IITP: Multiobjective Differential Evolution based Twitter Named Entity Recognition

Md Shad Akhtar, Utpal Kumar Sikdar and Asif Ekbal
Dept of Computer Science and Engineering
IIT Patna
Patna, India
(shad.pcs15,utpal.sikdar,asif)@iitp.ac.in

Abstract

In this paper we propose a differential evolution (DE) based named entity recognition (NER) system in twitter data. In the first step, we develop various NER systems using different combinations of the features. We implemented these features without using any domain-specific features and/or resources. As a base classifier we use Conditional Random Field (CRF). In the second step, we propose a DE based feature selection approach to determine the most relevant set of features and its context information. The optimized feature set applied to the training set yields the precision, recall and F-measure values of 60.68%, 29.65% and 39.84%, respectively for the fine-grained named entity (NE) types. When we consider only the coarse-grained NE types, it shows the precision, recall and F-measure values of 63.43%, 51.44% and 56.81%, respectively.

1 Introduction

During the last few years there has been a phenomenal growth in the number of users that make use of different social networking platforms to share their opinions and views. Twitter now has upto over 500 million users with approx 302 million active users [1]. One can easily imagine that amount of tweets generated per day would be enormous i.e. almost 500 million tweets per day [2]. These information are usually unstructured and noisy in nature. The reason behind its unstructured nature is that tweets are rather short messages (constitute upto 140 characters only), contains several grammatical & spelling mistakes etc. The size limitation bounds a user to invent several short forms (*e.g. 2mrw, tmrw for tomorrow*) of a valid word which a human mind can interpret easily but, on the other hand, becomes very difficult to come up with an accurate system for solving any problem related to natural language processing (NLP). Also in order to show their emotions, users sometime put extra emphasis by elongating a valid word (*e.g. yeeesssss!! for yes*).

Named entity recognition (NER) can be seen as one of the important and foremost tasks for many natural language processing (NLP) tasks such as machine translation, information extraction, question-answering etc. The task of NER can be thought of as a two-step process that involves identifying proper names from the text and classifying them into some predefined categories such as person, organization, location etc. Although the techniques (Bikel et al., 1999; Ekbal and Bandyopadhyay, 2008a; Ekbal and Bandyopadhyay, 2008b; Sikdar et al., 2012) for recognizing named entities (NEs) in newswire and other well-formatted traditional corpus has already matured but it is still a challenging task to perform in unstructured and noisy twitter data.

The concept of NER in twitter has recently drawn the attention of researchers worldwide. Very few authors have reported their works (Liu et al., 2011; Ramage et al., 2009; Li et al., 2012) for NER in twitter. A semi-supervised model for NER has been reported in (Liu et al., 2011) where K-nearest neighbour classifier is combined with CRF. Application of LabeledLDA (Ramage et al., 2009) in supervised environment can be found in (Ritter et al., 2011). Their method classifies NEs into fine-grained types of 10 classes (as in our case). In another work (Li et al., 2012), authors have used random walk model to build an unsupervised approach to NER. They modelled their system on local(*tweets*) and global (*www*) context without employing any of the linguistic features.

[1] http://en.wikipedia.org/wiki/Twitter

[2] http://www.cnet.com/news/report-twitter-hits-half-a-billion-tweets-a-day/

Few more related works can be found in (Derczynski et al., 2015) and (Locke and Martin, 2009).

Due to several challenges it pose, recently there has been a huge interest to identify NE in twitter data. In compliance with this a shared task "ACL2015 W-NUT: Named Entity Recognition in Twitter"[3] was organized. The work that we report here is a part of this shared task. The main objective of the shared task was to efficiently identify various coarse-grained and fine-grained named entities. Fine-grained NE types include 10 different categories namely, person, product, company, geo-loc, movie, musicartist, tvshow, facility, sportsteam and other. We have used a rich feature set based on lexical and syntactic properties of a tweet as discussed in Section 3.9. Our proposed work uses Conditional Random Field (CRF) (Lafferty et al., 2001) as learning algorithm, which is very efficient as a sequence learner. Subsequently we have applied Differential Evolution (DE), a stochastic, population based optimization algorithm, introduced by Storn and Prince in 1996 (Storn and Price, 1997), to obtain the optimal feature set for NER in twitter data.

The organization of the paper is as follows. Section 2 provides a very brief theoretical discussion of DE. Feature set and methodology used in the proposed work are discussed in Section 3. Experimental result and analysis can be found in Section 4. We conclude the paper in Section 5.

2 MultiObjective Differential Evolution (DE)

Differential Evolution (DE) (Storn and Price, 1997) is a heuristic search optimization technique and it provides near optimal solution for an optimization problem. Within a search space the parameters are encoded in the form of string, which is called chromosome/vector. A chromosome is, therefore, nothing but of D number of real values. A collection of such types of chromosomes is called population. A fitness value is associated with each chromosome. For single objective optimization the fitness value depends upon the these D number of real parameters. For multiobjective optimization, more than one fitness value is associated with each chromosome. The fitness value denotes the goodness of the chromosome. DE generates new vector by adding the weighted difference between two vectors to the third vector. This operation is called the mutation. In the next step, the mutant vector parameters are mixed with the parameters of the predefined vector. The new vector is termed as the trial vector, and the parameter mixing process is called crossover. The best vectors are selected from the trial vectors. The process of selecting new vectors from the current population is known as selection. The algorithm that we follow for this is known as the crowding distance sorting algorithm. The processes of mutation, crossover and selection continue for a fixed number of generation.

[3]http://noisy-text.github.io/

3 Methods

The proposed system is consisting of two steps. In the very first step we generate many models based on the best fitting feature sets. Following this heuristic based approach we select the best model by fine-tuning on the development data. In the second step we develop a multiobjective DE based feature selection approach to find out the best feature combinations and its contextual information from the selected feature set. Schematic diagram of the proposed system is depicted in figure 1.

3.1 Problem Formulation

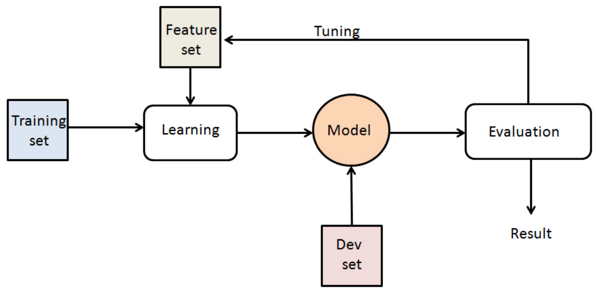
Suppose there are D features available, and these are denoted by $F_1, \ldots, F_D$, where $\mathcal{A} = \{F_i : i = 1; D\}$ Determine the subset of features $\mathcal{A}^{'} \subseteq \mathcal{A}$ such that we learn a classifier with these subset of features and optimize some metrics. In our proposed multiobjective DE, we optimize two functions, namely precision and recall.

3.2 Problem Representation and Population Initialization

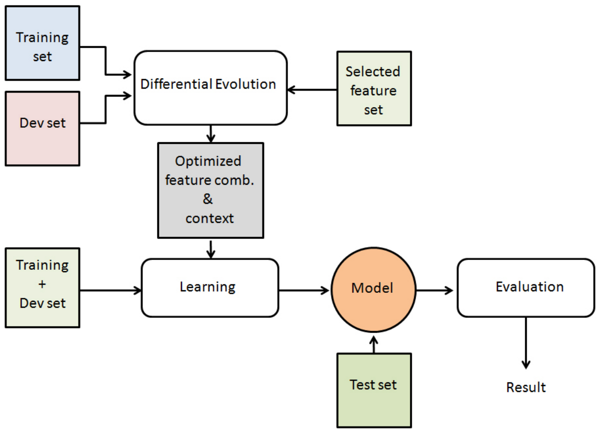
All the chromosomes are initialized with the binary values of either 0 or 1, where 1 denotes that the corresponding feature is present and 0 denotes that the corresponding feature is off. Total number of available features denote the length of the chromosome, and we set this as D. A classifier learns with the available set of features. One example of chromosome representation is shown in Figure 2.

3.3 Fitness Computation

The fitness computation corresponds to determining the values precision and recall as two objective functions. If M number of features are present in the chromosome, a classifier is trained with these



(a)



(b)

Figure 1: Proposed methodology (a) Step 1 (b) Step 2.

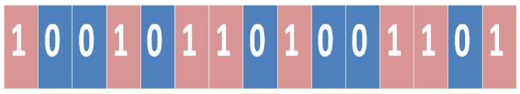


Figure 2: Chromosome representation: Here #available features = 15 and #features present = 8

M number of features. The classifier is then evaluated on the development data. We calculate precision and recall as the two objective functions. The goal is to maximize these two functions.

3.4 Mutation

In mutation process, a mutant vector $V_{i,G+1}$ is generated for each target vector $X_{i,G}$; $i = 1, 2, 3, \ldots, NP$, according to

$$V_{i,G+1} = x_{r1,G} + F \times (x_{r2,G} - x_{r3,G}), \quad (1)$$

where $r1$, $r2$, $r3$ are generated randomly with different indices, not equals to current index i and belong to $\{1, 2, \ldots, NP\}$, G is the generation number and F is the mutant factor which is set to 0.5. If the parameters of mutant vector $v_{i,j,G+1} > 1$, then the parameter values are set to 1. If the parameters of mutant vector $v_{i,j,G+1} < 0$, then the parameter values are set to 0.

3.5 Crossover

To generate better solutions (represented by the chromosomes) to the next generation population, crossover is needed. The parameter mixing of the target vector $X_{i,G}$ and mutant vector $V_{i,G+1}$ is called crossover. Crossover generates a trial vector as follows:

$$U_{i,G+1} = (u_{1,i,G+1}, u_{2,i,G+1}, \ldots, u_{D,i,G+1}) \quad (2)$$

where

$$u_{j,i,G+1} = v_{j,i,G+1} \text{ if } (r_j \leq CR) \text{ or } j = i_r \quad (3)$$
$$= x_{j,i,G} \text{ if } (r_j > CR) \text{ and } j \neq i_r \quad (4)$$

for $j = 1, 2, \ldots, D$, where r_j is an uniform random number of the jth evaluation which belongs to $[0, 1]$ and CR is crossover constant which is set to 0.5. The index value i_r belongs to $\{1, 2, \ldots, D\}$ that ensures that at least one parameter of trial vectors $U_{i,G+1}$ gets one parameter from the mutant vector $V_{i,G+1}$.

3.6 Selection

In selection process, trial vectors are merged with the current population to get the best NP solutions from the merged solutions $2 \times NP$ in the next generation population. The merged solutions are sorted based on dominated and non-dominated concept and generate ranked solutions. As an example, the dominated and non-dominated sorting are shown in Figure 3. The non-dominated solutions are represented in the pareto-optimal surface. The non-dominated solutions are added to the next generation population until the number of solutions becomes equal to NP. If the number of solutions in a particular rank exceeds NP, then it is sorted based on crowding distance algorithm. The required number of solutions are added from the beginning of the sorted rank to make NP number solutions in the next generation population. The selection process determines the best NP number of solutions in the next generation population.

3.7 Termination Condition

Mutation, fitness computation, crossover and selection processes run for a maximum number of generations. At the end, we get a set of non-dominated solutions.

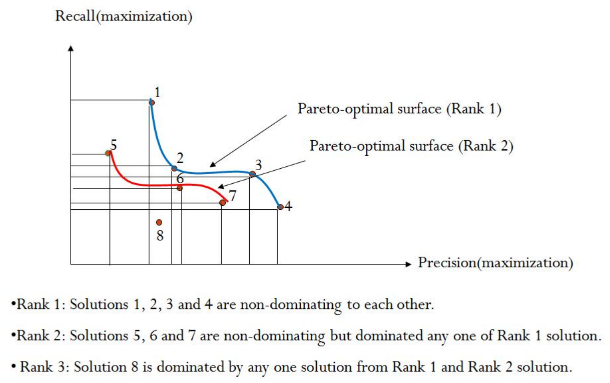


Figure 3: Representation of dominated and non-dominated solutions

3.8 Selecting the best solution

The multiobjective optimization (MOO) based algorithm yields a set of solutions on the Pareto optimal front at the end. None of these solutions is better compared to the others. However, we may often require to find out a solution at the end. Depending upon the user's requirements different criteria for selecting the best solutions can exist. Each feature vector of the final Pareto optimal front generates a classifier. We compute the F-measure value on the development set for each classifier. We select the solution which reports highest F-measure value. The features encoded in this chromosome is used to train a CRF and report the final evaluation on the test data.

3.9 Feature Set

In this section we describe the features that we implement for performing NER. The features are domain-independent and we implement these without using any external resources and/or tools.

1. **Local context:** We use local contextual information as the features of CRF. We use previous few and succeeding few words as the features for learning.

2. **Part-of-Speech information:** PoS information is one of the prominent features in identifying the NE. We have used CMU-ARK Twitter NLP tool[4] for extracting the PoS information. We use the PoS information of preceding and succeeding few tokens as the features.

3. **Word length:** From the given training data we observed that NEs generally become longer in lengths. We define a feature that is set to high if the length of the candidate token exceeds a predetermined threshold. In our case we assume the token to be a NE if its length exceeds 5 characters.

4. **Suffix and Prefix:** Suffixes and prefixes of length upto 4 characters of the current word are used as the features.

5. **Word normalization:** We normalize the current token and use it as a feature. For normalization we map the capitalized letter to 'A', small letter to 'a' and numbers or symbols to 'x'.

6. **Previous word:** We prepare a list of most frequent words that appear before a NE in the training data. A binary valued feature is then defined that fires if the current word appears in this list.

7. **Stop word:** This checks whether the current word appears in the list of stop words or not. We obtain the list of stop words available at [5].

8. **Uppercase:** This feature checks whether the current word starts with a capital letter or contains a upper case letter inside the word or all the characters of the word are capitalized.

9. **All digit:** This feature checks whether the current token is consisting of only digits.

10. **AlphaDigit:** Tokens having combination of alphabet and digit have less probability of being a NE. This concept is used to define a binary feature in the proposed work which fires when the token is alphanumeric.

11. **First & last word:** Tweet level information are employed for defining two features i.e. if the current token is the first or last word of a particular tweet.

12. **Word frequency:** We observe that most frequently occurring words have a tendency of not being NE. We prepare a list of most frequent words from the training data. A binary

[4]http://www.ark.cs.cmu.edu/TweetNLP/

[5]http://ir.dcs.gla.ac.uk/resources/linguistic_utils/stop_words

valued feature is then defined that checks whether the current word appears in this list or not.

13. **Gazetteer:** We prepare a list of NEs from the training and development datasets. Along with the NE we also store the NE types. We define an integer-valued feature that takes the value that corresponds to the respective NE type.

4 Datasets and Experiments

In this section we firstly describe the datasets and then report the evaluation results.

4.1 Data Set

As discussed earlier, objective of the shared task was to identify both the coarse-grained and fine-grained NE from the tweets. Shared task organizers provided two separate versions of training ($train_{notype}$ and $train_{10type}$) datasets and four versions of development datasets (dev_{notype}, $dev2015_{notype}$, dev_{10type} and $dev2015_{10type}$). The training dataset comprise of 1,795 tweets while development datasets comprises of 599 & 420 tweets for dev and $dev2015$, respectively. A total of 1,768 NEs are present in the dataset, out of which 1,140 are present in the training set and rest 628 are present in the development set. Brief statistics of the datasets are shown in Table 1 and Table 2 for the coarse-grained NE tagged and fine-grained NE tagged datasets, respectively. Gold standard test datasets comprise of 1,000 tweets.

Dataset	# Tweets	# Token	# NE
$train$	1795	34899	1140
dev	599	11570	356
$dev2015$	420	6789	272
$test2015$	1000	16261	-

Table 1: Statistics of the coarse-grained dataset

4.2 Experimental Results

As a base learning algorithm we make use of Conditional Random Field (CRF)(Lafferty et al., 2001). We use the CRF++ [6] based package for our experiments. Evaluation of all the systems are performed in compliance with CoNLL 2002 evaluation script[7] as recommended in the shared task. For comparative analysis a baseline system was also provided by the organizers for both fine-grained and coarse-grained versions. We started our experiments by training the model on the features defined in Section 3.9. Iteratively we have trained, tested and evaluated the system in order to find out the best fitting feature sets. Afterwards we shifted our focus to DE for optimizing the feature set in terms of relevant features and its context information. DE was initialized with the population size equal to 100, and it was executed for 50 generations. We have carried out these experiments for both fine-grained and coarse-grained datasets. On termination, multiobjective differential evolution (MODE) reported optimized feature combinations for both the types of datasets. At the final step these optimized feature combinations were used to build the final system. We show the optimized feature sets as determined by MODE in Table 3.

Types	***train***	***dev***	***dev*2015**
person	332	117	73
product	79	18	9
company	130	41	33
geo-loc	218	58	46
movie	31	3	3
musicartist	43	12	13
tvshow	26	8	6
facility	84	20	7
sportsteam	33	18	35
other	164	61	47

Table 2: Statistics of the fine-grained dataset.

Results of various models along with the baseline are reported in Table 4. The upper half of the table contains the experimental results for three systems. These three models correspond to the official baseline model, model developed with all the features and the model developed with the selected features of DE. The MODE based feature selection model yields the F-measure value of 56.81% for the $test2015$ dataset. It is evident that it performs well above the official baseline that showed the F-measure value of 49.88%. Similarly for the fine-grained NE types (lower half of the table) our system (39.84% F-measure) is convincingly ahead of the baseline model (31.97% F-measure) for the official test data ($test2015$).

[6]http://taku910.github.io/crfpp/

[7]http://www.cnts.ua.ac.be/conll2002/ner/bin/conlleval.txt

Types	Dataset	Model	Precision	Recall	F-measure	Accuracy
notype	*dev*	Baseline	65.25	55.90	60.21	96.95
		All features	65.08	57.58	61.10	96.92
		MODE	69.81	62.36	65.88	97.12
	*dev*2015	Baseline	55.79	49.82	52.63	95.08
		All features	51.49	50.92	51.21	94.31
		MODE	60.97	53.51	57.43	95.60
	*test*2015	Baseline	53.86	46.44	49.88	95.01
		All features	52.37	56.32	54.27	95.55
		MODE	63.43	51.44	56.81	95.50
10type	*dev*	Baseline	57.04	44.38	49.92	96.44
		All features	61.23	39.04	47.68	96.29
		MODE	70.71	39.33	50.54	96.43
	*dev*2015	Baseline	38.53	30.88	34.29	94.14
		All features	37.14	23.90	29.08	93.50
		MODE	48.33	24.26	32.35	94.33
	*test*2015	Baseline	35.56	29.05	31.97	93.41
		All features	42.41	30.00	35.14	94.94
		MODE	60.68	29.65	39.84	94.54

Table 4: Results of various systems on different dataset. All values are in %.

Features	*C-grained*	*F-grained*
POS	√	√
WordLength	√	√
Suffix	√	√
Prefix	√	√
WordNorm	√	√
PrevOccur		√
Stop word		
InitCap	√	
AllCap		
InnerCap		√
AllDigit		
AlphaDigit	√	√
First & last word		
WordFreq		
Gazetteer		√

Table 3: Optimized feature sets.

5 Conclusion

In this paper we have presented our works that we carried out as part of our participation in the Twitter NER shared task. We have used a set of features which were implemented without using much domain specific resources and/or tools. We have considered various combinations of features and finally select the combination that yields the best result. We further apply MODE based feature selection on this feature set. Official evaluation shows F-measure of 39.84% for the fine-grained NE types and 56.81% F-measure for the coarse-grained NE type.

In future we would like to carry out more comprehensive analysis on the evaluation results. The features that we used here are very general in nature. In future we would like to investigate domain-specific features to improve the accuracy of the system.

References

Daniel M. Bikel, Richard Schwartz, and Ralph M. Weischedel. 1999. An algorithm that learns what's in a name. *Mach. Learn.*, 34(1-3):211–231, February.

Leon Derczynski, Diana Maynard, Giuseppe Rizzo, Marieke van Erp, Genevieve Gorrell, Raphal Troncy, Johann Petrak, and Kalina Bontcheva. 2015. Analysis of named entity recognition and linking for tweets. *Information Processing & Management*, 51(2):32–49.

Asif Ekbal and Sivaji Bandyopadhyay. 2008a. Bengali named entity recognition using support vector machine. In *Third International Joint Conference on Natural Language Processing, IJCNLP 2008, Hyderabad, India, January 7-12, 2008*, pages 51–58.

Asif Ekbal and Sivaji Bandyopadhyay. 2008b. Named entity recognition in indian languages using maximum entropy approach. *Int. J. Comput. Proc. Oriental Lang.*, 21(3):205–237.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *ICML*, pages 282–289.

Chenliang Li, Jianshu Weng, Qi He, Yuxia Yao, Anwitaman Datta, Aixin Sun, and Bu-Sung Lee. 2012. Twiner: Named entity recognition in targeted twitter stream. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, pages 721–730, New York, NY, USA. ACM.

Xiaohua Liu, Shaodian Zhang, Furu Wei, and Ming Zhou. 2011. Recognizing named entities in tweets. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 359–367, Stroudsburg, PA, USA. Association for Computational Linguistics.

B. Locke and J. Martin. 2009. Named entity recognition: Adapting to microblogging. *University of Colorado*.

Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D. Manning. 2009. Labeled lda: A supervised topic model for credit attribution in multilabeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 248–256, Stroudsburg, PA, USA. Association for Computational Linguistics.

Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: An experimental study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1524–1534, Stroudsburg, PA, USA. Association for Computational Linguistics.

Utpal Kumar Sikdar, Asif Ekbal, and Sriparna Saha. 2012. Differential evolution based feature selection and classifier ensemble for named entity recognition. In *COLING 2012, 24th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, 8-15 December 2012, Mumbai, India*, pages 2475–2490.

Rainer Storn and Kenneth Price. 1997. Differential evolution a simple and efficient heuristic for global optimization over continuous spaces. *J. of Global Optimization*, 11(4):341–359, December.