A Privacy-Preserving Approach to Extraction of Personal Information through Automatic Annotation and Federated Learning

Rajitha Hathurusinghe[1,*], **Isar Nejadgholi**[2,*], **Miodrag Bolic**[3]
[1,2,3]University of Ottawa, Ottawa, Canada
[2]National Research Council Canada, Ottawa, Canada
[1,3]{rhath050,Miodrag.Bolic}@uottawa.ca
[2]isar.nejadgholi@nrc-cnrc.gc.ca

Abstract

We curated WikiPII, an automatically labeled dataset composed of Wikipedia biography pages, annotated for personal information extraction. Although automatic annotation can lead to a high degree of label noise, it is an inexpensive process and can generate large volumes of annotated documents. We trained a BERT-based NER model with WikiPII and showed that with an adequately large training dataset, the model can significantly decrease the cost of manual information extraction, despite the high level of label noise. In a similar approach, organizations can leverage text mining techniques to create customized annotated datasets from their historical data without sharing the raw data for human annotation. Also, we explore collaborative training of NER models through federated learning when the annotation is noisy. Our results suggest that depending on the level of trust to the ML operator and the volume of the available data, distributed training can be an effective way of training a personal information identifier in a privacy-preserved manner. Research material is available at https://github.com/ratmcu/wikipiifed.

1 Introduction

Extraction of Personally Identifiable Information (PII) from unstructured text is a crucial task in many industries such as healthcare (e.g (Li and Qin, 2017; Kushida et al., 2012)) legal documents (Oksanen et al., 2019), mining of user-generated data (Mosallanezhad et al., 2019) and publication process (Aura et al., 2006). PII is a laborious task, often necessary for de-identification purposes, among other applications. For example, the extracted information can be used for indexing of documents, categorization and other applications. Identification of PII elements is a laborious task that can be automated by deploying Named Entity Recognition (NER) models (Hassan et al., 2018; Graliński et al., 2009). We formulate PII recognition as a NER task that extracts predefined PII entities. Our goal is to develop NER models to decrease this task's cost by preprocessing documents before manual information extraction.

Supervised machine learning approaches such as Conditional Random Field (CRF) models, Support vector machines, and extensive feature engineering based on lexical and phrase embeddings have been used to train NER systems (Luo et al., 2015; Passos et al., 2014; Retinov and Roth, 2009). With improvements of deep learning models, recurrent neural networks and specially LSTM models became the default model for training NER systems (Chiu and Nichols, 2016). Recently, a combination of pre-trained transformer-based language models and linear or recurrent prediction layers achieved the state-of-the-art in most NER tasks (Dai et al., 2019; Fraser et al., 2019).

Training a NER model for extraction of PII demands a massive corpus of text, rich in personal information, which raises privacy concerns in the process of data annotation and model training. Although NER models achieve high performance in cross-validation settings, the generalization of off-the-shelf models remains poor (Fu et al., 2020). For training a robust PII recognizer, a customized domain-specific annotated dataset is needed (Chen et al., 2015). In this work, our goal is to bring privacy to the front line of designing a PII extractor from dataset creation to model training.

We consider a scenario where an institution intends to build an assistant tool to decrease the cost of manual PII extraction. We assume that the institution has accumulated documents alongside their corresponding PII fields over the years, but PII elements are not necessarily marked within the text. This is the typical case for many institutions (such as hospitals and banks), which have manually extracted PII elements for years. For example, a hospital has access to patients' names and ages for every specific health record. However, the locations of occurrences of name and age within the documents are unknown. Also, the name and age can

[*] Equal contribution.

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 36–45

https://doi.org/10.26615/978-954-452-056-4_005

come in various forms and lengths when mentioned in free text. To build a useful training dataset for a PII recognizer, the hospital needs to mark phrases related to name and age in the free text through text mining. However, sharing this data for annotation and training involves privacy considerations. We consider the following steps to ensure our process is compliant with the privacy of data subjects.

- Annotating the free text programmatically without the need for sharing the data for human annotation.
- Distributed storing of annotated documents so that the data can be kept in authorized locations.
- Remote training of the PII extraction model without the need for sharing annotated documents with machine learning practitioners.

To conduct a reproducible research, we show the feasibility of the proposed approach on a dataset collected from Wikipedia and share the created dataset and results with research community. Our contributions are as following:

- We create and release an automatically labeled dataset comprised of 77703 sentences from Wikipedia biography pages annotated for 5 classes of personal information.
- We develop a method for remote training of a transformer-based model on distributed datasets, using PySyft platform.
- We explore the impact of label noise and dataset size on the performance of remotely trained NER models.

2 WikiPII Dataset

Our goal is to create and annotate a customized textual dataset for training a PII extractor (the scenario described in Section 1). Approaches like snorkel (Ratner et al., 2017) had been embracing noise of automatic annotations and compensated the noise by adding to the volume of inexpensive data. We took the same approach for annotating Wikipedia pages and benefited from the fact that a version of entities was available in the infobox. We used the infobox to generate noisy and inexpensive data annotations, whereas snorkel uses multiple noisy parallel annotation functions and weak-supervision from alternative sources.

Figure 1: Infobox a) viewed on web page, b) in HTML format, c) converted to a dictionary

We collected our data from Wikipedia biography pages because 1) they are rich in terms of PII, 2) with the infobox available on each page, they comply with our assumption of having access to extracted PII, 3) they are publicly available and can be shared and used as a benchmark for research purposes. Similar automated annotation tasks such as Nothman et al. (2013) utilized a broader set of Wikipedia pages and contain general CoNLL style (Sang and De Meulder, 2003) (location, organization, person, miscellaneous) classes of entities and does not focus on granular personal information as ours. We refer to this dataset as WikiPII and release this data for further research.

2.1 Data Collection

We scraped our raw textual data from biography page entries of living people in Wikipedia (about 900K pages). For programmatic annotation of each page's textual body, we first read the HTML-coded infobox and converted it to a PII element dictionary, using the BeautifulSoup[1] package. An example of this conversion is demonstrated in Figure 1.

Next, we normalized the similar entity types to acquire consistent entity types across all pages. For example, the spouse's name can come under the titles' Spouse',' Spouse(s)' and' Spouses', which are all normalized to the 'SP' tag. After normalization, we manually inspected the entities and chose the ones with high coverage in the dataset. At last, we decided to include BD (date of birth), PR (names of parents), SP (names of spouse(s)), CH (names of children) and ED (terms of education institutes attended). Our final tags and their corresponding

[1] https://www.crummy.com/software/BeautifulSoup/bs4/doc/

infobox entries are shown in Table 1 [2].

Tag	Corresponding entries in infobox
Birth Date (BD)	'Born', 'Born:'
Parents (PR)	'Parent', 'Parent(s)', 'Parents', 'Father', 'Father's name', 'Mother', 'Mother's name
Spouses (SP)	'Spouse', 'Spouse(s)', 'Spouses'
Children (CH)	'Children'
Education (ED)	'Education', 'High school', 'High school:', 'Law School', 'School', 'Schools', 'College', 'College(s)', 'Colleges', 'Alma mater','Almat mater'

Table 1: Tags included in our dataset and corresponding entities in infobox.

2.2 Labeling of Entities in Text

Once the PII was extracted from the infobox, we had to locate them in the text and generate a tag for each word to create an annotated dataset. This step's main challenge is that the mentions of entities in the text might be variations of the ones extracted from the infobox.

We parsed the textual body into sentences and removed citation brackets and numerals using regular expressions. For each tag shown in Table 1, we develop a function that takes the extracted phrase from infobox and locates that phrase within the free text. We combine two different methods of matching to get a more accurate match. Our entities are subcategories of places, organizations, persons and dates, which SpaCy already covers. We leveraged the part-of-speech and named entity recognition capabilities of the SpaCy[3] package to find noun chunks, person names, locations, organizations and dates. Then, depending on the type of the entity, we choose a subset of extracted phrases and use fuzzy string matching[4] to find the closest phrase to our target phrase. For example, for the tag, ED (education), we extract all the organization names by SpaCy. We then use fuzzy matching to find the variations of the education institute pulled from the infobox.

[2]We first extracted birthplace from the infobox, but places are mentioned in several formats such as town, province, country, etc. We omit this entity in the final tagging.

[3]We used the *en_core_web_lg* pipeline from https://spacy.io/usage/facts-figures#benchmarks, which is highly accurate in NER task and optimized in terms of speed.

[4]We utilized the implementation published at https://github.com/axiak/fuzzyset/ for fuzzy matching.

We used the BIO scheme for tagging of words. NER is a sequence to sequence learning task that predicts a label for each word, specifying whether the word is within or outside an entity and the entities' type. In the BIO format, the tags 'B_', 'I_', and 'O_' mark the beginning, inside and outside of an entity, respectively. For example, 'B_CH' specifies the beginning of a phrase tagged as 'Children'. The combination of these tags specifies the boundary and tag of the extracted entity. Therefore, error analysis of an NER task is based on errors in tag and boundary detection. These error are reflected in the evaluation metrics described in Section 3.

2.3 Manual Annotation

To evaluate the quality of the programmatic annotation, we manually annotated a subset of the pages. We selected pages that include highest numbers of entities and made sure that the manually annotated dataset contains 50 to 100 mentions of each class. Manual annotation is done by re-annotating the entities already found by the automated annotator. A human annotator can choose to confirm, reject or correct the labels created by the automatic annotation. We designed a user interface for the manual annotation where the annotator had access to the infobox elements and their corresponding tags. An example of the designed annotator user interface is shown in Figure 2.

Figure 3 shows examples of common mistakes in the automatic annotation. Entities extracted from the infobox are shown in the left column. The yellow entities are missed by automated annotation and corrected by the human annotator. These entities are missed because they are missing from the infobox. Also, since the automatic annotation does not consider the context, it cannot resolve ambiguities. In the example of Figure 3 'Troy' is a city name but is tagged as CH (children) since it appears as a child name in the infobox. Also, 'Harvard University', which is tagged as an education institute, is not a PII element for the main subject of the page but an affiliation of someone else. In manual annotation, 'Troy' and 'Harvard University' will be corrected and not tagged as an entity.

2.4 Statistics of WikiPII dataset

Our data source contains over 900K entries. Our annotation method could only use a little over 23K entries due to formatting changes in the Wikipedia pages where infobox is not available. We filtered the sentences that do not include any of our target

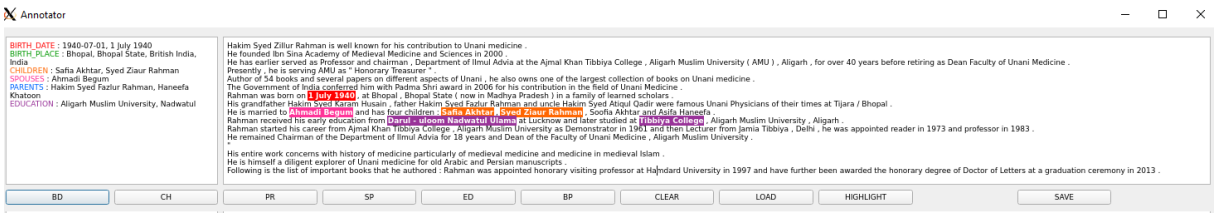


Figure 2: Annotator UI for manual annotation.

Figure 3: Examples of mistakes in automated annotation.

entities. The dataset contains a large number of PII instances belonging to over 23K individuals worldwide belonging to 5 classes. Separate splits of the created dataset and numbers of entities contained are presented in Table 2.

3 Evaluation of PII extraction

Averaged F-score is a common metric to evaluate a NER system. However, the definition of a True Positive and a True Negative prediction is not always trivial. Since identifying an entity involves finding both the span and type of the entity, some of the system's predictions can be partially correct. Multiple evaluation schemes have been developed. Shared tasks such as IREX (Sekine and Isahara, 1999), and CoNLL (Sang and De Meulder, 2003) only gave credit to the exacted entities with the exact type and boundary matches. Other works have adopted *type matching* or *partial matching* evaluation schemes, which reward partially correct entity extractions (Tsai et al., 2006; Chinchor and Sundheim, 1993; Segura Bedmar et al., 2013). Learning-based evaluation methods are developed to predict the user experience in specific tasks (Nejadgholi et al., 2020).

PII extraction is a sensitive task, and a fully automatic system cannot be reliable. Instead, the output of such systems are used to augment the performance of manual PII extraction. In practice, when a human is in the loop, partial matching can reduce the manual effort of PII extraction. We adopted the metrics introduced by the MUC-5 task (Chinchor and Sundheim, 1993), and SemEval-13 task 9 (Segura Bedmar et al., 2013) and implemented the following evaluation metrics ordered in terms of strictness:

- **Strict Matching:** rewards a prediction only if boundary and type of entity match with gold standard label. This metric evaluates the system in a fully automated PII extraction setting.
- **Exact Boundary:** rewards a prediction if the boundary of extracted entity matches the gold standard labeling. This metric evaluates the system where the human annotator relies on boundaries predicted by the system and only corrects the label if necessary.
- **Type Matching:** rewards the strict matches and partially ($\times 0.5$) rewards the extracted entities where the type is correct and boundary overlaps with the gold standard. This metric evaluates the system where the human annotator relies on types predicted by the system and only corrects the boundary if necessary.
- **Partial Boundary:** rewards strict matches and partially ($\times 0.5$) rewards where the boundary overlaps with the gold standard label regardless of type. This metric evaluates the

Data split / annotation method	Pages	Sentences	BD	PR	SP	CH	ED
training/automatic	20039	77703	16883	6326	25163	10824	24365
validation/automatic	2744	12267	2512	1509	3844	1846	3831
test/automatic	307	2051	303	331	609	604	534
test/manual	91	320	76	50	80	62	92

Table 2: Count of pages, sentences which contain at least one of the target entities and number of mentions per each class of entities for different splits of the WikiPII dataset.

Predicted Entity	strict	exact	type	partial	implication in PII extraction
name Adam London.	✓✓	✓✓	✓✓	✓✓	no need for correction.
Adam *name* London.	✗	✗	✓	✓	boundary should be corrected.
place Adam London.	✗	✓✓	✗	✓	type should be corrected.
Adam *place* London.	✗	✗	✗	✓	entity is located, boundary and type should be corrected.

Table 3: Examples of predicted entity with respect to various evaluation metrics. ✗indicates no reward, ✓indicates half point reward and ✓✓indicates a full reward.

system where the human annotator relies on the location of predictions and corrects both label and boundary if necessary.

Table 3 shows examples of predicted entities by a PII recognizer with respect to the evaluation metrics and the cost of correction in a human-in-the-loop PII recognition task.

4 PII Extraction model

First, we evaluate the automated annotation compared to the manual annotation. Then we use the automatically annotated train set to train a BERT-based NER model with a fully connected linear layer as the prediction layer. We then evaluate the performance of the trained PII recognizer on both automatically and manually annotated test sets.

4.1 Comparison between Manual and Automatic Annotation

To evaluate the automatic annotation, we take manual annotations as the gold standard and score the corresponding automated annotations with the metrics described in Section 3. Table 4 shows the results of this evaluation. As discussed in Section 3, we used different metrics to evaluate this model based on the real-application scenario. For example, partial metric evaluates the scenario where the model is used to assist the human annotator in locating the entities. We observe that the rule-based annotation tool leads to high levels of noise. With partial evaluation, we conclude that automatic annotation spots about half of the entities correctly, but the boundary and type might not be fully correct. On the other side, the strict metric indicates that about one-third of the entities are perfectly annotated. Also, the type metric is higher than the exact metric, indicating that automatic annotation performs better in predicting types than boundaries. This is expected because of the complexities and subjectivity of boundary identification.

	strict	exact	type	partial
precision	0.31	0.32	0.39	0.46
recall	0.45	0.46	0.57	0.65
F1-score	0.37	0.38	0.47	0.54

Table 4: Evaluation of automated annotation compared to the manual annotation.

4.2 Performance of PII Extraction Model

We fine-tuned a BERT-based NER model with the training split of the automatically annotated WikiPII dataset to build a PII recognizer and tested the trained model with the test split of automatically annotated dataset and the manually annotated test set. We choose a batch size of 128 sentences and a maximum length of 50 tokens and present the results for one epoch of training. The optimum number of epochs varies between 1 and 3 for different datasets, but for the sake of comparison we choose to run all experiments with one epoch. Table 5 shows the results of this experiment.

We observed that despite the high level of noise in the automatically annotated training dataset the trained NER model reaches an acceptable performance. This is due to the large size of the automatically annotated dataset. As Rolnick et al. (2017) showed, deep learning models are robust to label noise when the size of the dataset is adequately large. We observed that the partial metric is

80%, which indicates a significant decrease in manual cost of PII extraction. While these predictions might still need corrections of type and boundary the system can locate most of the entities. From the strict metric, we conclude that half of the PII elements are predicted correctly in label and span and do not need any correction. Comparing of the exact and type metric shows that in most cases the system predicts the label correctly and boundaries need to be corrected.

5 Distributed Training

Modern deep learning models are known as data-hungry algorithms. In the task of PII extraction, sharing data across organizations will lead to more robust models. However, sharing of personal data in a central location involves concerns of privacy. To mitigate the risk of data breaches, we can train machine learning models in a distributed fashion while leaving the data in a location governed by the data owners. In this work, we explore Federated Learning (FL) (Yang et al., 2019) for training a NER model with noisy labelled data. Federated learning involves training statistical models over remote data centers, such as mobile phones or hospitals, while keeping data localized without requiring transfer of the whole dataset to a central location.

To implement FL, we use the PySyft framework, developed by *OpenMined* [5]. This framework is developed in PyTorch and provides the platform for executing tensor operations remotely (Ryffel et al., 2018). PySyft has been developed under the theme *"Answer questions you cannot see"*, to perform machine learning inference with zero knowledge about the specifics of the data.

In this framework, a central entity orchestrates the training scenario. Data is maintained and tagged by its owners at a remote location. At each data location, a worker follows the commands of the central entity. The model is transferred to the remote location, and updates are completed remotely at each training iteration. Subsequently, the final model is updated by averaging weights, averaging remote gradient updates or consecutive updates at each dataset location (Li et al., 2020).

5.1 Federated Training of BERT-based Model

BERT (Bidirectional Encoder Representations from Transformers) is a transformer-based language model pre-trained on a massive corpus of written text. BERT and a series of language models belonging to BERT's family form the backbone of today's deep learning NLP models. The language model generates a vector representation for the input text and passes it on to the downstream task. The pre-trained language model is usually fine-tuned with the task-specific data during task-specific learning. In this work, we only use BERT-based NER model, with a fully connected linear layer as the prediction layer, but the general idea applies to other transformer-based language models.Training and testing splits are the same as the ones used in Section 4.2.

The input to a BERT model contains three tensors: Token type id (specifies single sentence or double sentence use of the model), position ids (specifies the position of the token in the sentence), and input ids (specifies the id of the word in the vocabulary) (Devlin et al., 2018). Except for input ids, all the other inputs are tensors generated dynamically at the training time. The pre-trained tokenizer model generates these inputs to be based on the dimensions of the input sentence batches.

PySyft is designed in a way that abstracts the remote tensor objects by wrapping them around an empty tensor located centrally. Wrapping the tensor is the process of maintaining an empty local tensor object, while executing tensor operations on the remote tensor through the network. This method abstracts the location separation and allows the central worker to operate on tensor objects just as they were situated centrally. One drawback of this wrapping-based abstraction is that the functions such as size querying are operated on the local empty tensor rather than the native real tensor located in the remote worker. For that reason, the PySyft framework cannot query the dimensions of the input data tensors while operating in a remote worker (Ryffel et al., 2018).

For remote tokenization through PySyft, we modified the model to carry these inputs as static non-trainable parameters embedded in the form of tensor buffers. Using PySyft, we can move a model between the remote workers and the central worker using the API calls. Initially, these APIs were developed to handle the trainable parameters of the models among workers involved in federated learning. We contributed to the PySyft framework's codebase by developing a federated BERT tokenizer method, which handles the movement of

[5] http://www.openmined.org/

	Automatically annotated				Manually annotated			
	strict	exact	type	partial	strict	exact	type	partial
precision	0.64	0.72	0.70	0.74	0.55	0.56	0.68	0.79
recall	0.62	0.69	0.68	0.72	0.56	0.56	0.68	0.80
F1-score	0.64	0.70	0.69	0.73	0.55	0.56	0.68	0.80

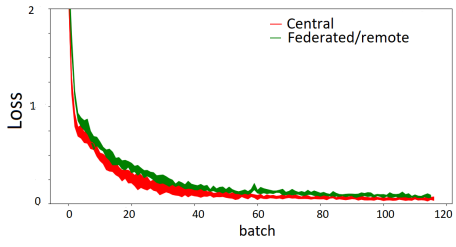
Table 5: Test accuracy of the BERT-based NER model on both test sets

non-trainable parameters and allows full remote functionality of the model. Our implementation of the *BERT-base* model for remote operation will be released for further research.

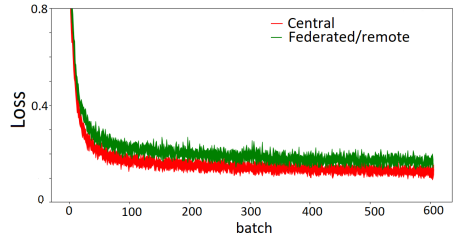
5.2 Training Scenarios

We deploy two settings of FL to share training data in a privacy-preserved manner. In practice one of these scenarios might be preferred depending on how much trusted the central worker is.

- federated/central: A trusted central operator can receive data batches from remote data holders
- federated/remote: A mistrusted central operator sends model to a remote data holders



(a)



(b)

Figure 4: Training loss for central vs federated/remote on a) CoNLL-2003 and b) WikiPII dataset

In scenario 1, the operator is trusted to receive data from remote sources and updates the model in the central location. Data batches from distributed sources are called using the federated training iterator. Received data batches contribute to forward pass and back-propagation operations. Then the operator discards the data batch as agreed. Here the operator has full control over the data batches, and the BERT tokenizer works in its typical mode. The only different operations with respect to central training are data transfers from the remote workers towards the central worker. These transfers might lead to information loss because of data compression. Also, from machine learning perspective, distributed data cannot be shuffled randomly and data batches might be imbalanced which has an impact on the final performance of the model.

In scenario 2, the operator is not fully trusted, so the batch of data can not be fully transferred to the central operator. Federated training iterator holds the locations of the data holders or remote workers. The model owned by the central operator is sent to the remote worker and allowed to be remotely executed. Only the central operator's commands are allowed to reach the remote worker guaranteeing the central operator is not breaching into the data. In this scenario, we use our remote tokenization method. Interactions for this training involve sending the model, sending commands to execute the model, and receiving the trained model parameters back. Model weights are received back by the central operator after training for all the batches of data belonging to the remote worker. Then the model is sent to the other workers. An epoch is completed when the model cycles all the workers.

5.3 Performance of Model Trained with Distributed Data and Noisy Labels

To gain insight into the impact of distributed training on NER models' performance, besides the WikiPII dataset, we trained our model on the widely used NER dataset, CoNLL-2003. Similar to our central training (Section 4.2), we restrict our experiments to one epoch of training. We simulated both scenarios with only two virtual workers and recorded the training loss to investigate how remote learning impacts the model's convergence. Also, for simplicity, we assume the remote workers' availability at all the times a central worker requests their computational resources for training, which might not be the real-world scenario and will require planning and robustness.

We observed that the case of federated/central

Dataset/Setting	no. of workers	F1 score
CoNLL2003		
central	N/A	0.90 ±0.005
federated/remote	2	0.85 ±0.003
federated/central	2	0.90 ±0.008
WikiPII		
central	N/A	0.70 ±0.006
federated/remote	2	0.56 ±0.02
federated/central	2	0.70 ±0.01

Table 6: Exact F1-score for central vs federated model

training does not impact the convergence of the model. Figures 4a and 4b show the convergence of the loss when model trained on two workers in federated/remote scenario compared to the typical centralized training, for both CoNLL2003 and WikiPII. In the case of CoNLL2003, where the annotations are of gold-standard quality, the federated training does not significantly impact the model's convergence. In WikiPII, with noisy labels, federated/remote training leads to higher loss function values. However, this impact is not detrimental.

The exact F-scores trained under our FL implementations are summarized in Table 6. We observed very close final model performance between the federated learning with centrally operated and typical centralized training. Federated training with the mistrusted central operator deviates from central training, with higher loss convergence values and a reduced final performance score, for both CoNLL2003 and WikiPII datasets. This observation can be explained by the loss of information in weight compression while transferring the model.

5.4 Effect of Dataset Size

Federated learning is most useful where multiple data holders participate in the training process. In reality, different distributed sources contributing to training can carry imbalanced amounts of data and features, which can have a negative impact on the results. Here we measure the effect of increasing the dataset size by increasing the number of workers. We randomly divided the training dataset among ten workers and, starting from 2 workers, increased the number of workers participating in the training process. Figure 5 shows the change of different types of F-score as more workers are utilized, and the dataset size increases as a result. We used the federated/central scenario here, which was shown to achieve comparable performance to central training. To control for the random sampling, we repeat each experiment 10 times and average the acquired F1-score. The error plot in Figure 5 demonstrates this experiment's final results for all the metrics.

In general, we observed that when the size of the noisy annotated data increases, higher performances are achieved. Since automatic labelling of data is inexpensive, generating and sharing noisy labelled data is a promising way of achieving high-quality models. However, note that the standard deviation of F-scores can be considerable. This observation indicates that the imbalances of distributed data can drastically impact the final model.

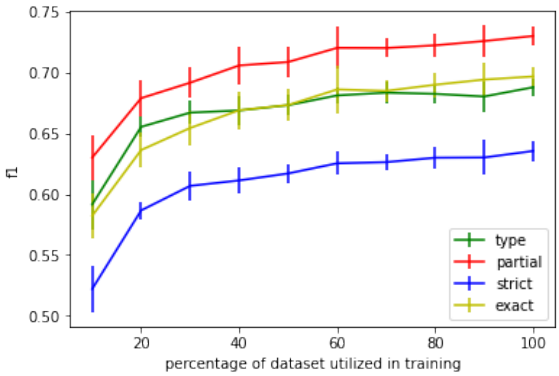


Figure 5: F1-scores vs. the training dataset size as more workers participate in federated/central training

6 Discussion

In this work, our goal is to use the historical data accumulated in an organization to build a customized NER for a human-in-the-loop PII recognition tool. We expect that this tool can significantly decrease the cost of manual extraction by locating PII entities in the free text.

First, we assume that the organization has access to a corpus of unstructured documents along with the structured dataset containing their corresponding PII entities. We propose that these parallel datasets can be used to create a noisy annotated training set. Our method of automatic annotation is based on matching of phrases and the raw data is not exposed to a third party for annotation. Using Wikipedia biography pages as an example, we show the feasibility of creating a noisy annotated dataset and training a PII recognition model in a privacy-preserved fashion. Our automatic annotation is inexpensive. Therefore it can generate large volumes of annotated datasets to compensate for the label noise. This is in line with previous work showing that deep learning is robust against noise

when trained with massive noisy datasets (Rolnick et al., 2017).

Furthermore, we looked at the feasibility of distributed training in cases that multiple organizations have similar datasets and are willing to collaborate to build more robust models but cannot share the data due to privacy concerns. We showed that where the operator is trusted, distributed training will not affect performance regardless of the annotation quality. For both CoNLL2003 (clean annotations) and our WikiPII dataset (noisy annotations), the F-score of NER models does not suffer from distributed training. However, when the operator is not trusted, the F-score is impacted and the drop of F-score is more significant in the case of the noisy dataset. In model transfer, all the parameter tensors of the model go through the simplification, serialization and compression steps followed by decompression, de-serialization and decompression steps. We suspect this mechanism affects the precision of the weights. In future work, a rigorous analysis should be carried out to analyze the effect of object transfers in a distributed system.

Lastly, in the federated/central scenario, we showed that the increase in the dataset size is a promising way to achieve higher accuracies. Distributed training allows organizations to share their data which results in a bigger size of the data. We conclude that there is a trade-off between the drop in performance because of the distributed training and the increase in performance because of the higher volume of data.

This work has limitations. NER is a very challenging task, and it is difficult to achieve a fully reliable NER model for a sensitive task such as PII extraction. Also, even highly accurate NER models can be vulnerable to adversarial attacks (Zhang et al., 2020). For this reason, throughout this work, we only envisioned this system to assist human annotators by locating the entities and suggest a highly likely tag. Although this system does not reach very high performance, it is still instrumental in reducing the cost of PII extraction when compared to a fully manual procedure. We only considered a BERT-based NER model, but the general idea applies to other transformer-based NER models. In future, an ensemble of different techniques should be considered to improve the utility of the system.

7 Conclusion

We propose an inexpensive and privacy-preserved method that automatically annotates parallel structured/unstructured datasets to train a customized NER models. The final models can be used to decrease the cost of manual extraction of PII elements by preprocessing the documents in a human-in-the-loop setting. Our results demonstrate that federated training is a promising tool to compensate for label noise by increasing the volume of the noisy labeled dataset.

Acknowledgement

We would like to thank IMRSV Data Labs for partial funding and support in manual annotation. We also acknowledge Mitacs for partially funding this project.

References

Tuomas Aura, Thomas A Kuhn, and Michael Roe. 2006. Scanning electronic documents for personally identifiable information. In *Proceedings of the 5th ACM workshop on Privacy in electronic society*, pages 41–50.

Yukun Chen, Thomas A Lasko, Qiaozhu Mei, Joshua C Denny, and Hua Xu. 2015. A study of active learning methods for named entity recognition in clinical text. *Journal of biomedical informatics*, 58:11–18.

Nancy Chinchor and Beth Sundheim. 1993. MUC-5 evaluation metrics. In *Fifth Message Understanding Conference (MUC-5): Proceedings of a Conference Held in Baltimore, Maryland, August 25-27, 1993*.

Jason PC Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional lstm-cnns. In *Transactions of the Association for Computational Linguistics*, volume 4, pages 357–370. MIT Press.

Zhenjin Dai, Xutao Wang, Pin Ni, Yuming Li, Gangmin Li, and Xuming Bai. 2019. Named entity recognition using BERT-BiLSTM-CRF for chinese electronic health records. In *2019 12th international congress on image and signal processing, biomedical engineering and informatics (cisp-bmei)*, pages 1–5. IEEE.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Kathleen C Fraser, Isar Nejadgholi, Berry De Bruijn, Muqun Li, Astha LaPlante, and Khaldoun Zine El Abidine. 2019. Extracting umls concepts from medical text using general and domain-specific deep learning models. *arXiv preprint arXiv:1910.01274*.

Jinlan Fu, Pengfei Liu, and Qi Zhang. 2020. Rethinking generalization of neural models: A named entity recognition case study. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7732–7739.

Filip Graliński, Krzysztof Jassem, Michał Marcińczuk, and Paweł Wawrzyniak. 2009. Named entity recognition in machine anonymization. *Recent Advances in Intelligent Information Systems*, pages 247–260.

Fadi Hassan, Josep Domingo-Ferrer, and Jordi Soria-Comas. 2018. Anonymization of unstructured data via named-entity recognition. In *International conference on modeling decisions for artificial intelligence*, pages 296–305. Springer.

Clete A Kushida, Deborah A Nichols, Rik Jadrnicek, Ric Miller, James K Walsh, and Kara Griffin. 2012. Strategies for de-identification and anonymization of electronic health record data for use in multicenter research studies. *Medical care*, 50(Suppl):S82.

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. 2020. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60.

Xiao-Bai Li and Jialun Qin. 2017. Anonymizing and sharing medical text records. *Information Systems Research*, 28(2):332–352.

Gang Luo, Xiaojing Huang, Chin-Yew Lin, and Zaiqing Nie. 2015. Joint named entity recognition and disambiguation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, page 879–888, USA. Association for Computational Linguistics.

Ahmadreza Mosallanezhad, Ghazaleh Beigi, and Huan Liu. 2019. Deep reinforcement learning-based text anonymization against private-attribute inference. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2360–2369. Association for Computational Linguistics.

Isar Nejadgholi, Kathleen C. Fraser, and Berry De Bruijn. 2020. Extensive error analysis and a learning-based evaluation of medical entity recognition systems to approximate user experience. In *The 19th SIGBioMed Workshop on Biomedical Language Processing (BioNLP2020)*, volume 19, pages 177–186.

Joel Nothman, Nicky Ringland, Will Radford, Tara Murphy, and James R Curran. 2013. Learning multilingual named entity recognition from wikipedia. In *Artificial Intelligence*, volume 194, pages 151–175. Elsevier.

Arttu Oksanen, J Tuominen, E Mäkelä, M Tamper, Aki Hietanen, and Eero Hyvönen. 2019. Semantic finlex: Transforming, publishing, and using finnish legislation and case law as linked open data on the web. *Knowledge of the Law in the Big Data Age*, 317:212–228.

Alexandre Passos, Vineet Kumar, and Andrew McCallum. 2014. Lexicon infused phrase embeddings for named entity resolution. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 78–86.

Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 11, page 269. NIH Public Access.

Lev Retinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL)*, page 147–155, USA. Association for Computational Linguistics.

David Rolnick, Andreas Veit, Serge Belongie, and Nir Shavit. 2017. Deep learning is robust to massive label noise. *arXiv preprint arXiv:1705.10694*.

Theo Ryffel, Andrew Trask, Morten Dahl, Bobby Wagner, Jason Mancuso, Daniel Rueckert, and Jonathan Passerat-Palmbach. 2018. A generic framework for privacy preserving deep learning. *arXiv preprint arXiv:1811.04017*.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the 7th Conference on Natural Language Learning, Edmonton, Canada*, pages 142—147.

Isabel Segura Bedmar, Paloma Martínez, and María Herrero Zazo. 2013. Semeval-2013 task 9: Extraction of drug-drug interactions from biomedical texts (ddiextraction 2013). Association for Computational Linguistics.

Satoshi Sekine and Hitoshi Isahara. 1999. IREX project overview. In *Proceedings of the IREX Workshop*, pages 7–12. Citeseer.

Richard Tzong-Han Tsai, Shih-Hung Wu, Wen-Chi Chou, Yu-Chun Lin, Ding He, Jieh Hsiang, Ting-Yi Sung, and Wen-Lian Hsu. 2006. Various criteria in the evaluation of biomedical named entity recognition. In *BMC bioinformatics*, volume 7, page 92. Springer.

Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19.

Wei Emma Zhang, Quan Z Sheng, Ahoud Alhazmi, and Chenliang Li. 2020. Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(3):1–41.