May I Ask Who's Calling? Named Entity Recognition on Call Center Transcripts for Privacy Law Compliance

Micaela Kaplan

CallMiner Inc

Abstract

We investigate using Named Entity Recognition on a new type of user-generated text: a call center conversation. These conversations combine problems from spontaneous speech with problems novel to conversational Automated Speech Recognition, including incorrect recognition, alongside other common problems from noisy user-generated text. Using our own corpus with new annotations, training custom contextual string embeddings, and applying a BiLSTM-CRF, we match state-of-the-art results on our novel task.

Data

Speaker 1: Thank you for calling our company how may i help you today.

Speaker 2: Id like to pay my bill.

Figure 1:An example of turns of a conversation, where each person's line in the dialogue represents their turn.

The training set is a random sample of turns from 4 months of call transcripts from the client, but was manually curated to contain examples of NPI/PII to compensate for its relatively rarity in call center conversation.

Annotation Schema

We created a schema to annotate the training and validation data for a variety of different categories of NPI/PII as described below.

- **NUMBERS** A sequence of numbers relating 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. ``a as in apple")

Model Structure

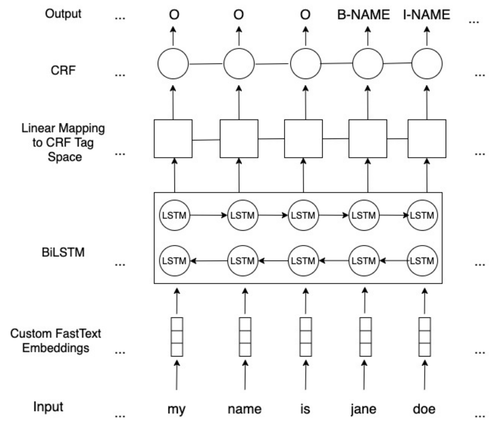

Figure 2:A schematic of our BiLSTM-CRF model. The text of each turn is passed to a word embedding layer which is followed by a BiLSTM layer, and then a linear layer that maps the word BiLSTM output into tag space. Finally, the CRF layer produces an output sequence.

Experiment 1: Hyperparameter Tuning

We used a grid search algorithm to maximize the performance of the model. The word embedding layer uses FastText embeddings trained on the client's call transcripts. We find that this helps mitigate the impacts of poor ASR in other aspects of our research. Grid Search Parameters

- **Epochs** Sampled distribution between 5 and 50
- **Dropout Layer Size** between 0 and .5, with .1 intervals of search
- **Number Hidden Layers** between 5 and 20 in increments of 5
- **CRF Encoding** BIO, BILOU, and IO

The other hyperparameters in our model were learning rate .001, batch size 1, 30 nodes in each fully connected layer, and the inclusion of bias in each layer. The experiments were run in parallel using Python's multiprocessing package on a virtual machine with 16 CPUs and 128GB of memory. Each experimental configuration ran on a time scale of a few hours, relative to the configurations of the hyperparameters being used.

Experiment 2: Custom Embeddings v. Pretrained Embeddings

While much of the previous research has fine-tuned existing word embeddings, the task of compensating for misrecognition seemed less straightforward than domain adaptation. We lessen the impact of the misrecognitions by understanding that frequent misrecognitions appear in contexts similar to the intended word. For example, ``why you're" is often misrecognized as ``choir" which would have a totally out of context vector from a pretrained model in this data set. A custom model gives ``choir" a vector that is more similar to ``why" than to ``chorus". **?** showed the importance of domain specific word embeddings when using ASR data.

We ran our best performing model configuration with the 300 dimensional GloVe 6b word embeddings. Our embeddings, in contrast, are trained on approximately 216 million words, making them substantially smaller than other state-of-the-art embeddings used today.

Results: Experiments 1 & 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	.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 1: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").

* Our custom model gets all 0s because many of its predicted EMAIL entities were off by a few words.

Experiment 3: Flair

We begin by training custom contextual string embeddings for this dataset, based on the findings in our original experiments. We conduct a number of experiments using Flair's SequenceTagger with default parameters and a hidden_size of 256. We adapt the work done by Akbik et al. (2018) and Akbik et al. (2019) to explore the impact of call center data on these state-of-the-art configurations.

We conduct the following experiments:

1. **Flair** uses only the custom trained Flair embeddings.
2. **Flair**$_{\text{+ FastText}}$ uses the custom trained Flair embeddings and our custom trained FastText embeddings using Flair's StackedEmbeddings.
3. **Flair**$_{\text{mean pooling}}$ uses only the custom trained Flair embeddings within Flair's PooledFlairEmbedding. We use mean pooling due to the results of Akbik et al. (2019) on the WNUT-17 shared task.
4. **Flair**$_{\text{mean pooling + FastText}}$ uses the PooledFlairEmbeddings with mean pooling and the custom trained FastText embeddings using Flair's StackedEmbeddings.

Flair Experiment Results

Entity	Flair	Flair$_{\text{+ FastText}}$	Flair$_{\text{mean pooling}}$	Flair$_{\text{mean pooling + FastText}}$
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 2:The F1 scores on the test set for each entity type for each Flair embedding experiment.

Key Discussion Points

1. Using custom trained word embeddings shows a significant improvement over using pretrained GloVe embeddings.
   - Flair always outperforms the custom FastText model in the Micro Average F1.
   - The Flair model variations were similar in performance in most cases, although each version had it's strengths and weaknesses
2. In all cases, EMAIL and SPELLING perform significantly worse than all other label types, followed by ADDRESS as the third worst performing category. Why?
   - EMAILS and ADDRESSES often contain Spellings, and Spellings don't often appear alone.
   - Our annotation schema only allowed one tag per word, so nesting a SPELLING within an EMAIL or ADDRESS was difficult.

References

Alan Akbik, Tanja Bergmann, and Roland Vollgraf. 2019. Pooled contextualized embeddings for named entity recognition. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, page 724–728.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. 2018. Contextual string embeddings for sequence labeling. In *COLING 2018, 27th International Conference on Computational Linguistics*, pages 1638--1649.