-grained issues. We notice that, while users express an elevated level of “Anger” (34%) towards “Spying” (typically general mentions of spying actions), they do not shy away from expressing joy at “No Spying” (even if the latter is of a smaller volume). Surprisingly, the “Spying on Spouse” issue is dominated by “joy” emotions, indicating that this is a highly appreciated feature. This illustrates the potential for Hark to
Alternatively, developers could learn about users' theme titles missing some of the issues or emotions interpreted rarely become frequent issues. Other errors, however, such as the privacy classifier, our pipeline can mitigate these as they errors, such as inaccurate issues or false positives produced by high linguistic variability of our domain. For certain kinds of from the inaccuracies of our models when dealing with the our models manifest a variety of error levels. This originates concerns in the original language.

Error Mitigation: At different stages of the Hark pipeline, our models manifest a variety of error levels. This originates from the inaccuracies of our models when dealing with the high linguistic variability of our domain. For certain kinds of errors, such as inaccurate issues or false positives produced by the privacy classifier, our pipeline can mitigate these as they rarely become frequent issues. Other errors, however, such as theme titles missing some of the issues or emotions interpreted inaccurately would be noticed by the developers, which we accept as a limitation.

Volume Estimation: Sometimes users express similar concerns differently, e.g., our fine-grained issue generation can separate “Spying App” and “Spying” into two distinct fine-grained issues. This would affect the individual issue-level volume estimates. This is potentially mitigated when estimating the themes’ volume as these issues eventually make it to the same theme. Solving this completely would require us to further fine-tune in-domain embeddings for issues similarity.

Further Studies: This paper focuses on describing and evaluating the system and models behind Hark. A detailed deep dive into the various aspects of privacy topics on the Play store is out of scope of this work. In the future, we aim to use Hark to conduct various studies: to understand temporal trends in privacy issues, to compare issues based on the emotions dimension, to analyze the type of feedback that leads users to uninstall apps, or to explore particular themes of interest (e.g., “Blackmailing Concerns”, “Financial Privacy”, “Audio Surveillance”, “Parental Controls”, etc.) We also plan to explore when our issue tags can be mapped to actionable suggestions as compared to cases of user misunderstanding or purely sentimental reviews.

X. Discussion and Limitations

Reviews Selection: In order to avoid apps with a handful of privacy reviews, our dataset only includes apps with 10k installs and 1k reviews. These apps constitute a significant proportion of the Play store, and comparing their issues vs. popular apps is an opportunity for future research. Furthermore, we also limited our corpus to English text only. Translating text from other languages may lose privacy-related nuances and introduce translation errors. We plan to better tackle this in the future via multilingual models [56] that capture privacy concerns in the original language.

Error Mitigation: At different stages of the Hark pipeline, our models manifest a variety of error levels. This originates from the inaccuracies of our models when dealing with the high linguistic variability of our domain. For certain kinds of errors, such as inaccurate issues or false positives produced by the privacy classifier, our pipeline can mitigate these as they rarely become frequent issues. Other errors, however, such as theme titles missing some of the issues or emotions interpreted inaccurately would be noticed by the developers, which we accept as a limitation.

Volume Estimation: Sometimes users express similar concerns differently, e.g., our fine-grained issue generation can separate “Spying App” and “Spying” into two distinct fine-grained issues. This would affect the individual issue-level volume estimates. This is potentially mitigated when estimating the themes’ volume as these issues eventually make it to the same theme. Solving this completely would require us to further fine-tune in-domain embeddings for issues similarity.

Further Studies: This paper focuses on describing and evaluating the system and models behind Hark. A detailed deep dive into the various aspects of privacy topics on the Play store is out of scope of this work. In the future, we aim to use Hark to conduct various studies: to understand temporal trends in privacy issues, to compare issues based on the emotions dimension, to analyze the type of feedback that leads users to uninstall apps, or to explore particular themes of interest (e.g., “Blackmailing Concerns”, “Financial Privacy”, “Audio Surveillance”, “Parental Controls”, etc.). We also plan to explore when our issue tags can be mapped to actionable suggestions as compared to cases of user misunderstanding or purely sentimental reviews.
References


### APPENDIX

#### A. Models’ Parameters

In Table III, we show the main hyperparameters for the various models trained in this paper. All of the models are finetuned versions of T5-11B. For our qualitative analysis in Section IX, we chose a privacy classifier threshold of 0.91, resulting in an average precision of 83% and an average recall of 82% on Hark’s test set.

#### B. Privacy Classifier Data Analysis

In Figure 14, we can see that how each concept described in Table II is represented in our sampled dataset used for manual labeling in Section V-B2. We further show the breakdown by ground truth label. We can observe that, when they occur, these concepts are predominantly privacy related.

#### C. Emotions Classifier Results

In Table IV, we show the detailed classification results of the emotions classifier.

<table>
<thead>
<tr>
<th>emotion</th>
<th>precision</th>
<th>recall</th>
<th>f1-score</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td>neutral</td>
<td>0.68</td>
<td>0.67</td>
<td>0.68</td>
<td>1787</td>
</tr>
<tr>
<td>admiration</td>
<td>0.64</td>
<td>0.79</td>
<td>0.71</td>
<td>504</td>
</tr>
<tr>
<td>approval</td>
<td>0.51</td>
<td>0.32</td>
<td>0.39</td>
<td>351</td>
</tr>
<tr>
<td>gratitude</td>
<td>0.92</td>
<td>0.90</td>
<td>0.91</td>
<td>352</td>
</tr>
<tr>
<td>annoyance</td>
<td>0.43</td>
<td>0.23</td>
<td>0.30</td>
<td>320</td>
</tr>
<tr>
<td>amusement</td>
<td>0.75</td>
<td>0.90</td>
<td>0.82</td>
<td>264</td>
</tr>
<tr>
<td>curiosity</td>
<td>0.60</td>
<td>0.47</td>
<td>0.53</td>
<td>284</td>
</tr>
<tr>
<td>love</td>
<td>0.76</td>
<td>0.87</td>
<td>0.81</td>
<td>238</td>
</tr>
<tr>
<td>disappointment</td>
<td>0.51</td>
<td>0.41</td>
<td>0.45</td>
<td>267</td>
</tr>
<tr>
<td>optimism</td>
<td>0.67</td>
<td>0.52</td>
<td>0.59</td>
<td>186</td>
</tr>
<tr>
<td>anger</td>
<td>0.50</td>
<td>0.56</td>
<td>0.53</td>
<td>198</td>
</tr>
<tr>
<td>joy</td>
<td>0.64</td>
<td>0.61</td>
<td>0.63</td>
<td>161</td>
</tr>
<tr>
<td>confusion</td>
<td>0.47</td>
<td>0.52</td>
<td>0.49</td>
<td>153</td>
</tr>
<tr>
<td>sadness</td>
<td>0.62</td>
<td>0.54</td>
<td>0.58</td>
<td>156</td>
</tr>
<tr>
<td>disappointment</td>
<td>0.51</td>
<td>0.25</td>
<td>0.33</td>
<td>151</td>
</tr>
<tr>
<td>realization</td>
<td>0.53</td>
<td>0.17</td>
<td>0.26</td>
<td>145</td>
</tr>
<tr>
<td>caring</td>
<td>0.47</td>
<td>0.45</td>
<td>0.46</td>
<td>135</td>
</tr>
<tr>
<td>surprise</td>
<td>0.59</td>
<td>0.52</td>
<td>0.56</td>
<td>141</td>
</tr>
<tr>
<td>excitement</td>
<td>0.54</td>
<td>0.40</td>
<td>0.46</td>
<td>103</td>
</tr>
<tr>
<td>disgust</td>
<td>0.52</td>
<td>0.45</td>
<td>0.48</td>
<td>123</td>
</tr>
<tr>
<td>desire</td>
<td>0.66</td>
<td>0.47</td>
<td>0.55</td>
<td>83</td>
</tr>
<tr>
<td>fear</td>
<td>0.63</td>
<td>0.77</td>
<td>0.69</td>
<td>78</td>
</tr>
<tr>
<td>remorse</td>
<td>0.54</td>
<td>0.86</td>
<td>0.66</td>
<td>56</td>
</tr>
<tr>
<td>embarrassment</td>
<td>0.55</td>
<td>0.46</td>
<td>0.50</td>
<td>37</td>
</tr>
<tr>
<td>nervousness</td>
<td>0.53</td>
<td>0.43</td>
<td>0.48</td>
<td>23</td>
</tr>
<tr>
<td>relief</td>
<td>0.75</td>
<td>0.27</td>
<td>0.40</td>
<td>11</td>
</tr>
<tr>
<td>pride</td>
<td>0.80</td>
<td>0.25</td>
<td>0.38</td>
<td>16</td>
</tr>
<tr>
<td>grief</td>
<td>0.50</td>
<td>0.50</td>
<td>0.50</td>
<td>6</td>
</tr>
</tbody>
</table>

**micro avg** 0.64 0.59 0.61 6329

**macro avg** 0.60 0.52 0.54 6329

**weighted avg** 0.63 0.59 0.60 6329

**samples avg** 0.65 0.62 0.62 6329

#### D. Additional Analysis Graphs

In addition to the figures we showed in Section IX, we show statistics around the number of issues per app in Figure 15 and the number of reviews per issue in Figure 16. We can observe that, among the apps with privacy issues, the median number of privacy issues is 2. We also see that, for issues with 10 reviews or above, the median number of issues is 25. Still a few issues are widely popular (occurring in tens of thousands of the reviews).

#### E. Qualitative Examples

In Table V, we provide examples of the outputs produced by Hark compared to the baselines for the privacy classifier, issue generation, and title generation models respectively.
TABLE III: MAIN HYPERPARAMETERS FOR THE MODELS USED IN THE PAPER.

<table>
<thead>
<tr>
<th>Task</th>
<th>Learning Rate</th>
<th>Dropout Rate</th>
<th>Batch Size</th>
<th>Training Steps</th>
<th>Label Smoothing Coefficient</th>
</tr>
</thead>
<tbody>
<tr>
<td>Privacy Classifier</td>
<td>0.005</td>
<td>0.1</td>
<td>64</td>
<td>500</td>
<td>0.1</td>
</tr>
<tr>
<td>Issue Generation</td>
<td>0.005</td>
<td>0.1</td>
<td>64</td>
<td>500</td>
<td>0.1</td>
</tr>
<tr>
<td>Theme Title Generation</td>
<td>0.005</td>
<td>0.1</td>
<td>64</td>
<td>500</td>
<td>0.1</td>
</tr>
<tr>
<td>Emotion Generation</td>
<td>0.005</td>
<td>0.1</td>
<td>64</td>
<td>2000</td>
<td>0.1</td>
</tr>
<tr>
<td>Quality Classifier</td>
<td>0.005</td>
<td>0.1</td>
<td>64</td>
<td>500</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Fig. 14: Representation of each privacy concept in the labeled dataset, broken down by the ground truth label annotators assigned later (privacy vs. not-privacy).

Fig. 15: ECDF of the number of issues per app (bottom) with the corresponding box plot (top)

Fig. 16: ECDF of the number of reviews per issue (bottom) with the corresponding box plot (top)
### (a) Examples of Hark Privacy Classifier Results vs. the Baselines

<table>
<thead>
<tr>
<th>Review</th>
<th>T5-11B Hark Data</th>
<th>T5-11B ICSE Data</th>
<th>SVM Hark Data</th>
<th>RoBERTa-Large Hark Data</th>
<th>RoBERTa-Large ICSE Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Love this! I can share with only the people I choose.</td>
<td>privacy</td>
<td>not-privacy</td>
<td>not-privacy</td>
<td>not-privacy</td>
<td>not-privacy</td>
</tr>
<tr>
<td>Why do you need all my info from my phone to play this! WHACK!!</td>
<td>privacy</td>
<td>not-privacy</td>
<td>not-privacy</td>
<td>privacy</td>
<td>not-privacy</td>
</tr>
<tr>
<td>Why do I need to give my mobile number to obtain my reward? It's not on.</td>
<td>privacy</td>
<td>not-privacy</td>
<td>not-privacy</td>
<td>privacy</td>
<td>not-privacy</td>
</tr>
<tr>
<td>Please add app lock feature using pin or password</td>
<td>privacy</td>
<td>not-privacy</td>
<td>not-privacy</td>
<td>privacy</td>
<td>not-privacy</td>
</tr>
<tr>
<td>becoming bloatware , time to give users permissions to delete sections we never use . still not impressed .</td>
<td>not-privacy</td>
<td>privacy</td>
<td>privacy</td>
<td>not-privacy</td>
<td>privacy</td>
</tr>
<tr>
<td>this does n't give permission to save two route at a time ... .please develop add option to save more route at a time ...</td>
<td>not-privacy</td>
<td>privacy</td>
<td>not-privacy</td>
<td>privacy</td>
<td>privacy</td>
</tr>
<tr>
<td>i cant play youtube because my phone is old please give old phones permission to watch youtube</td>
<td>not-privacy</td>
<td>privacy</td>
<td>not-privacy</td>
<td>not-privacy</td>
<td>privacy</td>
</tr>
</tbody>
</table>

### (b) Examples of Hark Issue Generation Outputs vs. the Baselines

<table>
<thead>
<tr>
<th>Review</th>
<th>Hark Issue Gen</th>
<th>T5 Wikihow</th>
<th>RE-BERT</th>
</tr>
</thead>
<tbody>
<tr>
<td>App monitors your texts and calls. Uses app to exploit your personal information. Download at own risk.</td>
<td>Unauthorized Texts Monitoring, Unauthorized Calls Monitoring, Personal Information Exploitation</td>
<td>Using App to Monitor Your Texts and Calls</td>
<td>Calls, Personal Information, Download At</td>
</tr>
<tr>
<td>First screen and wants my mobile number to send me loads of spam or fake accounts in expect to get me to pay like all the rest</td>
<td>Unwanted Spam, Unwanted Mobile Number Access</td>
<td>Getting a Mobile Number</td>
<td>Send Me Loads, Me</td>
</tr>
<tr>
<td>I think this is a great screen recorder! I specially like the pause button icase your doing something private you dont want other people to see in the video, I say this is THE BEST SCREEN RECORDER EVER!</td>
<td>Pause Button, Private Activity Protection</td>
<td>Using the Screen Recorder</td>
<td>Screen, Pause Button, Video</td>
</tr>
<tr>
<td>Excellent app to offer for sale or buy items that other people no longer need easy to use good results I prefer this app over others that are offered they also keep your safety in mind they remind you not to give out your phone number or your address</td>
<td>Safety Reminders, Phone Number Protection, Address Protection</td>
<td>Using the app is easy and fun</td>
<td>Sale, Buy Items, Results</td>
</tr>
<tr>
<td>No SSL support. Anything you type using this app can be sniffed over the local network. It's otherwise a no-fuss app that works perfectly, but you should probably pass this one up for something more secure.</td>
<td>Typed Data Snooping, No SSL Support</td>
<td>Using a VPN App</td>
<td>Type</td>
</tr>
<tr>
<td>So far the best way to sell stuff. It has a very large market of users which provides great response time once posting an item for sale. Also let's you view what other people rated the seller so some trust can be added to the transaction. On top of that, your own identity and information is secure since you don't have to reveal any phone numbers, email or social media.</td>
<td>Identity Security, No Phone Numbers Required, No Email Address Required</td>
<td>Sell stuff on the internet.</td>
<td>Sell Stuff, Posting An Item For Sale .. View What Other People, Email, Media</td>
</tr>
</tbody>
</table>

### (c) Examples of Hark Theme Generation Outputs vs. GPT-J

<table>
<thead>
<tr>
<th>Issues</th>
<th>Hark Theme-Gen</th>
<th>GPT-J</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abusing Language, Bad Words, Bad Language, Abusing Words, Abusing People</td>
<td>Abuse Concerns</td>
<td>Abusive Language</td>
</tr>
<tr>
<td>Anti Theft, Prevents Theft, Thieves, Anti Theft Feature, Prevents Stealing</td>
<td>Theft Protection</td>
<td>Anti Theft</td>
</tr>
<tr>
<td>Blocking Third Party Cookies, Cookie Opt-Out, Cookie Consent Requests Blocking</td>
<td>Cookie Blocking</td>
<td>Privacy and Security</td>
</tr>
<tr>
<td>Confidential Attachments, Transferring Secret Files, Confidential Documents Taken, Unlocking Files</td>
<td>Data Confidentiality</td>
<td>Privacy Violation</td>
</tr>
<tr>
<td>Email Compromise, Credit Card Compromise, Phone Compromise, App Compromise</td>
<td>Data Compromise</td>
<td>Email Compromise</td>
</tr>
<tr>
<td>Face Recognition, Face Detection, Voice Recognition, Face Capture, Speech Recognition</td>
<td>Face/Voice Recognition</td>
<td>Face Recognition</td>
</tr>
<tr>
<td>Unneeded Photo Gallery Access, Locking Gallery, Required Photo Access, Controlling Photo Access</td>
<td>Photo Access Controls</td>
<td>Photo Gallery Access</td>
</tr>
<tr>
<td>No Tracking, No Trackers, No Logging, No Monitoring, No History Tracking</td>
<td>No Tracking</td>
<td>Privacy</td>
</tr>
</tbody>
</table>