An Investigation into Named Entity Recognition for Call Center Transcripts to Ensure Privacy Law Compliance

Abstract

This study explores the application of Named Entity Recognition (NER) on a novel form of user-generated text, specifically call center conversations. These dialogues present unique challenges, blending the complexities of spontaneous speech with issues specific to conversational Automatic Speech Recognition (ASR), such as inaccuracies. By employing a custom corpus with manual annotations, training contextual string embeddings, and implementing a BiLSTM-CRF model, we achieve results that are on par with the state-of-the-art for this new task.

1 Introduction

This paper addresses the crucial need to identify and handle sensitive personal information within call center transcripts, which are generated as a result of speech recognition systems. Although these transcripts are typically redacted for Payment Card Industry (PCI) compliance, they still often contain a caller's name and internal ID number, which can be useful for quality assurance. However, new privacy laws, such as the General Data Protection Regulation (GDPR) in the EU, establish stringent guidelines concerning data collection, storage, and an individual's right to withdraw consent for data usage. To adhere to these regulations without losing the data's value, it is essential to pinpoint non-public personal and personally identifiable information (NPI/PII) in call transcripts.

We utilize Named Entity Recognition (NER) to locate instances of NPI/PII within the transcripts, remove them, and replace them with appropriate tags that denote the type of removed data. For instance, a transcript such as "This is john doe reference number 12345" would be transformed into "This is [NAME] reference number [NUMBER]". This task is distinctive to call centers for several reasons. First, these transcripts consist of natural human conversations, which have many common problems of user-generated content such as incomplete sentences and unusual words. Furthermore, transcript text is produced by Automatic Speech Recognition (ASR) systems, which are susceptible to errors, as will be described in Section 3.1. Even though modern ASR systems are usually reliable, the source audio is from phone calls, which is often low quality and contains background noise. The poor audio quality leads to incorrect ASR, producing ungrammatical sentences. This makes understanding the call semantics and identifying features essential to NER systems more difficult. Moreover, call transcripts frequently lack capitalization, numeric digits, and proper punctuation, which are crucial features for classic NER methods. Also, traditional NER systems are inadequate for handling emails, addresses, or spellings, which makes it difficult to use pre-trained NER models.

In this paper, we apply the current best neural network architecture for sequence labeling, a BiLSTM-CRF, to the task of identifying NPI and PII in call transcripts. We match the state-of-the-art performance on standard datasets by using our model with annotated data and custom contextual string embeddings.

2 Related Work

Named Entity Recognition has become a focus in the field of Natural Language Processing (NLP), particularly since the Message Understanding Conferences (MUCs) in the 1990s. The CoNLL2003 shared task in 2003 concentrated on language-independent NER and popularized feature based systems. The OntoNotes corpus, released in 2006, has been vital to the progress of NER research.

Following the CoNLL task, Conditional Random Field (CRF) based models became the most successful, which requires that features be manually produced. Current research utilizes neural networks to generate these features. Bidirectional Long Short Term Memory models with a CRF layer (BiLSTM-CRF) have been used successfully on CoNLL2000 and CoNLL2003 datasets. A BiLSTM-CNN-CRF has been used for NER on the CoNLL2003 dataset, producing superior results. Similar results were achieved by a BiLSTM-CNN with features from word embeddings and the lexicon. Embeddings have been used for both words and entity types to create more robust models. Flair, with character-based embeddings and a pooling approach, has set the state of the art. Crossweigh uses Flair embeddings to address mishandled annotations.

In 2006, the word confidence scores from ASR systems were used as a feature for NER. Similar experiments were done on French radio and TV audio. Neither of those used natural conversation, and the quality of the audio was superior, making ASR a more accurate task.

2.1 Conversations are Different: The Twitter Analogy

Much of the past research has used newswire datasets. While newswire data is expected to conform to standard text conventions, call center transcripts do not have these conventions. This presents a problem for the usual approaches to NER and is further complicated by our poor audio quality.

Speaker 1:	Thank you for calling our company how may i help you today.
Speaker 2:	Id like to pay my bill.

Table 1: An example of turns of a conversation, where each person's line in the dialogue represents their turn. This output matches the format of our data described in Section 3.

The most similar research area to this is work on Twitter data. Similar to our transcripts, tweets are user-generated and may not have conventional grammar or spelling. Initial research tackled this problem with a K-nearest neighbors model combined with a CRF. A model combining a multi-step neural network with a CRF output layer achieved first place in the 2017 Workshop on Noisy User-generated Text (W-NUT). The success of pooled contextualized string embeddings was also shown with this data. We use prior work on tweets to direct our model creation for call center data.

3 Data

Our dataset includes 7,953 training, 500 validation, and 534 test samples. Each sample represents a complete speaker turn from a debt collection call center. A speaker turn is defined as a complete transcription from one speaker before another speaker starts, as shown in Table 1. The training set is a random sample of turns from 4 months of call transcripts. The transcripts were generated using a proprietary speech recognition system, which outputs all lowercase transcripts without punctuation or numeric digits. We used spaCy to convert each turn to a document that begins with a capital letter and ends with a period, as this is the default for spaCy. In order to make use of entities, a Sentencizer module was added, which defaults to this capitalization and period structure.

3.1 Data Annotation

We created a schema for annotating the training and validation data with different types of NPI/PII, which are shown in Table 2.

Initial annotations were performed using Doccano. The annotators were trained in NPI/PII recognition, and were instructed to err on the side of caution in unclear instances. Ambiguity often came from errors in the ASR model. The lack of audio meant it was sometimes unclear if "I need oak leaves" was actually "Annie Oakley". The opposite was also true such as when "Brilliant and wendy jeff to

Entity Type	Description
NUMBERS	A sequence of numbers related to a customer's information (e.g. phone numbers or internal ID number)
NAME	First and last name of a customer or agent
COMPANY	The name of a company
ADDRESS	A complete address, including city, state, and zip code
EMAIL	Any email address
SPELLING	Language that clarifies the spelling of a word (e.g. "c as in cat")

Table 2: A brief description of our annotation schema.

process the refund" was actually "Brilliant and when did you want to process the refund". Emails were also difficult, as errors in ASR made it difficult to determine the bounds of the email address. Also, the transcripts were pre-redacted for PCI compliance. This redaction can obscure important data, for example, sometimes a customer ID is redacted as part of the PCI redaction process. To lessen false negatives, we use context to include the [redacted] tag as part of the numbers sequence when possible. No steps to clean the transcripts were taken; the natural noise in the data was left for the model to interpret.

Due to limitations with spaCy and the complexity of nested entities, we only allowed one annotation per word in the dataset. This means, for instance, that "c a t as in team at gmail dot com" would be labeled either as SPELLING[0:6] EMAIL[6:] or as EMAIL[0:] with the indices corresponding to the position of words in the text. This ultimately results in a lower count of SPELLING entities, because these are often part of EMAIL or ADDRESS entities, which influences our analysis in Section 6.

4 Model Design

We utilized a standard BiLSTM-CRF model in PyTorch, adapted from a GitHub repository. We wrote our own main.py to use our spaCy preprocessing, and adapted the code to handle batch processing. After preprocessing, we trained the model on the training set and used the validation set for model tuning. All numbers in this paper are reported on the test set. A visualization of our model is shown in Figure 1.

5 Experiments

5.1 Basic Hyperparameter Tuning

We used a grid search algorithm to maximize model performance. The word embedding layer uses FastText embeddings trained on the client's call transcripts. This aids in mitigating the impacts of poor ASR, and this will be explored in Sections 5.2 and 5.3. The grid search included the parameters: epochs (a sampled distribution between 5 and 50), the size of a dropout layer (between 0 and 0.5, with 0.1 intervals of search), the number of hidden layers (between 5 and 20 in increments of 5), and the encoding type used in the output of the CRF (BIO, BILOU, IO). The other hyperparameters were a learning rate of .001, a batch size of 1, 30 nodes in each fully connected layer, and the inclusion of bias in each layer. The experiments were run in parallel on a virtual machine with 16 CPUs and 128 GB of memory. Each experiment took a few hours to run.

To understand the performance of the model, we broke down the measurements of precision, recall, and F1 by entity type. Table 3 shows these results for the best model configuration. This model used 46 epochs, a dropout rate of 0.2, 5 hidden layers, and a BIO encoding.

5.2 Training Word Embeddings

Most past research has fine-tuned existing word embeddings, but the task of mitigating misrecognition seemed more complex than domain adaptation. To lessen the impact of the errors, we understand that frequent misrecognitions appear in contexts similar to the intended word. A custom model gives a misrecognized word a vector similar to the word it should be and not to the other meaning it has. The importance of domain specific word embeddings when using ASR data has been shown in research.

We ran our best performing model with the 300 dimensional GloVe 6b word embeddings. Our embeddings were trained on roughly 216 million words. The results from the best epoch of this model (16) are shown in Table 3.

2*Entity Type	Precision		Recall		F1	
	Custom	GloVe	Custom	GloVe	Custom	GloVe
O	89.8	84.2	81.7	76.6	85.6	80.2
NUMBERS	95.6	88.7	85.4	82.9	90.1	85.7
NAME	89.6	92.1	91.1	88.7	90.3	90.3
COMPANY	98.8	99.5	72.9	64.3	83.9	78.1
ADDRESS	70.6	0.3	75.0	18.7	72.7	23
EMAIL	0	07.1	0	03.1	0	04.4
SPELLING	45.8	34	52.4	40.5	48.9	37.0
Micro Average	89.2	85.6	79.6	74.0	84.1	79.4

Table 3: The performance by entity type of the BiLSTM-CRF model on the held out test set. This table compares the results of our custom embeddings model ("Custom") against the GloVe embeddings ("GloVe").

5.3 Using Flair

Previous experiments highlighted the importance of custom word embeddings to account for misrecognition in call center transcripts. Here, we test the performance of Flair and its contextual string embeddings.

We begin by training custom contextual string embeddings based on the results of the first experiments. We use the same corpus as in Section 5.1. The tutorial on the Flair GitHub page was used with the following parameters: hidden size: 1024, sequence length: 250, mini batch size: 100. We use the newline to indicate a document change, and each turn as a separate document for consistency. The model's validation loss stabilized after epoch 4, and the best version of the model was used.

We conduct experiments using Flair's SequenceTagger with default parameters and a hidden size of 256.

Flair uses only the custom trained Flair embeddings.

Flair + FastText uses the custom trained Flair embeddings and the custom trained FastText embeddings using Flair's StackedEmbeddings.

Flairmean pooling uses only the custom trained Flair embeddings within Flair's PooledFlairEmbedding. Mean pooling was used.

Flairmean pooling + FastText uses PooledFlairEmbeddings with mean pooling and the custom trained FastText embeddings using Flair's StackedEmbeddings.

These results are shown in Table 4.

Entity	Flair	Flair + FastText	Flairmean pooling	Flairmean pooling + FastText
O	98.3	98.5	98.2	98.5
NUMBERS	83.1	87.9	87.7	86.2
COMPANY	81.1	80.7	80.7	80.3
ADDRESS	87.5	94.1	61.5	94.1
EMAIL	58.8	50.0	73.3	66.7
SPELLING	55.0	57.1	55.8	57.9
Micro Average	97.5	97.7	97.3	97.7

Table 4: The F1 scores on the test set for each entity type for each Flair embedding experiment.

6 Discussion

Table 3 shows that using custom embeddings is beneficial over using GloVe embeddings, with the exception of the EMAIL category. The Flair embeddings show a large improvement over other word embeddings; however all four varieties of Flair models have nearly identical Micro Average F1s. The best performing Flair models are those that use both the custom contextualized string embeddings and the custom FastText embeddings.

Across all of the models in this paper, EMAIL and SPELLING consistently performed worse than other categories. This is due to the overlap in their occurrences and their variable appearance. The custom embeddings model often identified parts of an email correctly but labeled some aspects, such as a name, as NAME followed by EMAIL instead of labeling the whole thing as EMAIL. SPELLING often appears within an EMAIL entity. Due to the previously discussed limitations, the SPELLING entity had a limited presence in our training data, with many EMAIL and ADDRESS entities containing examples of SPELLING. All models frequently misidentified EMAIL as SPELLING and vice versa. Additionally, the test data had a number of turns that consisted of only SPELLING, which was poorly represented in training. The Flairmean pooling model outperforms the other models in EMAIL by a large margin.

The results in Table 4 highlight that the NUMBERS category contains strings that appear frequently in the text. There are a finite number of NUMBER words in our corpus (those numeric words along with many instances of "[redacted]"), and the numbers of interest in our dataset appear in very similar contexts and do not often get misrecognized. The COMPANY entity performs well for similar reasons; when the model was able to identify the company name correctly, it was often in a common error form and in a known context. The model's failures can be attributed to the training data because the company name is a proper noun that is not in standard ASR language models, including the one we used. Thus, it is often misrecognized since the language model has higher probabilities assigned to grammatically correct phrases that have nothing to do with the company name. This causes variability in appearance, which means that not every version of the company name was present in our training set.

Interesting variability also occurred in ADDRESS entities. Both models that used Flair and FastText embeddings strongly outperformed the models that used only Flair, and standard Flair embeddings strongly outperformed the Pooled Flair embeddings. Neither version of the Flair-only model identified addresses in which numbers were shown as "[redacted]" but both models that utilized FastText had no issue with these instances.

7 Conclusion and Future Work

Through the use of a BiLSTM-CRF model, paired with custom-trained Flair embeddings, we achieve state-of-the-art NER performance on a new call center conversation dataset with distinct entity types. We also show the importance of training word embeddings that fully capture the intricacies of the task. Although we cannot release our data for privacy, we have shown that existing state-of-the-art techniques can be applied to less common datasets and tasks. Future work will include evaluating the model with call transcripts from other industries. We would also like to explore how well these techniques work on other user-generated conversations like chats and emails.