openRedact: Automated Text Redaction using Named Entity Recognition

James Carney
School of Information
UC Berkeley
jamescarney@berkeley.edu

Jakob Köhler
School of Information
UC Berkeley
jakob.koehler@berkeley.edu

Rajvardhan Oak
School of Information
UC Berkeley
rvoak@berkeley.edu

Priyam Srivastava
School of Information
UC Berkeley
priyam_srivastava@berkeley.edu

***Abstract*—Today, privacy has become an important consideration when it comes to public release of data. When it comes to releasing documents, governments and companies often need to redact the text in order to remove sensitive information. In particular, the data needs to be sanitized so that it does not disclose identities of individuals. In this paper, we propose openRedact, a framework for automatic document redaction. First, we scrape a public website and create our own annotated dataset which marks the sections to be redacted. We then propose a redaction hierarchy of sensitive attributes, which allows us to sanitize the document to different extents. Third, to address the challenges of naive automatic redaction methods, we explore machine learning methods which can identify sensitive entities in a document. Our results show that random forest based classifier performs better (F-1 score of 70%) as compared to a naive, self-implemented baseline (F-1 score of 57%) Finally, we put all of this together and create a publicly available web-application which serves as an end-to-end redaction pipeline, on which users can (i) annotate their data, (ii) train their own models, and (iii) serve models such that any document provided it is automatically redacted.**

I. Introduction

Text based documents are a common way for publishers to disseminate information to researchers or the public. These documents can include but are in no way limited to: medical records, patient transcripts, and government or private industry documents. Often, the publication of these documents is valuable to both the publisher and researchers. The publisher may find the insights garnered from the researchers valuable and likewise the researchers need the data from the publishers to conduct their research. However, a problem exists if the documents contain privacy sensitive information that cannot be disclosed to the researchers or the public. Typically, publishers address this problem by employing humans who manually annotate private information within the documents. Human readers must be trained on what information should be redacted and then go through the laborious process of manually annotating the documents. Human annotation of personally identifiable information (PII) can be accurate, especially when annotators are trained and working in teams of 2 or 3 with an overlap in documents; such accuracy can average as high as 99.8% [2].

This accuracy, however, does not come cheaply, the cost when factoring in labor and general overhead can average USD 2.76 per PII instance or USD 30.16 per document annotated [2]. Due to these high financial and time costs, publishers may be reluctant to publish sanitized documents despite the fact that they want to contribute data for researchers to use. Text redaction represents an example of the privacy-utility tradeoff: to maintain maximum privacy, extensive parts of the document must be redacted, which has a negative impact on utility. On the contrary, failing to redact sensitive information poses a risk to privacy while increasing utility. Therefore, having a way to classify data as private, with a high accuracy, can effectively balance this tradeoff in privacy and utility. A system that helps automate redaction is therefore an important part of aiding publishers who want to publish documents do so without compromising private information which may be latent in the text.

The concept of privacy in the context of text based documents can take on many meanings. Within the documents there are potentially many different privacy issues. Such issues can include: membership disclosure where the document reveals if a person is specifically a member of the data (or specifically not a member) and attribute disclosure where the data reveals what specific attributes a member of the data has. Membership disclosure and attribute disclosure are often highly dependent on context and not very generalizable to different use cases. We determined that the most important privacy issue that should be addressed in our project is identity disclosure, or being able to determine the specific identity of an individual in the data.

Protecting against identity disclosure is essential to preserving anonymity and an intrinsic part of landmark privacy legislation such as the General Data Protection Regulation (GDPR). In fact GDPR explicitly addresses identify disclosure through defining 'personal data' which should be protected as: '*any information relating to an identified or identifiable natural person ('data subject'); an identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that natural person*[1] Focusing on identity disclosure, in addition to being an obvious core component of privacy, was also attractive due to

[1]https://www.gdpreu.org

its tractability and generalization. Information that constitutes identity disclosure can be clearly defined and is present is many different types of documents.

This paper describes an open tool which aids in the redaction of personally identifying information from published text using machine learning. Our contributions can be summarized as follows:

- **Redaction Hierarchy.** As part of openRedact, we constructed a redaction hierarchy by arranging a set of quasi-identifiers into two levels based on identifying sensitivity.
- **Dataset.** To facilitate the machine learning classification we created a manually annotated dataset consisting of scraped text data with tags for information deemed privacy sensitive according to our hierarchy
- **Model.** For automatically identifying sensitive information we used machine learning based classification methods to learn and identify the sensitive text from a document.
- **Redaction Tool.** To achieve the goal of making a democratized tool to encourage publication, we built a GUI-based application which can take in text and generate a redacted version as output using our machine learning classification models

The rest of the paper is organized as follows. In section II, we present an overview of the related work that has been carried out in the text redaction space. Section III then describes our methodology including datasets, redaction hierarchy and redaction pipeline. Section IV which describes our machine learning approaches for text redaction. We showcase our experimental results in Section V, and conclude in Section VI. Finally, Section VII describes the role played by each member in the team.

II. Related Work

The importance of automatic redaction of privacy sensitive text is prevalent across industries and applications. Bier et al. describes the importance of such a tool in the domains of healthcare, finance, and litigation [1]. An example laid out by Bier et al. examines a real scenario where a legal team was ordered by a Judge to remove any references to "thermal tilt" from all documents in discovery [1]. This posed a large burden to the legal team who had to hire experts to determine what text contained references to "thermal tilt" and then redact all such references [1]. Bier et al. also proposed an automatic redaction tool using linguistic content analysis [1]. The content analysis described in their work is sophisticated, but due to the fact that it does not incorporate learning the generalizability is still rather limited. Other approaches to this problem which are similar to Bier et al. can be categorized as symbolic artificial intelligence, as they rely heavily on predefined rules and symbolic representations of problems. M. M. Douglass et al. used a combination of regular expression, simple heuristics, and lexical look-up tables to redact protected health information (PHI) from free text nursing notes [6]. Their approach had rather high occurrences of false positives and also suffers from the issue of not being very generalizable beyond a specific medical subdomain [6]. There are several off-the-shelf commercial products aimed at automatic text redaction. Such products include Extract Systems and Trapeze by Softworks AI [2] [3]. We did not have access to these products to evaluate and analyze their capabilities and technology due to their proprietary nature. The drawbacks to these products are cost and questionable verifiability of claims made in the marketing material. Both commercial products appear to only offer a semi-automated process which relies heavily on human input to identify sensitive key phrases or text patterns.

The most relevant research to our project approached the privacy redaction problem by using machine learning techniques as opposed to symbolic artificial intelligence. Kniola et al. outlined two approaches; the first was more simple and used regular expression and the second used the spaCy NLP library to build a named entity recognition (NER) classifier [10]. Their research was specific to the medical domain and focused on classifying patient ID, lab measurements, age, gender, race/ethnicity and date and achieved an accuracy of 83.4% [10]. The most robust approach we found in our survey of related work was done by Accenture Technology Labs researchers Chad Cumby and Rayid Ghani in 2011 [3]. They proposed a system which used a learned multiclass classifier to redact private text as well a tool which connects to Microsoft Word which redacts private text using the classifier [3]. Their system of redaction also focused on defining levels of privacy using a utility metric where over-redaction was to be minimized [3]. The primary limitation of their work is that it is patented and appears to only be available internally to Accenture [4]. Our work takes a similar approach to Cumby and Ghani's work with the primary difference being we intend to make our tool open source and freely available.

III. Methodology

We now describe in detail the approach we used to build an application for automated redaction of documents that contain PII. The approach is sufficiently generic, so that it can be applied to various definitions of PII, and can therefore be used in a range of application domains. However, for the sake of this paper we agreed on a single definition, which serves as an example for future applications of this approach. Our contribution is not only the annotated dataset and the resulting classifier, but also the framework that can be used in various contexts.

The framework defines four phases, which result in four deliverables that build on top of each other (see Fig. 1):

- **Phase 1: Hierarchy Definition** Depending on the definition of categories that should be redacted, a hierarchy of sensitive attributes is defined. The output of this phase is the *Redaction Hiearrachy*.

[2]https://www.softworksai.com/our-solutions/redaction

[3]https://www.extractsystems.com/automated-document-classification-and-indexing-software

- **Phase 2: Dataset Generation** Applying the Redaction Hierarchy to a set of documents, a dataset of annotated documents is created. The output of this phase is the *Dataset*.
- **Phase 3: Model Training** Using the Dataset as training data for a machine learning model, a text classifier is created. The output of this phase is the *Classifier*.
- **Phase 4: Text Classification** The classifier is used to categorize words in an unseen document according to our Redaction Hierarchy. The output of this phase is a set of *Predictions*.

These four phases do not necessarily have to be executed in a sequential order: the output of one phase can generate insights about previous phases, so that it might make sense to modify previous deliverables, and start the following phases again. For instance, lacking quality of the dataset can point to deficiencies of the Redaction Hierarchy. In this case, the hierarchy should be improved, and the Dataset Generation phase needs to be repeated. Similarly, a poorly performing classifier could be the symptom of a too small dataset, or to little training time. In these cases, the dataset needs to be enriched, and/or the training phase needs to be repeated. It is therefore important to monitor and evaluate the outputs of each phase carefully.

To support the whole process, openRedact was developed, a web application which can be used to store the dataset, collaboratively create and visualize text annotations, train the classifier and output predictions for new datapoints. openRedact is - together with our dataset and the trained classifier - freely available on GitHub[4], and can be easily deployed - locally or in a cloud environment - using Docker[5], a container virtualization software. We aim to promote the adoption of our framework by making it as easy as possible to install and use openRedact out-of-the-box. The tool complements the iterative nature of the process, since it allows to monitor the intermediate results: the Redaction Hierarchy can be evaluated by visualizing the annotations, and it can be modified if deemed necessary; the classifier can be trained and tested, and new documents can incrementally be added to the dataset if it performs poorly.

We will now go into further detail on the Redaction Hierarchy we used, as well as the resulting dataset. Further information about the classifier and an evaluation of the resulting predictions can be found in Section IV.

A. Redaction Hierarchy

We defined three levels of redaction: H_0, H_1 and H_2. Each level of this hierarchy redacts attributes that are less sensitive than the attributes belonging to the previous levels.

The level H_0 consists of information that is individually personally identifying. That is, access to an attribute from the H_0 class is enough to identify an individual without the need for any auxiliary information. Examples of information falling under this category are names, Social Security Numbers, bank account details, or any critical (identifying) medical conditions. We also include email addresses; although one may use aliases, a real email address is enough to identify the person.

The level H_1 consists of information which may be used in conjunction with some other auxiliary information to identify a person. We view these attributes as those which can significantly narrow-down the list of possible identities. Examples of H_1 category information are dates of birth and death, national origin, citizenship and employer name. We redact job titles only if they appear in combination with the employer name.

We denote all other kinds of information as H_2. This level in the hierarchy is public information; this does not have to be redacted and is safe for release. A full schema of our redaction hierarchy is depicted in Fig. 2.

We would like to draw attention to a number of subtleties here. First, note that the hierarchy we present is not exhaustive; there are definitely other attributes that fit into each of these levels. Depending on the domain of application, and the annotated dataset, this Redaction Hierarchy needs to be adapted, or completely replaced. Second, depending on the domain, some attributes may move across categories. For instance, if the text corpus we are sanitizing is about authors, then we have to redact any notable works they produced; in our framework, this would have been H_2 level information, but now it becomes H_0. Moreover, additional levels H_i can be added to the hierarchy in order to provide a more fine grained distinction between identification capabilities.

B. Dataset

We were unable to find an open-source dataset of annotated sentences. Most of related work (as can be seen in Section II), has been done on medical and legal data. As a result, it has not been released for open-source research. We believe that this dearth of datasets is problematic, and hence we also contribute a dataset of text complete with annotations.

To simulate data that may have been a candidate for release, we scraped articles from the English Wikipedia. We used a list of people who passed away between 2014 and 2016[6], and scraped a random subset of the articles, in order to ensure a certain level of variety in profession and nationality of the subjects. Using openRedact, We then manually annotated the data to identify the sections of text, which - according to our Redaction Hierarchy - belong to the H_0 or H_1 classes. Sections belonging to H_2 (public information) have been left unannotated. Overall, we annotated 100 articles with a total of over 30694 words.

Note that the main purpose of our dataset is to demonstrate our classification approach. This dataset does not contain

[4]https://github.com/jkhlr/openRedact

[5]https://www.docker.com

[6]https://en.wikipedia.org/wiki/Category:2016_deaths

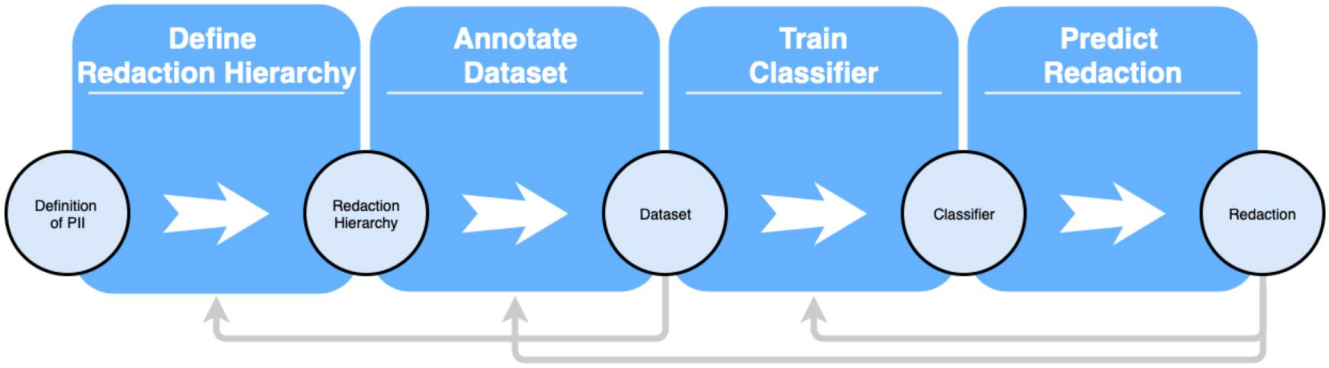

Fig. 1. Flow Diagram of the process; The four phases and deliverables.

H_0 - stand alone identifiers

- Name
- Home Address
- Email
- Critical Government IDs (N/A to our dataset)
  - SSN
  - Driver License
  - Passport Numbers
-
- Critical Financial (N/A to our dataset)
  - Bank accounts
  - Credit Cards
- Username (N/A to our dataset)
- IP Address (N/A to our dataset)
- Device IDs (N/A to our dataset)
- Phone number

H_1 - auxiliary identifiers

- All QIDs in H_0
- Date of Birth
  - Place of Birth - including Country, and all sub locations
- Date of Death
  - Place of Death - including Country, and all sub locations
- Race
- Critical Medical Information - medical conditions that are rare
- Critical Salary
- Employer
- Job Title (only when in-conjunction in with Employer)
- Credit Score
- Marital Status
- Education
- Religion
- Citizenship
- Political Affiliation

Fig. 2. The Redaction Hierarchy used to annotate our dataset.

instances of all the identifiers we described in Section III-A (but mainly names, dates of birth and death, citizenships, nationalities, and professions. The dataset could greatly be improved by adding and annotating texts from sources other than the Wikipedia, primarily to achieve a wider variety in writing style. For productive applications, a dataset should ideally be comprised of texts similar to the documents that should later be redacted by the classifier.

IV. Models

A. Preliminaries

In this section, we provide an overview of some preprocessing and machine learning concepts which are essential for fully comprehending the approach we propose.

Text Processing. In text classification there are two main steps: (i) transforming the text into a numerical representation so that the text may be consumed by a machine learning algorithm, and (ii) applying appropriate machine learning methods on this numeric representation for classification. Algorithmic and methodological choices made in both of these steps affects the overall performance of the classifier [7]. The first step, also known as *feature extraction*, is generally performed using term frequency-inverse document frequency [7], or word embeddings [11]. An *embedding* is a fixed-length vector representation of a word. Words which are similar in meaning are close to each other in the vector space. A recent approach, known as contextual embeddings [5] computes embediings based on context. Therefore, the word *die* has a different embedding in the sentence *'I rolled a die and it was a six'* versus the sentence *'It was painful to watch him die'*.

Machine Learning. The class of machine learning algorithms we use for this task is known as *supervised machine learning*. In this approach, we have data and associated ground truth labels. Popular supervised algorithms include support vector machines, decision trees, random forests and deep neural networks used as classifiers. A network contains multiple layers of *neurons*, which are essentially computational units. Data flows from the input layer, undergoes several transformations through the intermediate layers (also called as *hidden layers*), and produces an output which is then used to determine the predicted class label. Long Short-Term Memory (LSTM) [8] excels at processing sequential data (like text) because of its ability to remember past inputs. Decision Tree is a data structure in which the leaf nodes refer to class labels.

All the non-leaf notes represent certain input features, and their children represent the various possible values taken by the features. Basically, a decision tree implements a rule-based classification system. For classifying a particular data point, we begin traversing from the root of the tree till we reach a leaf node. In practice, an ensemble version of decision trees called *Random Forest* is often used for classification tasks. The benefit of using a multitude of decision trees instead of a single one is that the former overcomes the problem of *overfitting* (a model being tuned to the specific dataset and thus leading to loss of generalizability) caused by the latter.

B. Baseline

We were unable to find any publicly available baselines for document redaction. Some patents have been published [4], but the datasets have not been released. Therefore, we are unable to re-implement those methods to compare against our data. In addition, most commercial tools are proprietary and hence we were unable to compare our performance against them either.

We use a naive, self-defined baseline model for comparison. We use named entity recognition (NER), and identify sensitive entities. An open source package called SpaCy [9] provides NER implementation to identify entities such as `PERSON`, `DATE`, `GPE`, etc. In our naive model, we just identify these entities and mark those sentences as H_0, H_1 and H_2 accordingly. For example, we redact *all* names as H_0, all dates as H_1 and so on.

This baseline involves no machine learning approach. Furthermore, it suffers from an important drawback; it can redact text which is a quasi-identifier, but not a threat to identity disclosure. For example, in the sentence *'Narendra Modi landed at the Indira Gandhi international airport on Sunday'*, our naive model would redact both 'Narendra Modi' and 'Indira Gandhi'. However, the latter name refers to the name of an airport, and is not resulting in any identity disclosure. This drawback can be addressed by a model that *learns* when entities result in attribute disclosure.

C. Machine Learning Models

We explored two main approaches in machine learning based classification: sentence level classification, and entity-level classification. In this section, we describe our efforts at both these approaches.

1) ***Sentence-Level Classification****:* In this approach, we predicted a label (H_0, H_1 and H_2) for every sentence. We did not focus on identifying individual entities belonging to each class. This was a first step in our redaction mechanism; we would redact entire sentences which have the sensitive attributes.

For classifying sentences, we used the state-of-the-art language model called BERT [5]. This model has shown excellent performance in fine-tuning on downstream tasks like text classification. We added a softmax layer on top of the BERT architecture, and fine-tuned both BERT and our layers jointly on our annotated dataset. We had around 2000 annotated sentences, out of which we used 1650 for training, and the remaining 350 for testing our classifier. Both the training and test sets were stratified by class of sentences; both sets had an equal ratio of private versus non-private sentences.

2) ***Entity-Level Classification****:* In this approach, we classified each word in each article as private (H_0 or H_1) or non-private. This methodology can help us simply redact specific words which may lead to identity disclosure.

First, we used an ensemble method to find the private entities in the document. This ensemble method includes our trained model on our annotated dataset. We use our trained model and Spacy's inbuilt `en_core_web_sm` model to predict the private entities in parallel. Our model recognizes two entities $H0_{pred}$ and $H1_{pred}$. Spacy's `en_core_web_sm` model has 18 inbuilt entities.

All the entities are then passed into a classifier which classifies the entities as H_0 and H_1. The classifier is built using Random Forest algorithm with 10 decision trees. The output from the classifier is then fed into the redactor which redacts the private entities based on their levels.

V. Results

In this section, we discuss our results. We begin by setting our evaluation metrics, followed by results from the sentence classification approach. Then, we present our results from the entity-level approach, which we actually leverage in our redaction tool. Finally, we discuss the findings and implications of these results.

A. Evaluation metrics.

To evaluate our model, we will use two key metrics: Accuracy, and F-1 score, which could be calculated from the number of true positives (TP), true negatives (TN), positives (P) and negatives (N) [12]. Table I illustrates how these are calculated.

Accuracy is the fraction of the number of correct predictions over the total number of samples, calculated as $(TP + TN)/(P + N)$. *Precision*=TP/P' indicates the percentage of entities from the redacted ones which were actually sensitive. *Recall*=TP/P measures the percentage of sensitive entities which were actually redacted by the model. The F-1 score is defined to be the harmonic mean of the precision and recall. There are equivalent definitions for these metrics at the sentence level as well.

		True Label		
		Positive	Negative	Total
Predicted Label	Positive	TP	FP	P'
	Negative	FN	TN	N'
	Total	P	N	

TABLE I
CONFUSION MATRIX ILLUSTRATION.

B. Sentence Level Classification

As described in Section IV-B, we use a naive NER based model as a baseline. The model has an accuracy of just 64% with F-1 score of 64%. Because this is a multiclass setting, these results reported are the weighted average of the metrics for each class. On the other hand, the BERT-based classifier

works relatively better. The accuracy was 83% with an F-1 score of 76%. These results are summarized in Table II.

MODEL	Accuracy	Precision	Recall	F-1 Score
Baseline	66%	64%	64%	64%
BERT	83%	80%	75%	76%

TABLE II
SENTENCE-LEVEL CLASSIFICATION RESULTS

C. Entity Level Classification

As described in Section IV-B, we use a naive NER based model as a baseline. The model has an accuracy of just 84% but with a low F-1 score of 57%. Because this is a multiclass setting, these results reported are the weighted average of the metrics for each class. On the other hand, the BERT-based classifier works relatively better. The accuracy was 92.5% with the F-1 score of 70% being significantly better than the baseline. These results are summarized in Table III.

MODEL	Accuracy	Precision	Recall	F-1 Score
Baseline	84%	66.5%	50%	57%
Our Ensemble	92.5%	83%	60%	70%

TABLE III
ENTITY-LEVEL CLASSIFICATION RESULTS

D. Discussion

From our results, we see that the sentence-level classifier shows better performance than the entity level classifier. While the former shows a better F-1 score, it compromised on utility; we would have to redact the entire sentence even if contained a single sentence attribute. Other information in the sentence (which may not be sensitive) would have been lost. This is a classic case of the privacy-utility trade-off where we achieve privacy at the cost of low utility. In the favor of increased utility, we use the entity level model in our final website.

We can also observe that our entity-level model shows a boost of 23% in the F-1 score, but less than 10% improvement in the accuracy. The reason for this is high imbalance in the training data. A large part of the entities are non-private; therefore the model can achieve high accuracy simply by leaning towards the majority class. The true improvement, in such a scenario, can be seen in the F-1 score.

Note that these results show one of the use cases of openRedact. Because our approach and the tool we provide is generic, one can plug in their own data, classifiers and/or pretrained models specific to their domains and redact documents.

VI. Conclusion and Future Work

We have proposed a general framework that can be used to automate the redaction of documents, based on a machine learning approach. We have shown the utility and feasibility of this framework by applying it to the task of redacting PII from a set of Wikipedia articles. In the course of this endeavor, we have defined a two-level Redaction Hierarchy for PII, and compiled a dataset of 100 Wikipedia articles, with each word categorized according to the Redaction Hierarchy. To support future applications of our framework, we have built openRedact, an open-source web-application that helps with the compilation of annotated text-datasets and the training and evaluation of classifiers. All our contributions have been publicly released of GitHub.

Building on our results, we can propose future work in various areas: First, in order to improve prediction performance, the dataset could be enhanced with texts from additional sources, in different writing styles, or on different subjects, following our proposed Redaction Hierarchy. Alternatively, additional Redaction Hierarchies could be defined, which could then be used as a base for new datasets. Since the performance of the classifier is also highly dependent on the volume of training data, crowd-sourcing-platforms such as Amazon Mechanical Turk[7] could be used to annotate large amounts of texts. The Redaction Hierarchy can be useful in this case to brief people about the desired annotations. Moreover, the modular nature of openRedact also allows for easy addition of alternative classification approaches. A promising method for identifying relevant redaction candidates would be contextual embeddings such as BERT[8], or the employment of Part of Speech recognition, which goes beyond the scope of this paper.

VII. Individual Contributions

- **James Carney** was the lead in the dataset collection and slide decks. He developed the web scraper, and made significant contributions to the report.
- **Jakob Köhler** was the lead in developing the web application. He also contributed to building the classifier and writing the report.
- **Rajvardhan Oak** was the lead on implementing baselines and the report. He also contributed to the slide decks.
- **Priyam Srivastava** was the lead on developing the ML models. He also made significant contributions to the report.

Acknowledgements

This work would not have been possible without the support of Daniel Aranki, postdoctoral scholar, EECS, UC Berkeley. The authors are thankful for his continuous feedback and encouragement. We would also like to thank Qutub Khan Vajihi and Dheeraj Khandelwal who helped is in annotating the dataset.

References

[1] E. Bier, R. Chow, P. Golle, T. H. King, and J. Staddon. The rules of redaction: Identify, protect, review (and repeat). *IEEE Security Privacy*, 7(6):46–53, Nov 2009.

[7] https://www.mturk.com

[8] https://arxiv.org/pdf/1810.04805.pdf

[2] D. S. Carrell, D. J. Cronkite, B. A. Malin, J. S. Aberdeen, and L. Hirschman. Is the juice worth the squeeze? costs and benefits of multiple human annotators for clinical text de-identification. *Methods of information in medicine*, 55(04):356–364, 2016.

[3] C. Cumby and R. Ghani. A machine learning based system for semi-automatically redacting documents. In *Twenty-Third IAAI Conference*, 2011.

[4] C. Cumby and R. Ghani. Classification-based redaction in natural language text, Jan. 20 2015. US Patent 8,938,386.

[5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[6] M. Douglass, G. Cliffford, A. Reisner, W. Long, G. Moody, and R. Mark. De-identification algorithm for free-text nursing notes. In *Computers in Cardiology, 2005*, pages 331–334. IEEE, 2005.

[7] J. Han, J. Pei, and M. Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

[8] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[9] M. Honnibal and I. Montani. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear, 2017.

[10] L. Kniola, U. W. Song, and E. Xogene. Text as data in the context of anonymising clinical study reports.

[11] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[12] Wikipedia contributors. Confusion matrix — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Confusion$_matrix oldid = 906886050, 2019. [$Online$; $accessed 28 - August - 2019$].