Homework Report

Named Entity Recognition using Conditional Random Field

Mohammad Rajiur Rahman

COSC6336 Spring 2020

Abstract

This report presents an implementation of a named entity recognizer to detect and classify named entities in noisy and unstructured text from twitter using Conditional Random Fields (CRFs). Three models were designed, and the best performing model achieved 33.22% f1-score (micro average) over the given dev data set are achieved.

1 Introduction

A named entity (NE) can be defined as a real-world object, such as persons, locations, organizations, products, etc., that can be denoted with a proper name. It can be abstract or have a physical existence. Named-entity recognition (NER) seeks to locate and classify NE mentioned in unstructured text into pre-defined categories such as person names, organizations, locations, etc.[1].

Twitter creates an enormous corpus of 140 characters long noisy informal message, by receiving more than 500 million tweets every day (Twitter Usage Statistics, 2020). This collective data is proved to be more informative and inclusive than news articles under many circumstances. However, tweets, like any other social media data, are different from traditional texts from books, newspapers, magazines etc. They are full of informal language, misspellings, abbreviations, hashtags, @-mentions, URLs, and unreliable capitalization and punctuation. Specially hashtags and "@" play a major role in tweets. People talk about everything including new entities never used by anyone before. Moreover, due to twitter's 140 character limit, tweets often lack enough context to determine an entity's class or label. These factors present a challenge for general-purpose NER systems to detect and classify NEs in noisy and unstructured text from twitter.

2 Related Work

There are significant amount of contributions in NER using CRF for social media data (Jain, 2018; Carreras, 2003; Alan Ritter, 2011; Tkachenko, 2019; Sutton, 2012). The present state of the art Flair (Akbik, 2018) reported 93.18% F1-measure on for NER on CoNLL-03 dataset, but their F1-measure drops to 49.49% F1-measure for WNUT-17 (tweeter) dataset. It outperforms previous best results of 92.22% F1-measure on CoNLL-03 (Peters, 2018) and 45.55% F1-measure on WNUT-17 (Guilar, 2017). These lower performance metrics of 'state of the art' systems on twitter data indicates the level of difficulty for NER on any social media data.

3 Dataset

For this assignment, training and development sets come from WNUT corpus, and test data is from an anonymous social media platform.

Set	Total Post	Total Token	Total Types
Train	2395	46469	10586
Dev	1001	16261	6255
Test	4271	42413	8634

Table 1: Total posts, token & types

The datasets are not balanced. Training dataset has 5.30% named entity tokens and dev dataset has 6.94% named entity tokens. Class distribution of NE tokens is shown in Figure 1. Dev dataset has more ratio of unique NEs if compared to the

[1] https://en.wikipedia.org/wiki/Named-entity_recognition

training set. Person, other and location classes comprise the major portion of the training dataset. Whereas the dev dataset has a similar proportion for five classes other than company and title classes.

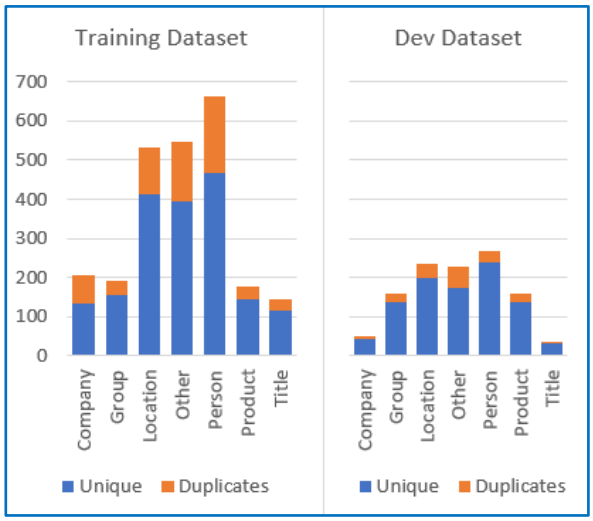

Figure 1: Number of NE in each class for training and dev dataset

4 Methodology

4.1 Preprocessing

Following preprocessing are carried out:

Checking missing values. Training and dev datasets are checked for missing values i.e. if there are any word without a valid NER tag. No such cases are found.

Removal of duplicate posts. Duplicate posts are removed because they would introduce more weights on same words appearing on the same sentence and the classifier would be biased towards correctly classifying that scenario over others.

Removal of posts with punctuations tagged as NE. It was postulated that punctuations tagged as NE may introduce noise in the system, allowing other punctuations treated as NEs. These posts are removed from the training dataset.

4.2 Features Selection

Features can be extracted from a token (word) being labelled and its surrounding context. A window ***W*** anchored in a word ω represents the local context of ω used by a classifier to decide on that word. Different features for the NER task are described in the class lecture (Solorio, 2020) and other related works. Different subsets of the following features are applied on a sequence of words in a sentence: ω_{i-2}, ω_{i-1}, m_i, ω_{i+1}, ω_{i+2} where the current word in consideration is i_n. Appendix A may be referred to for more detail.

- **Lexical:** Word forms, their lemmas and stems. Lemmas and stems are extracted using not[2].
- **Synthetic:** Parts of speech (POS) tags, extracted using nltk.
- **Orthographic:** Word properties with regard to how is it capitalized (initial-caps, all-caps), the kind of characters that form the word (contains-digits, all-digits, alphanumeric, roman-number), the presence of punctuation marks, membership of the word to a predefined pattern (URL).
- **Affixes**: The prefixes and suffixes of the word (up to 4 characters).
- **Capitalization**. A key feature for recognizing named entities is capitalization. However, capitalization in tweets is not a reliable feature for NE recognition. For example, non-entity words can be capitalized to emphasize. On the other hand, the entire tweet can be lowercased (including named entities). "Congratulations Tigers!", "tigers won the match" and "Save the Tiger from Extinction". First two tweets may refer to any sports team with nickname "Tiger". Whereas, the third tweet refers to the animal tiger. Sometimes all the words of a post are capitalized to express yelling (Willingham, 2018). To address this issue, it was checked if the entire post is capitalized.
- **Trigger Words**: Triggering properties of window words. An external list is used to

[2] https://www.nltk.org/

determine whether a word may trigger a certain Named Entity (NE) class (e.g., "Mr" or "Dr" may trigger person class). List of personal titles, collected from Wikipedia[3], is used.

- **Gazetteers**: A simple way to guess whether a phrase is a named entity or not is to look it up in gazetteers, which are lists containing names of entities like cities, persons, location, etc. Gazetteers for this task are generated from several lexicons.

4.3 Model

Sklearn-crfsuit library is used to create the following three models using different subsets of features described in section 4.2:

- Model-1 has all features except external sources like gazetteers and trigger-word list.
- Model-2 has all the features.
- Model-3 excludes lowercase form of the word (ω_i) from the features.

The library (sklearn-crfsuit) implements five training algorithms for CRF:

- Gradient descent using the L-BFGS method (lbfgs)
- Stochastic Gradient Descent with L2 regularization term (l2sgd)
- Averaged Perceptron (ap)
- Passive Aggressive (PA)
- Adaptive Regularization of Weight Vector (AROW)

All three model uses Gradient descent using the L-BFGS method (lbfgs) algorithm with 100 iterations and the following coefficients for L1 & L2 regularization (c1 & c2).

- Model-1 uses c1=0.36 and c2=0.08.
- Model-2&3 use c1=0.15 and c2=0.20.

The c1 and c2 values are selected after running the models over varying c1 and c2 values over the range of 0 to 10 and then select where the models perform best. All the models have 'all_possible_transitions' set to True and 'all_possible_states' set to False.

5 Results

Following are the performance metrics of the models (determined using seqeval[4] library):

	Precision	Recall	F1-score
Person	0.475	0.345	0.400
Location	0.560	0.362	0.440
Group	0.333	0.020	0.038
Title	0.000	0.000	0.000
Other	0.281	0.137	0.185
Company	0.636	0.179	0.280
Product	0.333	0.065	0.108
Micro Avg	**0.458**	**0.220**	**0.297**
Macro Avg	**0.422**	**0.220**	**0.273**

Table 2: Performance metrics of Model-1

	Precision	Recall	F1-score
Person	0.400	0.065	0.111
Location	0.613	0.412	0.493
Group	1.000	0.020	0.039
Title	0.302	0.122	0.174
Other	0.576	0.376	0.455
Company	0.636	0.179	0.280
Product	0.200	0.062	0.095
Micro Avg	**0.534**	**0.239**	**0.330**
Macro Avg	**0.582**	**0.239**	**0.302**

Table 3: Performance metrics of Model-2

	Precision	Recall	F1-score
Person	0.400	0.065	0.111
Location	0.605	0.418	0.495
Group	0.750	0.030	0.058
Title	0.315	0.130	0.184
Other	0.559	0.369	0.444
Company	0.636	0.179	0.280
Product	0.200	0.062	0.095
Micro Avg	**0.528**	**0.242**	**0.332**
Macro Avg	**0.538**	**0.242**	**0.305**

Table 4: Performance metrics of Model-3

[3] https://en.wikipedia.org/wiki/List_of_titles

[4] https://pypi.org/project/seqeval/

6 Discussion

The dev dataset has less than 7% named entities. Any classifier that returns 'O' tag for every token would get a little more than 93% accuracy. Hence F1-measures give us a better understanding of the models' performance.

The accuracy of the model-1 is little more than 93%, but the F1-measure is very poor compared to the provided baseline results. However, without external features like gazetteers and list of trigger words, model-1 performs very good for person and location class as there are more data from person and location classes in the training set compared to data for other classes (Figure 1).

Inclusion of gazetteers and trigger word features in model-2 increases the overall F-1 measure (Figure 2). It reduces the recognition of person class and increases the recognition of title class named entities.

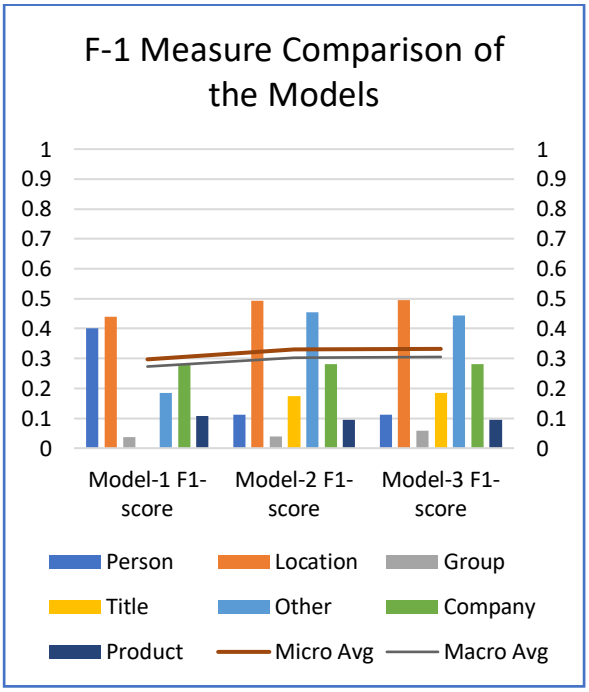

Figure 2: Performance Comparison of different Models

After adding gazetteer and trigger words feature, it is found that the model has learned a few company names like Facebook, Twitter, YouTube, etc. (Figure 3) from the lowercase feature of the word. The lowercase feature is removed in the model-3 to make it more generalized.

Removing the feature lowercase of the word itself does not change in recognition F-1 measure for 'company' labels. Moreover, there are no significant changes in other classes as well. Rather model-3 has slightly higher f1-measures.

Features learned from location label (Figure 4) shows that in the window of size 2 (two) "B-location" labels have "in", "where" and "at" prior to the word under consideration. If features with negative weights are checked, it is found that if any token is followed by "with" it's unlikely to be a "B-location".

Weight[?]	Feature
+0.307	word.lower():twitter
+0.196	word.lower():facebook
+0.108	word.lower():youtube
+0.107	word.lower():fb
+0.099	word.lower():ione
+0.092	word.lower():msn
+0.090	word.lower():yahoo
+0.090	word.lower():tumblr
+0.083	word.lower():walmart
+0.080	word.lower():ufc
... 83 more positive ...	
-0.034	-1:word.islower()
-0.037	+1:word.islower()

Figure 3: Top features for 'B-company' label the model has learned

Weight[?]	Feature
+1.181	-1:word.lower():in
+1.176	-2:word.lower():in
+0.861	+2:word.lower():where
+0.736	+1:word.lower():york
+0.720	-1:word.lower():at
... 383 more positive ...	
... 31 more negative ...	
-0.198	-1:word.lower():with
-0.207	+2:word.lower():,
-0.212	+2:word.lower():(
-0.226	+2:word.lower():after
-0.279	-2:word.lower():f

Figure 4: Top features for 'B-location' label the model has learned

Few more similar examples over other classes can be analyzed as well (Figure 5):

- "4.201385 O word.ispunctuation" shows that if the word is punctuation it must be labelled as 'O'.

- "3.258766 O BOS, 3.046755 O EOS" indicates that the beginning and ending words of a post are mostly labelled 'O'. This might have happened because I didn't remove '#' and '@' to read the word. Many tweets begin with a word beginning with @ and ends with a word with hashtags ('#' symbol).

- "-0.933649 B-location shape (word): Xxx" shows that the model has learned that any place doesn't have a three-character word shape (Xxx).

Top positive:	
4.201385 O	word.ispunctuation
3.258766 O	BOS
3.046755 O	EOS
2.584555 O	isHashTagOrUserName
2.270356 B-company	company.gazetteer
2.161867 O	shape(word):xx
Top negative:	
-0.933433 O	-2:word.lower():happy
-0.933649 B-location	shape(word):Xxx
-0.950819 B-product	-1:word.istitle()
-0.966924 O	word[-2:]:ex
-0.974401 O	word[-2:]:ro
-0.975828 O	word[-2:]:za
-0.977214 B-person	-1:word.istitle()

Figure 5: Top positive and top-negative features

These weights are learned by the model from the training set for the selected features.

7 Future Work

This section describes some limitations of features selection and experiments in this work. Probable ways to improve are also highlighted.

7.1 Additional Features

Following features can be explored to improve the performance of the task:

- Lemmas & stems of other words in the window.

- Word position in the window (e.g., W(1)="street") and in the sentence.

- Chunk tags.

- Single character patterns (lonely initial, punctuation-mark, single-char), or the membership of the word to a predefined class.

- Type pattern of consecutive words in the context. The type of a word is either functional (f), capitalized (C), lowercased (l), punctuation mark (.), quote (') or other (x). For instance, the word type pattern for the phrase "John Smith paid 3 euros" would be CClxl.

- Gazetteer information for other words in the window.

- Emojis are not handled. The training and dev dataset had emojis converted to the text description. However, the test data had emojis as Unicode characters. It is postulated that; the model has never seen emoji characters and any word with one or multiple unseen characters (single emoji or multiple consecutive emojis) would be tagged as 'O'.

- Number generalization, for example, considering masks of numbers as 10-12-1996 → *DD*-*DD*-*DDDD*, 1999 → *DDDD*

- Preprocessed text with a truecaser, which attempts to automatically figure out what the correct capitalization of the text should be.

- Brown clusters (Turian, 2010)

- Word2Vec clusters (Sienčnik, 2015)

7.2 Additional Experiments

Experiments with other algorithms inside the sklearn-crfsuit library can be carried out to check performance.

Hyperparameter tuning using randomized search and multi-fold cross-validation can be used to tune the parameters for algorithms used.

7.3 Limitations of Gazetteers

Adding gazetteers didn't improve the performance to the expected level. The implementation should ensure that it has features that can counter gazetteer feature to let the model balance itself.

Another reason for not having much improvement could be that the list (gazetteer) is closed and it is not context-sensitive. For example, “Sunday” is a surname (label: I-person) in one of the gazetteers whereas “Sunday” is also a day of the week (label: B-other).

7.4 Observation on Learning of Classifiers

Detail observation can be carried out on what the model is learning to select a better set of features for the given task. This work highlights only one such feature like ‘lowercase’ word form. There is scope to look into other features as well.

8 Conclusion

This report presents studies and experiments on different feature extractions techniques and implementing them in models based on Conditional Random Fields (CRFs) to perform NER task on twitter data. Ranking of the best model of this work is 11 with a weighted F1 score of 24.24% on the test dataset. The report also analyzes the limitation of the work and includes possible improvements as future work.

9 References

Akbik, A. a. (2018). Contextual String Embeddings for Sequence Labeling. *International Conference on Computational Linguistics*, (pp. 1638--1649).

Alan Ritter, S. C. (2011). Named Entity Recognition in Tweets: An Experimental Study. *EMNLP 2011 - Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference* (pp. 1524-1534). Ritter, Alan & Clark, Sam & Mausam, Mausam & Etzioni, Oren.

Carreras, X. &. (2003). A Simple Named Entity Extractor using AdaBoost. *Seventh Conference on Natural Language Learning at HLT-NAACL*, (pp. 152–155).

Guilar, G. &.-M. (2017). A Multi-task Approach for Named Entity Recognition in Social Media Data. *3rd Workshop on Noisy User-generated Text*, (pp. 148-153).

Jain, D. &. (2018). Simple Features for Strong Performance on Named Entity Recognition in Code-Switched Twitter Data. *hird Workshop on Computational Approaches to Linguistic Code-Switching*, (pp. 103-109).

Peters, M. &. (2018). Deep contextualized word representations. . *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Sienčnik, S. K. (2015). Adapting word2vec to Named Entity Recognition. *Proceedings of the 20th Nordic Conference of Computational Linguistics, NODALIDA 2015* (pp. 239-243). Linköping University Electronic Press, Linköpings universitet.

sklearn-crfsuit. (2020). *sklearn-crfsuit*. Retrieved from readthedocs.io: https://sklearn-crfsuite.readthedocs.io/en/latest/index.html

Solorio, T. (2020, Feb 3). Class Lecture on Conditional Random Fields for Named Entity Recognition. COSC 6336 (Fall 2020).

Sutton, C. a. (2012). An Introduction to Conditional Random Fields. *Found. Trends Mach. Learn.*, 267–373.

Tkachenko, M. a. (2019). Named entity recognition: Exploring features. *Conference on Natural Language Processing (KONVENS)*, (pp. 118-127).

Turian, J. a.-A. (2010). Word Representations: A Simple and General Method for Semi-Supervised Learning. *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics* (pp. 384-394). Uppsala, Sweden: Association for Computational Linguistics.

Twitter Usage Statistics. (2020, Feb 2). Retrieved from Internet Live Stats: https://www.internetlivestats.com/twitter-statistics/

Willingham, A. (2018, July 23). *CNN*. Retrieved from Why typing in all-caps looks like you're yelling (A brief history): https://www.cnn.com/2018/07/23/us/all-caps-typography-history-tweets-trnd/index.html

A Appendices

Specific Features Used

Given a sequence of words in a sentence: w_{i-2}, w_{i-1}, w_i, w_{i+1}, w_{i+2} and the current word in consideration is w_i, I used the following features:

- The lower-case version of w_i
- Prefixes and Suffixes of length 2 of w_i
- Prefixes and Suffixes of length 3 of w_i
- Prefixes and Suffixes of length 4 of w_i
- Length of the word w_i
- If w_i is at the end of the sentence
- If w_i is at the beginning of the sentence
- If w_i is all capitalized
- If w_i is a title (first character capital letter)
- If w_i contains digits only
- If w_i has any digits
- If w_i is alphabetic
- If w_i has '#' or '@' as the first character
- If w_i belongs to trigger word list
- The shape of the word w_i
- If w_i is URL
- If w_i is the Roman numeral
- Lemma of w_i
- The stem of w_i (Portar algorithm from nltk[5])
- The stem of w_i (Lancaster algorithm from nltk)
- If w_k belongs to person gazetteer
- If w_k belongs to company gazetteer
- If w_k belongs to group gazetteer
- If w_k belongs to location gazetteer
- If w_k belongs to product gazetteer
- If w_k belongs to 'other' gazetteer

I looked following features considering the entire post

- If the post has all words in upper case
- Use NTLK to get parts of speech (POS) tags for all the words

For each of the words w_k (i=k-1 and i=k+1) in the context window of 1, I used the following features:

- Prefixes and Suffixes of length 2 of w_i
- Prefixes and Suffixes of length 3 of w_i
- Prefixes and Suffixes of length 4 of w_i
- Length of the word w_i
- The shape of the word w_i
- If w_i is URL

For each of the words w_k (i=k-2, i=k-1, i=k+1 and i=k+2) in the context window of 2, I used the following features:

- The lower-case version of w_k
- If w_k is all capitalized
- If w_k is a title (first character capital letter)
- If w_k contains digits only
- If w_k is alphabetic
- POS tag of w_k
- POS Prefix of length 2 of w_k

For the previous two words w_k (w_{i-1} and w_{i-2}), I used the following features:

- If w_k belongs to trigger word list

[5] https://www.nltk.org/