DETECTING TEXTUAL SALIENCY IN PRIVACY POLICY

DS5500 PROJECT REPORT - PHASE 2
NORTHEASTERN UNIVERSITY

Kaushik Nishtala
nishtala.k@northeastern.edu

Shubhanshu Gupta
gupta.shubh@northeastern.edu

December 13, 2021

ABSTRACT

Websites, mobile apps, and other product and service providers share how they gather, use and manage customers' data in the form of privacy policy documents. However, due to their lengthy and complex nature, most people tend to ignore these documents. To counteract the risks posed by this ignorance, we present language models to extract and deliver salient policy information to end-users. We supplement our findings from the previous phase by scaling up the data and model complexities. This led to improved results that outperformed previously reported benchmark scores on classification and question-answering tasks by 6% and 4%, respectively. Besides making our methods completely reproducible, we demonstrate an automated web framework for privacy policy analysis to proliferate the general awareness of data privacy to end-users.

1 Introduction

Data privacy is a matter of importance for an average consumer. It is the right of any individual to have control over how their personal information is collected and used. However, with the advent of technology and social media, any trivial activity (browsing a webpage, using a product or service, or even a mobile app) allows companies to directly or indirectly track users' data. The collected data is then used for various purposes such as advertising, marketing, or research, to name a few. In order to regulate this process, government and regulatory bodies adopt privacy regulations that force these companies to share how they gather, use, and manage users' data. This regulation is primarily accomplished in the form of a privacy policy document.

However, the abundance of information in these documents makes them lengthy and challenging for lay users to comprehend their contents [13]. As the governmental bodies increasingly adopt regulations to promote transparency, it brings an opposite effect in driving users to ignore the policies altogether due to their wordiness and intricacy. To counter this ignorance of these documents and allow users to understand them better, we use machine learning algorithms and natural language processing to derive salient information from these documents to end-users. We do this in the form of the INFORM and the QUERY modules.

INFORM Module: The primary objective of the INFORM module lies in communicating a high level and a more granular breakdown of privacy policies. The functionality is twofold. The first part enables users to explore the general trends and patterns in privacy policy practices using Tableau dashboards. In contrast, the other part involves building a selection of models that can classify privacy policy paragraphs into a set of pre-defined privacy practice categories using supervised machine learning algorithms.

QUERY Module: On the other hand, the QUERY module can enable users to ask policy-related questions using a free form question-answering system for privacy policies. By retrieving the most relevant information from the policy document given a question, this system facilitates searching the target information from a lengthy policy document. To this purpose, we make use of a Question-Answer dataset [1] that provides 714 human-annotated questions written for a wide range of privacy practices.

2 Summary

The first phase of the project dealt majorly with implementing the INFORM module. We developed baseline models with extensive pre-processing and tuning and obtained reliable predictions. However, the models failed to effectively capture essential distinctions between the categories due to the limitations in how the data is represented (TF-IDF and random embeddings) and the models themselves. Thus, we utilize word embeddings and advanced neural architectures in the final phase of this work to gain improved performances on the privacy policy text classification.

In addition, we provide an extensive discussion of our methods and strategies concerning the QUERY module. Finally, we present our system design strategies that integrate both the modules with an interactive web framework. We believe that this could help the end-users explore, extract and query information from previously unseen privacy policies given as input to the system.

Organization. The report discusses the extension of the INFORM module from the previous phase, followed by a delineation of our approach pertaining to the QUERY module. The rest of the paper is divided as follows: in section 3, we briefly review existing and related studies on privacy policy analyses. Section 4 provides a technical description of the methods used in both modules. In section 5, we examine the limitations of our methods and provide a legal disclaimer to their usage in production. Finally, section 6 concludes the presented work and suggests future directions to pursue for further research.

3 Related Work

Many studies on privacy policy analysis have emerged in light of the General Data Protection Regulation (GDPR). According to the Polisis paper [7], the union-based gold standard is used for experiments with Convolutional Neural Networks seeded with domain-specific word embeddings. They scraped, curated, and processed texts of around 130K privacy policies to train these embeddings.

Despite the inspiring work, we believe the study lacks two main elements: They report only the macro-averages and further compute the average of F1 scores on the test set without using a validation set for training. It is also worth noting that a significant amount of information about the training methodology was not shared in the paper, much less about the unbiased performance metrics like the micro averages.

Secondly, a related work by [12] produced baseline models that set the benchmarks for privacy policy classification on the OPP-115 corpus. In the later sections, we show a comparison of our results and indicate how our models outperform the benchmarks with the help of more sophisticated and recently published models and corpus.

4 Methodology

4.1 Baselines

The previous phase of the project focused solely on implementing the INFORM module, where we presented a series of processes, decisions, and artifacts involved. Utilizing the OPP-115 corpus [21], we performed a slew of pre-processing tasks such as iterative splitting [18], cleaning, encoding, and tokenizing the policy texts. We posed the objective as a multi-label classification problem since each segment can contain information for multiple categories. Our primary focus was to establish baseline models while motivating the need for adding complexity from both the dataset and model architecture standpoints. Leveraging the idea of "the strength of weak learners", we developed traditional one-vs-rest classifiers, namely Logistic Regression, Support Vector Machines (RBF), Random Forest, and XGBoost. In addition, we employed Convolutional Neural Networks (CNN) for multi-label text classification with randomly initialized word embeddings.

4.2 Objective

Performing an organized ablation study helped us track and identify multiple facets and parameters of the modeling process that proved helpful in building complex architectures in this phase. Despite producing fairly decent performance results, the baselines failed to learn important distinctions between the categories due to the limitations in data representation and the model bias. Since there is considerable scope for performance improvement, we emphasize the need to add complexity from both the dataset and model architecture standpoints in the following sections.

4.3 Data

In the case of our traditional baseline models, we encoded the data into vector representations using the term-frequency inverse-document-frequency (TF-IDF) method. As for the CNN model, we used randomly initialized word embeddings and trained the model to learn their weights from scratch. However, these methods have a clear disadvantage: they do not capture the semantic relations between the words, ordering, and contextual meaning of individual words. In addition, owing to the low availability and cost of acquiring labeled data, we turn to unsupervised methods where no labeled data is necessary, but only significant amounts of correct text. Thus, we investigate domain-specific non-contextual word embeddings. We delineate the process of training word embeddings in the appendix A.1.

4.4 Classification

4.4.1 Approach

Convolutional Neural Network As previously reported in Phase 1, Convolutional Neural Network (CNN) helps derive meaningful spatial signals by using filters as n-gram feature extractors. They also integrate well with word embeddings to provide information about the semantic space of the text to the model. Thus, we extend the baseline CNN by seeding it with our trained domain-specific word-embeddings without modifying the core model architecture.

Bidirectional Encoder Representations from Transformers To construct a bidirectional representation of the tokens in a given text, the BERT architecture [6] employs numerous layers of transformer encoders [20]. The model is initially pre-trained on vast amounts of unlabeled data, followed by fine-tuning on specific labeled data to solve our downstream task, which is multi-label classification. BERT has a vocabulary size of 30,522 [6] and uses WordPiece tokenization [23], which captures character or subword level information. It helps detect similar prefixes/suffixes/common-roots among tokens.

BERT is pre-trained using two unsupervised tasks [6] - Masked language modeling (MLM) and next sentence prediction (NSP). MLM, a self-supervised pre-training objective, trains the model to predict 15% of the randomly "corrupted" or "masked" tokens in a sentence. Unlike traditional RNNs and autoregressive models like GPT, It allows the model to learn a bidirectional representation of the sentence. In the case of NSP, BERT is trained to recognize when a pair of sentences appear (or do not appear) together in the corpora. We encourage readers to review the original paper [6, 20] for a deep dive into BERT's attention mechanism.

We use two variants of a BERT[1]: The first one is a vanilla BERT-base which has 12 encoder layers, a hidden state size of 768, and 12 attention heads, adding up to 108M parameters. Next, we use RoBERTa (Robustly Optimized BERT) [9], fine-tuned[2] on the PrivaSeer corpus of 1M privacy policies [19] for two epochs using Pytorch Lightning, and achieved a binary cross-entropy loss on the MLM task of 0.1033.

This way, the model learns an internal representation of the English language that can then be used to extract features useful for downstream tasks. Subsequently, we modify the head of the architecture by adding a linear layer with dimensions similar to the number of categories in the dataset (12). Since it is a multi-label

[1]The models are not case-sensitive: it does not make a difference between policy and Policy.

[2]Fine-tuning BERT lasted for 31 hours for two epochs on a single CUDA-backed GPU. Training on the classification task took only a few hours, for five epochs.

Category	Majority-vote based dataset		Union-based dataset	
	CNN (Pre-trained)	CNN (Domain-specific)	CNN (Pre-trained)	CNN (Domain-specific)
	F1	F1	F1	F1
First Party Collection/Use	0.82	0.83	0.80	0.82
Third Party Sharing/Collection	0.76	0.81	0.78	0.79
User Access, Edit & Deletion	0.45	0.65	0.60	0.62
Data Retention	0.15	0.37	0.3	0.48
Data Security	0.81	0.82	0.71	0.73
International and Specific Audiences	0.83	0.83	0.92	0.92
Do Not Track	0.88	1.00	0.33	0.61
Policy Change	0.83	0.89	0.63	0.76
User Choice & Control	0.69	0.75	0.65	0.65
Introductory/Generic	0.63	0.75	0.63	0.65
Practice Not Covered	0.09	0.15	0.31	0.39
Privacy Contact Information	0.64	0.88	0.76	0.75
Micro Average	**0.741**	**0.791**	**0.703**	**0.719**
Macro Average	**0.634**	**0.713**	**0.619**	**0.662**
Weighted Average	**0.72**	**0.746**	**0.685**	**0.701**

Table 1: F1 scores on the test set for CNN (w/ pre-trained word embeddings) and CNN (w/ Domain-specific word embeddings) on both datasets (in decimals).

classification problem, we used sigmoid instead of softmax to get the probabilities and optimized the binary cross-entropy loss.

4.4.2 Evaluation

As previously reported in Phase 1, two gold standard datasets were compiled out of the OPP-115 dataset - union and majority-vote. The union-based dataset includes all expert annotations, and the majority-vote-based dataset only retains the annotations labeled with a given category by at least two annotators.

We conducted six experiments in total. Table 1 presents F1 scores of CNN across all labels with optimal category-specific thresholds for the two gold standard datasets. Table 2 shows the results produced from vanilla BERT and Domain-specific BERT models on the majority-vote dataset[3]. We applied Adam with momentum and dropout for CNN just before the last linear layer and tuned it with randomized search using the Optuna library [2]. BERT is optimized with the default configuration and LAMB optimizer [24].

Considering the scores of the BERT models in the context of the Fleiss expert agreements on the categories (Table 4 in the appendix), we trust them to be very accurate. In addition, we notice that the micro averages outperform macro averages because, for a few categories, the model failed to learn the class weights properly due to their low sample complexity. More specifically, the micro-average minimizes the impact of under-represented categories, whereas the macro-average considers a constant weightage of 1/12 for each category.

In addition, we observe low F1 scores for the Data Retention and the Practice Not Covered categories. This is due to the low sample count[4] in the Data Retention class, as it belongs to just 2-3% of segments in the dataset. Additionally, the Practice Not Covered category refers to all practices not covered by the other eleven categories and therefore has a generic and ambiguous language. Consequently, the model failed to capture the specific vocabulary for this class.

[3]The results for the union-based dataset are shown in Figure 3 in the appendix.

[4]The corresponding label distributions of each of the train, validation and test splits on the majority-vote and union-based datasets are shown in Figures 4 and 5 in the appendix.

Category	Majority-vote based dataset							
	Vanilla BERT				Domain-specific RoBERTa			
	Precision	Recall	F1	Support	Precision	Recall	F1	Support
Data Retention	0.00	0.00	0.00	11	0.75	0.27	0.40	11
Data Security	0.89	0.75	0.81	32	0.83	0.78	0.81	32
Do Not Track	0.00	0.00	0.00	4	1.00	0.75	0.86	4
First Party Collection/Use	0.90	0.87	0.89	181	0.88	0.93	0.91	181
International and Specific Audiences	0.94	0.76	0.84	45	0.91	0.91	0.91	45
Introductory/Generic	0.91	0.69	0.78	58	0.93	0.64	0.76	58
Policy Change	1.00	0.94	0.97	18	0.72	1.00	0.84	18
Practice Not Covered	0.00	0.00	0.00	19	0.82	0.47	0.60	19
Privacy Contact Information	0.85	0.74	0.79	31	0.92	0.71	0.80	31
Third Party Sharing/Collection	0.87	0.87	0.87	142	0.90	0.94	0.92	142
User Access, Edit and Deletion	0.94	0.73	0.82	22	0.95	0.86	0.90	22
User Choice/Control	0.83	0.66	0.74	53	0.82	0.79	0.81	53
Micro Average	**0.89**	**0.76**	**0.82**	**616**	**0.88**	**0.84**	**0.86**	**616**
Macro Average	**0.68**	**0.58**	**0.63**	**616**	**0.87**	**0.75**	**0.79**	**616**
Weighted Average	**0.84**	**0.76**	**0.80**	**616**	**0.88**	**0.84**	**0.85**	**616**

Table 2: F1 scores on the test set for BERT (pre-trained) and RoBERTa (Domain-specific) on the majority-vote dataset (in decimals).

From the Table 2, we observe that the RoBERTA-BASE fine-tuned with PrivaSeer corpus of 1M privacy policies [19] has drastically improved the F1 scores (esp. the macro average) on both datasets. We can see how a good language model can help learn the category weights more effectively irrespective of their sample sizes.

Lastly, we compare our results with two previously reported benchmark scores. We observe that our domain-specific RoBERTa outperforms both the Polisis [7] and [12] by 6% and 1%, respectively, on reported F1 scores.

4.5 Extractive Question-Answering

4.5.1 Problem Formulation

Our extractive question answering system will take in a natural language question as input along with some context and output natural language as an answer. The general design of a QA system [4], as shown in Figure 1, requires two main components: the document retriever that selects the n most relevant documents from a large collection and a document reader that processes these candidate documents in search of an explicit answer span.

Document Retriever The functionality of a typical retriever is twofold [4] - process the query, and use this query to retrieve the most relevant documents/passages. Query processing involves various techniques - such as stop word removal, n-gram tokenization, or extracting named entities as keywords. The processed query is then passed to an Information Retrieval (IR) algorithm that uses indexing, TF-IDF cosine matching, or ranking to retrieve the documents by relevance.

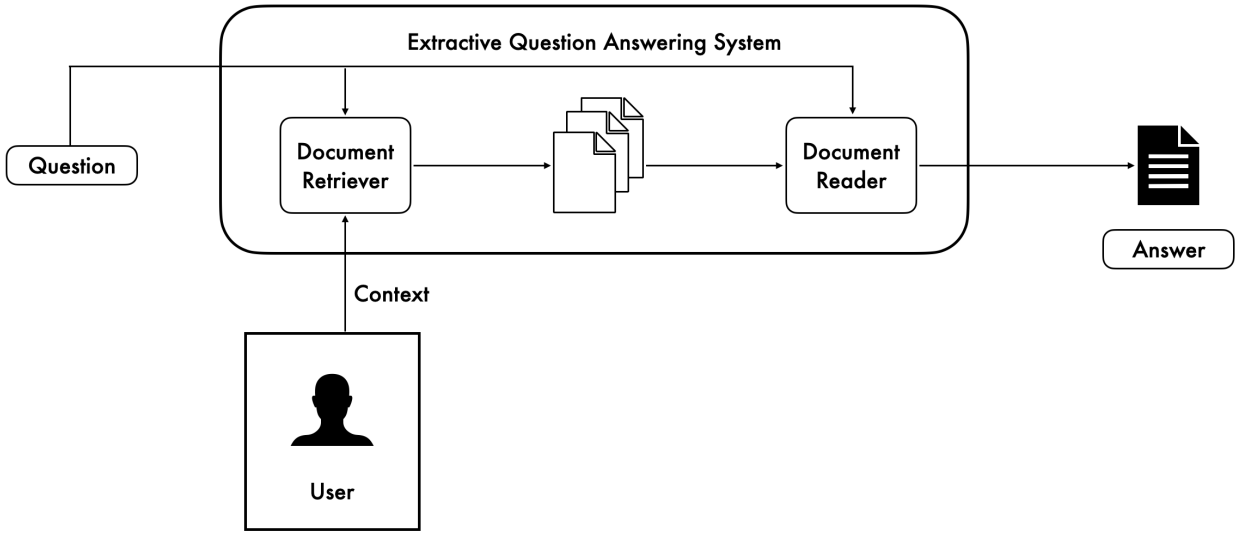

Figure 1: General design of our extractive Question Answering system.

Our system answers questions about a specific privacy policy given as input and is not similar to a general QA system backed by a vast knowledge source. Therefore, for a given privacy policy and a user query, the retriever of our system fetches most eligible passages (also referred to as documents) and sends them over to the reader module. Since the retriever in our case works on only one privacy policy at a time, our efforts on the performance and evaluation of the retriever are limited. Instead, our primary focus lies in implementing an efficient reader module, detailed in the following section. We nevertheless experimented with different IR methods such as TF-IDF, ElasticSearch, Dense Passage Retrieval, and Embedding Retrieval [15]. It is left as an option for end-users to choose a method before submitting their query on the web interface.

Document Reader After obtaining a list of relevant passages[5] from the retriever, the reader attempts to extract a specific text span from the passages that answer the given question. It does this using reading comprehension algorithms, which usually are associated with either feature-based (rule-based, regex pattern matching, parts-of-speech tagging, named entity recognition, etc.) or neural-based [4].

We adopt neural-based approaches, which leverage the idea of semantic similarity of the question and answer, for our QA system. Instead of focusing on individual tokens, these methods learn semantic embeddings from the query and the context, followed by similarity functions to extract the answer span.

The transformer architectures made huge strides on the QA task in recent times. The BERT models fine-tuned on the QA task accept a question and some context (a privacy policy) and return a text span that potentially answers the query. Despite their lack of transparency and high cost of inference, these models generally perform better based on the parameter space and dataset quality. In the following sections, we turn our attention to our training methodologies and evaluation of these models.

4.5.2 Dataset

We use the recently released PolicyQA dataset [1] curated from the OPP-115 corpus [21]. PolicyQA is a reading-comprehension style dataset that includes information about the annotated spans, corresponding policy segments, and the associated Practice, Attribute, Value triples derived from the OPP-115 corpus to form the examples in the dataset. Two annotators were involved in the annotation process to form questions for each specific triple, producing 714 individual questions from their 258 corresponding triples. The distribution of questions' trigram prefixes in PolicyQA is shown in the Figure 6. As shown, the questions are written in a generic fashion and include Yes/No question types (starting with Do/Does) in the majority, followed by

[5]The words passage, document and context are used interchangeably.

Model	Valid		Test	
	EM	F1	EM	F1
Vanilla BERT-base	30.61	59.70	28.25	56.08
Vanilla RoBERTa	32.92	61.51	31.02	58.42
Domain-specific RoBERTa	**35.57**	**63.16**	**32.20**	**59.48**

Table 3: Performance of our models on PolicyQA. The boldface values indicate the best performances.

Wh-question types. Figure 7 shows example questions from each privacy practice category. And more details about the statistics of dataset are referenced in Figure 8.

4.5.3 Approach

We implement three variants of BERT models fine-tuned on the QA task, where two linear classifiers predict the boundary of the evidence or the answer span. Initially, the BERT tokenizer [23] converts the input text into a model-ingestible format. The tokens and input format of the data change with the type of the model. When working with Question Answering, BERT requires the input chunk of text to be in specific format - [CLS] question tokens [SEP] context tokens [SEP] [6].

The format means that, for each segment (passage) of a privacy policy document (context), we must prepend the user query, followed by the next "chunk" of segment tokens. We get an answer for each segment. Since not every policy segment answers the user question, we evaluate the returned answers and rank them based on some pre-defined metrics. If the model cannot find an answer to the question from a given segment, it simply returns the [CLS] token. To sum up, the model returns the answer's start and end scores for each word in the text, and once we have the most likely start and end tokens, we grab all the tokens between them and return them as an answer to the user.

As PolicyQA comes in a similar setting as the Stanford Question Answering Dataset v2 (SQuAD2.0) [16], we pre-train each of the three BERT models on the SQuAD2.0 dataset before fine-tuning on the domain-specific PolicyQA dataset. SQuAD2.0 consists of 150,000 answerable and unanswerable questions posed on a set of Wikipedia articles, where the answer to every question is a segment (or span) of the corresponding passage.

We considered BERT-base (uncased), RoBERTa-base (cased) [9], and a fine-tuned version of the RoBERTa-base with PrivaSeer corpus of 1M privacy policies [19] (previously used for the classification task). RoBERTa (Robustly Optimized BERT), as opposed to vanilla BERT, is trained on a larger dataset (multiplier of 10) and longer sentences, uses dynamic masking pattern and replaces next sentence prediction (NSP) objective with complete sentences without NSP.

After training[6] the models on SQuAD2.0 for two epochs using the HuggingFace's transformers library [22], We applied the BertAdam optimizer and further trained each model with a learning rate of 0.0003 and 20% dropout for five epochs on the PolicyQA dataset.

4.5.4 Evaluation

We use exact match (EM) and F1 score, the two dominant metrics used by many question answering datasets, including SQuAD2.0 to evaluate the performance. Exact match is a strict all-or-nothing metric, whose value is 1 if if the characters of the model's prediction text exactly match the characters of the ground truth. On the other hand, F1 score is computed over the individual words in the prediction text against those in the ground truth. Its value is based on the number of shared words between the prediction and the ground truth.

[6]Fine-tuning each of the three models on SQuAD2.0 took about 3.75 hours per epoch whereas it took about 2.5 hours per epoch on PolicyQA dataset. Additionally, our NVIDIA CUDA-backed workhorse machine has 32GB CPU and 16GB GPU memory, which is sufficient for data processing and training most models on both datasets.

The results are presented in Table 3. Overall, the finetuned RoBERTa-base methods outperform the other variants by 1-4% and 1-3% in terms of EM and F1 score (on the test split), respectively. We notice that fine-tuning the RoBERTa model improves the downstream task performance.

Lastly, we compare our results with the benchmark scores produced by the authors of PolicyQA [1]. We observe that our domain-specific RoBERTa outperforms their best performing model by 4.19% and 3.88%, respectively, on reported EM and F1 test scores.

5 Discussion

Deployment We provide a prototype web application for end-users that comprises of two modules. The first one enables users to broadly visualize different aspects in the privacy policies, powered by Tableau. The second one enables policy specific analyses for policy passage classification and answering questions about privacy policies. More details on the architecture of the system design are provided in the appendix. The application is powered by Flask and deployed using Docker on Heroku and can be accessed at this `link`.

Limitations The performance of the models might be limited by the OPP-115 data size and privacy taxonomy, which does not fully capture certain types of practices. As the training data used for the pre-trained BERT models contain a lot of unfiltered content from the internet, which is far from neutral, these models can have biased predictions.

Legal disclaimer We stress that this project is not intended to replace the privacy policy – as a legal document – with our machine generated predictions and interpretations. Furthermore, the employed machine learning algorithms, despite their overall performances, may occasionally reflect inaccurate trends and patterns within a company's privacy policy.

6 Conclusion and Future work

The project was overall a success. We were able to explore a fascinating problem of proliferating data privacy awareness to end-users using machine learning techniques. All in all, our models produced promising results for both downstream tasks (Classification and Question Answering), outperforming previously published results on the same dataset. Looking back, we had some limitations where we could not make our models as optimal as we thought they could be due to computational and time constraints.

However, moving forward, we would like to find ways to optimize the performance of our models by considering ensembling techniques to leverage the expertise of multiple weak learners and build more efficient and feasible language neural architectures. We were able to get a deep dive into the world of Natural Language Processing and the timeline of doing a project within the field. It was a great cumulative experience for us to take the learned theoretical concepts and translate them to a real-world application.

7 Statement of Contributions

Kaushik Nishtala and Shubhanshu Gupta contributed equally to the project. Shubhanshu Gupta conceived and planned the experiments to train domain-specific word embeddings. In addition, he also carried out experiments on fine-tuning BERT models on the classification task and integrated the components of ASDUS segmenter into the pipeline. Kaushik Nishtala contributed to QUERY module data preparation, designed the model and the computational framework, and finally developed and deployed the web framework.

Github The code used for analysis, experiments and the web framework is provided in a repository at `https://github.com/Kau5h1K/ds5500-userprivacy`.

References

[1] W. AHMAD, J. CHI, Y. TIAN, AND K.-W. CHANG, *PolicyQA: A reading comprehension dataset for privacy policies*, in Findings of the Association for Computational Linguistics: EMNLP 2020, Online, Nov. 2020, Association for Computational Linguistics, pp. 743–749.

[2] T. AKIBA, S. SANO, T. YANASE, T. OHTA, AND M. KOYAMA, *Optuna: A next-generation hyperparameter optimization framework*, 2019.

[3] A. ATHREYA, *Asdus: Automatic segment detection using unsupervised and supervised learning is a system which is designed to detect title and prose segments in html documents.*

[4] M. R. BECK AND A. REED, *Nlp for question answering*.

[5] P. BOJANOWSKI, E. GRAVE, A. JOULIN, AND T. MIKOLOV, *Enriching word vectors with subword information*, arXiv preprint arXiv:1607.04606, (2016).

[6] J. DEVLIN, M.-W. CHANG, K. LEE, AND K. TOUTANOVA, *Bert: Pre-training of deep bidirectional transformers for language understanding*, 2019.

[7] H. HARKOUS, K. FAWAZ, R. LEBRET, F. SCHAUB, K. G. SHIN, AND K. ABERER, *Polisis: Automated analysis and presentation of privacy policies using deep learning*, 2018.

[8] F. LIU, S. WILSON, P. STORY, S. ZIMMECK, AND N. M. SADEH, *Towards automatic classification of privacy policy text*, 2017.

[9] Y. LIU, M. OTT, N. GOYAL, J. DU, M. JOSHI, D. CHEN, O. LEVY, M. LEWIS, L. ZETTLEMOYER, AND V. STOYANOV, *Roberta: A robustly optimized bert pretraining approach*, 2019.

[10] T. MIKOLOV, K. CHEN, G. CORRADO, AND J. DEAN, *Efficient estimation of word representations in vector space*, 2013.

[11] G. MOHANDAS, *Embeddings - made with ml*, (2021).

[12] N. MOUSAVI NEJAD, P. JABAT, R. NEDELCHEV, S. SCERRI, AND D. GRAUX, *Establishing a strong baseline for privacy policy classification*, in ICT Systems Security and Privacy Protection, M. Hölbl, K. Rannenberg, and T. Welzer, eds., Cham, 2020, Springer International Publishing, pp. 370–383.

[13] J. A. OBAR AND A. OELDORF-HIRSCH, *The biggest lie on the internet: ignoring the privacy policies and terms of service policies of social networking services*, Information, Communication & Society, 23 (2020), pp. 128–147.

[14] J. PENNINGTON, R. SOCHER, AND C. D. MANNING, *Glove: Global vectors for word representation*, in Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1532–1543.

[15] S. T. C. B. M. T. . K. B. PIETSCH, M., *Haystack by deepset, an end-to-end nlp framework*, (2020).

[16] P. RAJPURKAR, J. ZHANG, K. LOPYREV, AND P. LIANG, *SQuAD: 100,000+ questions for machine comprehension of text*, in Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, Texas, Nov. 2016, Association for Computational Linguistics, pp. 2383–2392.

[17] R. REHUREK AND P. SOJKA, *Gensim–python framework for vector space modelling*, NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic, 3 (2011).

[18] K. SECHIDIS, G. TSOUMAKAS, AND I. VLAHAVAS, *On the stratification of multi-label data*, in Machine Learning and Knowledge Discovery in Databases, D. Gunopulos, T. Hofmann, D. Malerba, and M. Vazirgiannis, eds., Berlin, Heidelberg, 2011, Springer Berlin Heidelberg, pp. 145–158.

[19] M. SRINATH, S. WILSON, AND C. L. GILES, *Privacy at scale: Introducing the privaseer corpus of web privacy policies*, 2020.

[20] A. VASWANI, N. SHAZEER, N. PARMAR, J. USZKOREIT, L. JONES, A. N. GOMEZ, L. U. KAISER, AND I. POLOSUKHIN, *Attention is all you need*, in Advances in Neural Information Processing Systems, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds., vol. 30, Curran Associates, Inc., 2017.

[21] S. WILSON, F. SCHAUB, A. A. DARA, F. LIU, S. CHERIVIRALA, P. GIOVANNI LEON, M. SCHAARUP ANDERSEN, S. ZIMMECK, K. M. SATHYENDRA, N. C. RUSSELL, T. B. NORTON, E. HOVY, J. REIDENBERG, AND N. SADEH, *The creation and analysis of a website privacy policy corpus*, in Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Berlin, Germany, Aug. 2016, Association for Computational Linguistics, pp. 1330–1340.

[22] T. WOLF, L. DEBUT, V. SANH, J. CHAUMOND, C. DELANGUE, A. MOI, P. CISTAC, T. RAULT, R. LOUF, M. FUNTOWICZ, J. DAVISON, S. SHLEIFER, P. VON PLATEN, C. MA, Y. JERNITE, J. PLU, C. XU, T. L. SCAO, S. GUGGER, M. DRAME, Q. LHOEST, AND A. M. RUSH, *Huggingface's transformers: State-of-the-art natural language processing*, 2020.

[23] Y. WU, M. SCHUSTER, Z. CHEN, Q. V. LE, M. NOROUZI, W. MACHEREY, M. KRIKUN, Y. CAO, Q. GAO, K. MACHEREY, J. KLINGNER, A. SHAH, M. JOHNSON, X. LIU, ŁUKASZ KAISER, S. GOUWS, Y. KATO, T. KUDO, H. KAZAWA, K. STEVENS, G. KURIAN, N. PATIL, W. WANG, C. YOUNG, J. SMITH, J. RIESA, A. RUDNICK, O. VINYALS, G. CORRADO, M. HUGHES, AND J. DEAN, *Google's neural machine translation system: Bridging the gap between human and machine translation*, 2016.

[24] Y. YOU, J. LI, S. REDDI, J. HSEU, S. KUMAR, S. BHOJANAPALLI, X. SONG, J. DEMMEL, K. KEUTZER, AND C.-J. HSIEH, *Large batch optimization for deep learning: Training bert in 76 minutes*, 2020.

A Appendix

A.1 Training Word Embeddings

Word embeddings are vector representations that capture the context of a word in a document, semantic and syntactic similarity, and relation with other words. A popular method called Word2Vec [10] uses a shallow neural network to construct such embeddings. The continuous bag of words (cbow) variant of Word2Vec creates a vector representation by predicting a word given its surrounding context words.

Despite its advantages, Word2Vec treats words as atomic units and fails to consider the internal structure of words. This could lead to poor performance on rarely occurring words or compound words like bedroom or brainstorm. Therefore, we instead consider facebook's fastText [5], which uses character-level information to handle out-of-vocabulary (OOV) or rare words. In specific, each word is represented as a bag of character n-grams in addition to the word itself. This helps preserve the meaning of shorter words, suffixes/prefixes that may appear as n-grams of other words.

A.1.1 Procedure

Initially, we utilized openly available word embeddings, trained on news or Wikipedia corpora (Word2Vec and GloVe [14]) with CNN. However, the performance saturated fairly quickly and produced comparable results as the baseline models. Therefore, we leverage Gensim's [17] fastText implementation to create more relevant embeddings to the current task. Training domain-specific embeddings helps the model pick up vocabulary specific to the privacy policy domain and understand the semantics effectively.

To that end, we utilize a corpus of around 1M raw privacy scraped from websites (PrivaSeer corpus) [19]. With access to this huge corpus of privacy policies, training semantic embeddings using fastText would enable us to use them for the downstream tasks (like classification and Question Answering). We tokenized and examined the vocabulary of this corpus and OPP-115 dataset. There are 1,020,080,504 tokens and 934,891 unique ones in the PrivaSeer corpus in total. We initially noticed that 672 words are seen only in OPP-115 corpus (OOV). After closer inspection, we identified that most out-of-vocabulary words are proper nouns (names of services, companies, products, web-URLs, etc.) and are completely discarded.

We begin with the PrivaSeer Corpus of 1M unlabeled privacy policies. We then convert each raw privacy policy into a format that the fastText accepts. Since we want to train the embeddings with a segment-wide context, we feed each privacy policy as a list of segments to the fastText method. To that end, we employ a previously published method called Automatic Segment Detection using Unsupervised and Supervised Learning (ASDUS) [3], which segments a raw HTML file based on its semantic information. We provide more details about the process in the following section. After producing a list of segments from the raw policies, we train them using the Skip-gram variant of the fastText method with a window size of 10. Subsequently, we obtained the 300-dimensional trained embeddings of all the tokens to be used alongside the labeled datasets.

A.1.2 Segmentation

ASDUS [3] uses natural language processing and machine learning techniques to automatically segregate a given web document into segments, which contain the corresponding title and prose text. The input for ASDUS is a raw HTML file, and it outputs a simplified version of the input HTML file, which contains all title text and prose text. This simple organization makes the detection of segments very straightforward. In order to integrate the segmenter into our pipeline, we built a binary JAR (Java Archive) file from the publicly hosted java and python files and configured its Input/Output settings. We would forward the reader to the relevant reference [3] for more detail on the approach.

A.2 System Design

The application provides an easily accessible interface to the users, thus facilitating high modularity with respect to privacy policy analyses. It handles user input that is given either in the form of text or URL of a policy webpage. Given the URL, the segmenter module scrapes the text by discarding all irrelevant HTML elements including the header, footer, sidebar, CSS and scripts. Finally, the segmenter outputs a series of fine-grained segments (passages) of the privacy policy document to the machine learning models, where they are automatically analyzed.

Our Machine Learning layer is responsible for producing reliable predictions from both the segment classifier and question answering models. For the QA models, we use the Haystack library [15], deployed via REST API. The API uses our hosted gunicorn web server to receive HTTP requests and pass them on to a running instance of Haystack for processing, before returning the query modes results as an HTTP response.

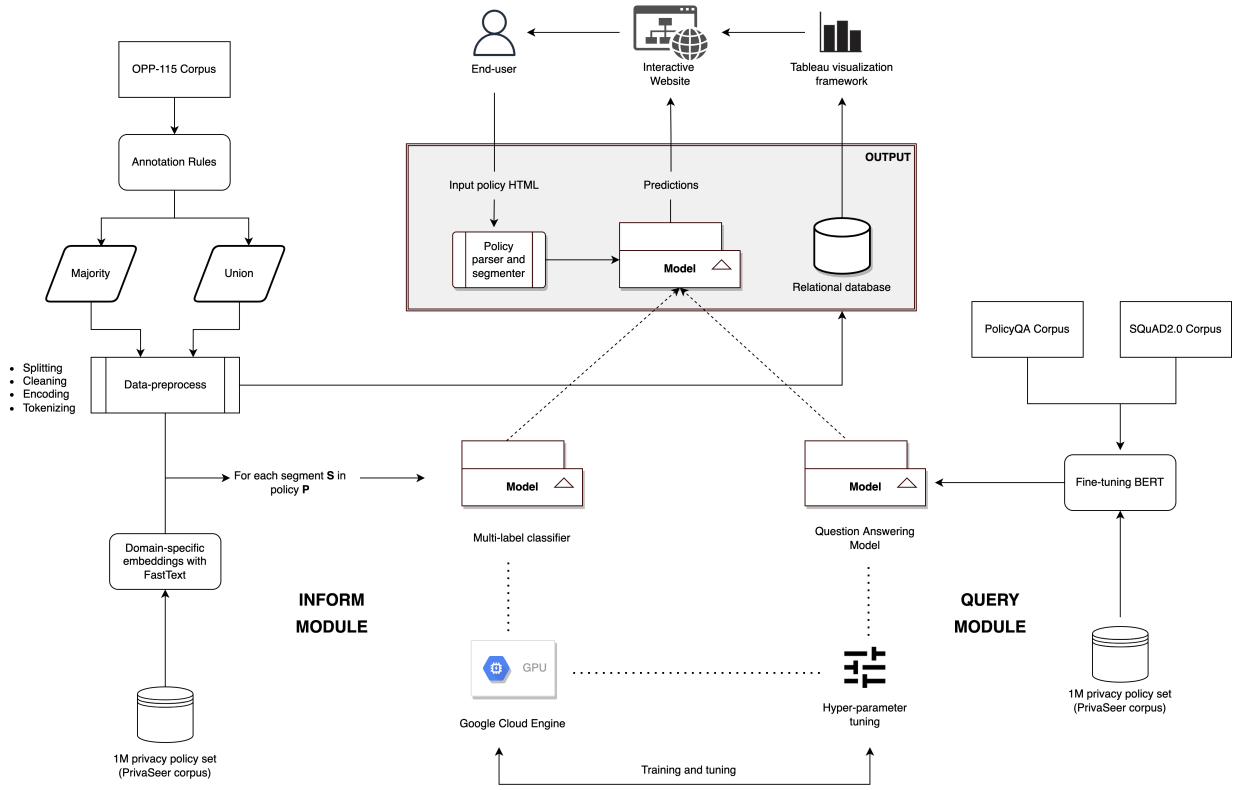

Figure 2: Architecture of the full system comprising of both INFORM and QUERY modules: a visual representation of the methods and artifacts involved in the entire framework.

B Figures

B.1 Fleiss expert agreements on the OPP-155 privacy practice categories

Category	Fleiss' Kappa
First Party Collection/Use	.76
Third Party Sharing/Collection	.76
Other	.49
User Choice/Control	.61
Data Security	.67
International and Specific Audiences	.87
User Access, Edit and Deletion	.74
Policy Change	.73
Data Retention	.55
Do Not Track	.91

Table 4: List of data practice categories in the corpus along with their Fleiss' Kappa values [21].

B.2 BERT Classification results on the union-based dataset

	Vanilla BERT				Domain-specific RoBERTa			
	precision	recall	f1-score	support	precision	recall	f1-score	support
Data Retention	0.64	0.38	0.47	24	0.78	0.58	0.67	24
Data Security	0.80	0.57	0.67	56	0.75	0.71	0.73	56
Do Not Track	0.00	0.00	0.00	5	1.00	0.40	0.57	5
First Party Collection/Use	0.86	0.86	0.86	228	0.84	0.89	0.87	228
International and Specific Audiences	0.87	0.87	0.87	53	0.92	0.91	0.91	53
Introductory/Generic	0.75	0.60	0.67	122	0.79	0.70	0.74	122
Policy Change	0.80	0.83	0.81	29	0.83	0.86	0.85	29
Practice not covered	0.60	0.42	0.49	96	0.65	0.46	0.54	96
Privacy contact information	0.89	0.80	0.84	49	0.93	0.76	0.83	49
Third Party Sharing/Collection	0.90	0.84	0.87	178	0.89	0.89	0.89	178
User Access, Edit and Deletion	0.83	0.71	0.77	35	0.96	0.71	0.82	35
User Choice/Control	0.80	0.51	0.62	95	0.82	0.47	0.60	95
micro avg	0.82	0.70	0.76	970	0.83	0.75	0.79	970
macro avg	0.73	0.61	0.66	970	0.85	0.70	0.75	970
weighted avg	0.81	0.70	0.75	970	0.83	0.75	0.78	970
samples avg	0.86	0.77	0.78	970	0.88	0.82	0.81	970

Figure 3: F1 scores on the test set for BERT (pre-trained) and RoBERTa (Domain-specific) on the union-based dataset (in decimals).

B.3 Majority-vote Dataset Label distribution

Number of unique segments in total: 3471

	TRAIN SET		DEV SET		TEST SET	
	Counts	Percentage	Counts	Percentage	Counts	Percentage
Data Retention	55	1.91%	11	1.79%	12	1.94%
Data Security	147	5.1%	32	5.19%	31	5.0%
Do Not Track	22	0.76%	4	0.65%	5	0.81%
First Party Collection/Use	845	29.3%	181	29.38%	181	29.19%
International and Specific Audiences	211	7.32%	45	7.31%	45	7.26%
Introductory/Generic	273	9.47%	58	9.42%	59	9.52%
Policy Change	83	2.88%	18	2.92%	18	2.9%
Practice not covered	90	3.12%	19	3.08%	20	3.23%
Privacy contact information	142	4.92%	31	5.03%	30	4.84%
Third Party Sharing/Collection	661	22.92%	142	23.05%	142	22.9%
User Access, Edit and Deletion	104	3.61%	22	3.57%	23	3.71%
User Choice/Control	251	8.7%	53	8.6%	54	8.71%
	2422		522		527	

Figure 4: The adjusted label proportions of splits across train, test and validation on the majority-vote dataset.

B.4 Union-based Dataset Label distribution

Number of unique segments in total: 3776

	TRAIN SET		DEV SET		TEST SET	
	Counts	Percentage	Counts	Percentage	Counts	Percentage
Data Retention	109	2.41%	24	2.47%	23	2.37%
Data Security	262	5.8%	56	5.77%	57	5.88%
Do Not Track	22	0.49%	5	0.52%	5	0.52%
First Party Collection/Use	1063	23.55%	228	23.51%	228	23.53%
International and Specific Audiences	247	5.47%	53	5.46%	53	5.47%
Introductory/Generic	569	12.61%	122	12.58%	122	12.59%
Policy Change	134	2.97%	29	2.99%	29	2.99%
Practice not covered	449	9.95%	96	9.9%	97	10.01%
Privacy contact information	226	5.01%	49	5.05%	48	4.95%
Third Party Sharing/Collection	830	18.39%	178	18.35%	178	18.37%
User Access, Edit and Deletion	161	3.57%	35	3.61%	34	3.51%
User Choice/Control	442	9.79%	95	9.79%	95	9.8%
	2645		563		568	

Figure 5: The adjusted label proportions of splits across train, test and validation on the union-based dataset.

B.5 PolicyQA Dataset: Data distribution of question types

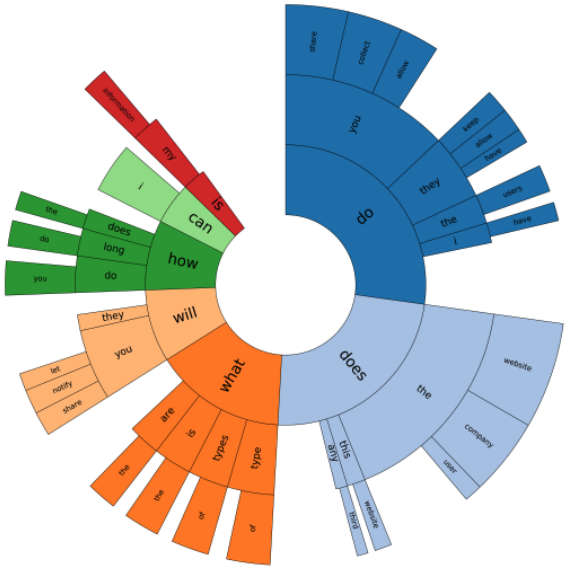

Figure 6: Distribution of trigram prefixes of questions in PolicyQA dataset. [1]

B.6 PolicyQA Dataset: Label distribution of all questions

Privacy Practice	Proportion	Example Question From PolicyQA
First Party Collection/Use	44.4 %	Why do you collect my data?
Third Party Sharing/Collection	34.1 %	Do they share my information with others?
Data Security	2.2 %	Do you use encryption to secure my data?
Data Retention	1.7 %	How long they will keep my data?
User Access, Edit and Deletion	3.1 %	Will you let me access and edit my data?
User Choice/Control	11.0 %	What use of information does the user choice apply to?
Policy Change	1.9 %	How does the website notify about policy changes?
International and Specific Audiences	1.5 %	What is the company's policy towards children?
Do Not Track	0.1 %	Do they honor the user's do not track preference?

Figure 7: OPP-115 categories of the questions in the PolicyQA dataset. [1]

B.7 PolicyQA Dataset: Descriptive statistics

Dataset	Train	Valid	Test
# Examples	17,056	3,809	4,152
# Policies	75	20	20
# Questions	693	568	600
# Passages	2,137	574	497
Avg. question length	11.2	11.2	11.2
Avg. passage length	106.0	96.6	119.1
Avg. answer length	13.3	12.8	14.1

Figure 8: Descriptive statistics of PolicyQA dataset. [1]