Iran Journal of Computer Science
https://doi.org/10.1007/s42044-018-0014-5

ORIGINAL ARTICLE

Sentence-based undersampling for named entity recognition using genetic algorithm

Abbas Akkasi[1]

Received: 7 November 2017 / Accepted: 25 February 2018

Abstract

Named entity recognition (NER), as one of the crucial tasks of information extraction (IE), has important effect on the quality of its subsequent applications such as answering the question, co-reference resolution, relation discovery, etc. NER can be considered as a kind of classification problem, which has to deal with its own challenging issues. Class-Imbalanced Problem (CIP) is one of the important problems in classification domain from which almost all NER tasks also suffer, because usually, the number of entity mentions of interest in the given text is much less than undesired entities. The quality of the IE's subtasks for which NER is the basis is directly affected by any improvement on the performances of NER systems. In this research, an effort has been made to increase the overall performance of NER systems by decreasing the curse of CIP as much as possible. A new heuristic approach based on the genetic algorithm has been devised to undersample the training data which is used for NER. Regarding the fact that given training patterns for NER are of individual sentence forms, in the developed approach, this issue is considered as well and it was applied to individual sentences from training data. The proposed method has been applied on two different corpuses: CoNLL corpus from newswire domain and JNLPBA from biomedical context to see its impact on different type of contexts. By increasing the performance in terms of F-score for both data sets, our proposed method outperforms the baseline systems using original data. Furthermore, in comparison with random undersampling, it results in better outcomes. In addition, the effect of considering sentences of training data individually in sampling process and taking all of them together has been investigated.

Keywords Named entity recognition · Class-Imbalanced Problem · Genetic algorithm · Undersampling

1 Introduction

Nowadays, useful information in all aspects of life is presented in the form of unstructured texts. This vast amount of hidden information can be used for different purposes such as indexing documents, detecting the topics of given texts, extraction of needed information and question answering system as well [1]. Text mining is one of the possible solutions to achieve such information [2]. Named entity recognition as a task of assigning meaningful tags from well-defined ontologies to the certain named entities is the basic step for aforementioned tasks. Depending on the context of texts, these entities of interests can be varied, for example, person names, location names, or dates can be of importance in general news wire domain and protein, genes, or chemical compounds can be thought as relevant in biomedical background [3]. Suitability of machine learning strategies for variety of problems caused widespread usages of its techniques. Classification as an important type of machine learning approach can be mapped to the NER problem aiming at assigning a proper label from predefined class set to the named entity mentions [4], so in this case, NER is going to be considered as a classification task, and it would not be free from the challenging issues related to this domain. Class imbalance problem (CIP) is one of the crucial subjects among 10 challenging problems in data mining researches [5] which needs the highly attentions to deal with. CIP refers to the cases that samples of a specific class type are much more than samples of other classes [6]. In this case, classifiers usually tend to bias towards identifying examples from the classes which majority of samples belong to. Since in NER, usually, the number of entity mentions of interest is much lower than other entities which do not carry useful

✉ Abbas Akkasi
Abbas.akkasi@fer.hr

[1] Faculty of Electrical Engineering and Computing, University of Zagreb, Zagreb, Croatia

Published online: 06 March 2018

information (from outside class), so CIP would be innate property of this kind of problems. Named entity recognition can be considered as sequence labeling task in classification domain; its important difference regardless of used classification algorithm with ordinary classification tasks is related to the structural format of input data. Each data sample is in the sentence form (i.e., each example can be imagined as a sequence of string words which goal is to map class labels to them). CIP in the typical classification tasks has got more attentions, while the number of researchers that specially paid attention to the CIP in NER is very few.

In this research, we proposed a novel heuristic approach using the genetic algorithm to decrease curse of CIP in NER problems and subsequently to improve the overall performance of entity identifier systems. In the proposed method, we tried to consider each individual input sentence for under sampling process, instead of taking all of them together into account. Applying heuristic method for data sampling process along with our focus on the structure of data samples in training data for NER can be considered as contributions of this research.

The rest of this paper is organized as follows, a very brief review on important concepts used in this paper is represented in Sect. 2 individually. Third part of paper is dedicated to the explanation of the proposed methods. In Sect. 4, the experimental setup that we used to our experiments during this research and achieved results are discussed. Finally, we ended this paper with conclusion part in Sect. 5.

2 Literature review

Here, in this section, some of important concepts mostly used in this paper are explained in detail.

2.1 Named entity recognition

Named entity recognition or entity identification is an important subtask of information extraction (IE), which tries to find named entities of interests in given text and assign a label form predefined set of class labels to them [3]. NER includes two subtasks, first locating entity mentions in a text and classifying those mentions into the classes which are the representatives of them [4]. Figure 1 shows an example of NER by labeling a text segment with given labels.

Named entity recognition is a basic step for other information extraction tasks such as question answering, relation discovery, name resolution, etc. [4]. To deal with NER problem in different contexts, four strategies can be used [3]: (I) dictionary-based approaches, which try to find existing entities in pre-created repositories of named entities within the given text; (II) rule base methods, work based on devising regular expressions to find mentions of interest; (III) machine learning approaches, which treat the NER problem as a kind of supervised or sometimes semi supervised classification task. Different classification algorithms such as conditional random fields (CRFs) [7], support vector machine (SVM) [7], are usually applicable here; and (VI) the last strategy is known as hybrid approach which focuses on combining the aforementioned methods together to take advantage of each individual method.

In case of machine learning approach to solve NER problem, each data sample can be seen as a sequence of tokens construct a sentence. The aim is to map the proper label from the class set to each token. In the other words, training data are of $\langle \vec{X}_i, \vec{Y}_i \rangle$ form, where $i = 1$ to N, and $\vec{X}_i$ shows the sequence i and $\vec{Y}_i$ indicates the corresponding labels of each token in the given sentence i. NER tries to find the most appropriate sequence of labels for given unlabeled input sequences [6].

2.2 Class-imbalanced problem (CIP)

Classification under an imbalanced data distribution is one of the current challenging issues in data mining. CIP appears when the number of samples of a specific class is much lower than samples from other classes, and identifying those rare samples is of crucial importance rather than detecting others. Fraud detection, risk management, fault detection in software engineering, text classification, etc. can be seen as the kinds of applications that CIP is their innate property [8]. In practice, CIP is shown by binary class problem, so multiclass cases usually translate into two class problems [8,9]. Since rare instances are of greater interests, usually, they are referred as positive samples and samples from the unimportant class are referred as negative examples. The important factor in dealing with CIP is the imbalance ratio (IR) which is also called as imbalanced degree or imbalance rate [10,11].

It shows the ratio of number of the negative samples (N_{Neg}) to the number of positive ones (N_{Pos}), as shown by Formula (1), regrading only for training set the labels of samples are known, IR calculates based on training data distribution:

$$\text{IR} = \frac{N_{\text{Neg}}}{N_{\text{Pos}}}. \tag{1}$$

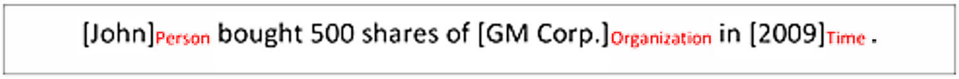

[John]Person bought 500 shares of [GM Corp.]Organization in [2009]Time .

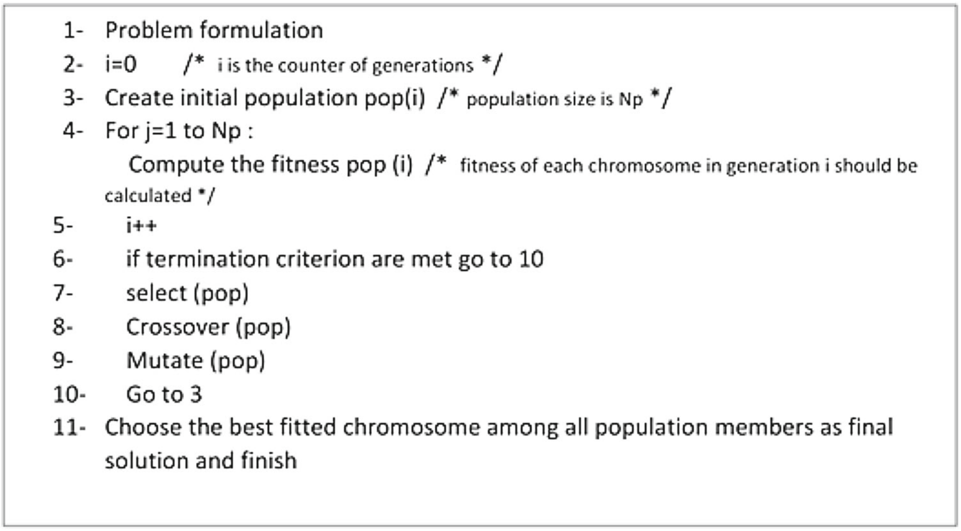
Fig. 1 Example of NER

There is no standard threshold value for different applications such that for IRs greater than it, one can say there is class-imbalanced problem. Japkowicz and Stephen [10] have shown that CIP is a kind of relative problem which depends on some factors in addition to IR: (i) complexity of underlying problem represented by the data; (ii) the total size of training data set; and finally (iv) used classifier for classification process. Dealing with CIP can be categorized into three groups: data-level solutions, algorithmic level, and ensemble learning [12]. Data-level approaches refer to the preprocessing tasks on data, known as resampling methods [13]. Resampling can be applied on data in two different ways: undersampling, which tries to make balance between negative samples and positives by removing some negatives. Another resampling strategy is oversampling which works in contrast to under sampling, i.e., in this case, balancing process will carry out by duplicating positive samples. Both approaches have their own drawbacks; for example, by applying under sampling, it would be possible to get rid of some useful information for pattern recognition process which lied on negative samples, and in the case of oversampling, the possibility of over fitting is high [14]. Different strategies for each kind of resampling methods are developed by the researchers as follow: random under sampling (RUS) [13], Tomek link [15], condensed nearest neighbor rule (CNN) [16], neighborhood cleaning rule (NCL) [17] and one side selection (OSS) [18] for under sampling and synthetic minority oversampling technique (SMOTE) [19], border line SMOTE [20], and clustering based oversampling [21] are most commonly used oversampling techniques which tries to convert the data distribution in new desired IR (optimum ratio). The solutions in algorithmic level involve with adjusting learning algorithm used for classification or tuning parameters for any special problem [13], so the main weakness of these approaches is being "problem oriented"; cost-sensitive learning [22] is an examples for such approaches. Ensemble-based approaches are another kind of strategies to deal with CIP, and the main idea behind these approaches is to leveraging the classification power of various weak classifiers together to improve the generalization performance using unclassified data samples [23]. Ensemble methods can be used in different modes such as Bagging [24], or Boosting algorithms, especially Ada Boost [25], which the main focus of these two approaches is to create different base line classifiers using randomly drawn data sets with replacement from original training data. The only difference between bagging and boosting approaches is the importance of created classifiers in final decision-making process. All created classifiers contribute equally in final aggregation phase applying bagging, while in boosting-based approaches, for each base line classifier, a special weight of impact would be defined by algorithm during classifiers creation phase. All mentioned strategies are applicable on ordinary classification tasks which samples are independent from each other. In addition, they will be applicable to NER domain if NER is considered as a common categorization task and if each sequence of characters (tokens) from sentences of given text consider as an input sample regardless of its dependencies on other tokens. Tomanek and Hahn [26] dealt with CIP in NER with changes that they applied on their active learning approach. Gliozzo et al. [27] used a kind of under sampling named instance filtering, where for each token, they calculated a weight, then eliminated the samples with lower weight's values. Ekrem et al. [6] recently proposed a new sampling method to use in NER systems, and the main idea behind their approach is to keep the number of negative samples around each positive sample balanced as much as possible.

2.3 Genetic algorithm

Genetic algorithm (GA) is a kind of metaheuristic search algorithm inspired by natural evolution and mimics the process of natural selection. GA is also one of the most successful algorithms for both search and optimization problems [28] which involve solving different types of complicated problems by means of simulating and applying the natural evolution concepts such as selection, inheritance, crossover, and mutation. The underlying idea behind GA is not very complicated, and it is iterative population based approach where population includes a number of candidate solutions which usually known as chromosome or solution vector [29], to find the best optimum solution the evolution process applies to the current population by means of genetic operators. Figure 2 depicts the algorithm in its simplest representation.

Formulating given problem into GA domain is the first step such as the length of chromosome, value type of each part member of chromosome (gene), population size, termination criteria, etc. Then, for the initial generation, population should be initialized (depends on the problem statement it can be done randomly or manually or combination of both) and fitness of each chromosome in population must be calculated using defined fitness function for the problem at hand. To get new generation of population, some of the current population members should be selected by different selection approaches such as tournament selection or roulette wheel [30] methods and then genetic operators, crossover, and mutation will be applied to the selected solutions to create new generation members. For each generation, all candidate solutions should be investigated, and if a solution could meet the stopping criteria, it should be selected as optimum solution. Each genetic operator can be implemented in several ways; for example, crossover can be of types such as: single point crossover, double point crossover, uniform crossover, and three parents' crossover, or for mutation operator, it is



```
1-  Problem formulation
2-  i=0      /* i is the counter of generations */
3-  Create initial population pop(i)  /* population size is Np */
4-  For j=1 to Np :
      Compute the fitness pop (i)  /* fitness of each chromosome in generation i should be
    calculated */
5-     i++
6-     if termination criterion are met go to 10
7-     select (pop)
8-     Crossover (pop)
9-     Mutate (pop)
10-    Go to 3
11- Choose the best fitted chromosome among all population members as final
    solution and finish
```

Fig. 2 Genetic algorithm

possible to apply substitution of its implementations such as bit string mutation, or flip bit [30]. All parameters should be adjusted to their appropriate values depending on the given problem.

3 Proposed approach

With respect to the effect of CIP in the performance of NER systems in different domains, a new heuristic undersampling method has been proposed here in this paper to decrease the number of the negative samples aiming at increasing the performance of entity identification process. We applied genetic algorithm as the underlying architecture for approach proposed. This method consists of four steps: (1) calculating original imbalanced ratio (IR) for training data; (2) adjusting genetic algorithm's parameters and definition of its constraints; (3) applying genetic algorithm to the training data for new desired sampling ratios starting from 1 to original IR by step 1, to find optimum sampling ratio; and (4) creating classifiers using under sampled training data and evaluating the unlabeled test data. Figure 3 shows the overall workflow of this research.

The first step above all is to determine the actual imbalance ratio for given training data. To calculate the value of IR, training data should be tokenized in appropriate mode. The most commonly used class representation scheme in NER domain is the IOB format [31], where "B" stands for beginning part of an entity, "I" mean that the token is one of the inner parts of an entity, and "O" shows that the entity is not important (from Out class). All tokens from class "O" are considered as negative samples, while other tokens taken as positives. In our proposed method, instead of taking all tokens from all sentences together and take the ratio of number of all negatives to all positives, the overall IR will be calculated according to following procedure: for all sentences, IR calculated individually (IR$_{S}$), and then, a small integer number is chosen as the increment step-s-after that for each (IR$_{S}$), the number of sentences which their imbalance ratio is in the range [(IR$_{S}$), (IR$_{S}$) + s) is counted, and then, we choose the (IR$_{S}$) + s from the range that (IR$_{S}$) of the majority of sentences belong to that range. In case that the same number of sentences belongs to different ranges, the highest upper bound of those ranges would be selected. For example, assume an imaginary case in which we have four sentences that their (IR$_{S}$) are as follows, respectively: 2, 4, 5, and 6. we took 2 as the increment value. Now, the number of sentences that their (IR$_{S}$) fall into the following ranges should be calculated: [2, 2 + 2), [4, 4 + 2), [5, 5 + 2), [6, 6 + 2). The corresponding numbers of sentences for those ranges are 1, 2, 2, and 2, respectively; now, the chosen original IR for whole data set should be 7. The main reason for taking imbalance ratio of sentences individually into account is considering the distribution of negative and positive samples within the sentences instead of sentences all together. In the next step, decisions about GA parameters such as population size, chromosome representation, fitness function, selection method, crossover, and mutation strategies, are made.

The used chromosome here is a kind of binary array with length of N, where N is the summation of number of tokens of all contributed sentences in training data. Every solution vector consists 0, 1 which shows the class statues of each token of sentences. It should be noticed that each chromosome here can be seen as concatenation of different binary arrays with different lengths per each sentence. Figure 4 shows the example of chromosome representation for two sentences which separated by different colors. For each token of "O" class, we put 0 in corresponding place in the chromosome and for tokens belonging to "B-CLASS", or "I-CLASS", 1 is put. "ClASS" can be everything based on the problem domain.

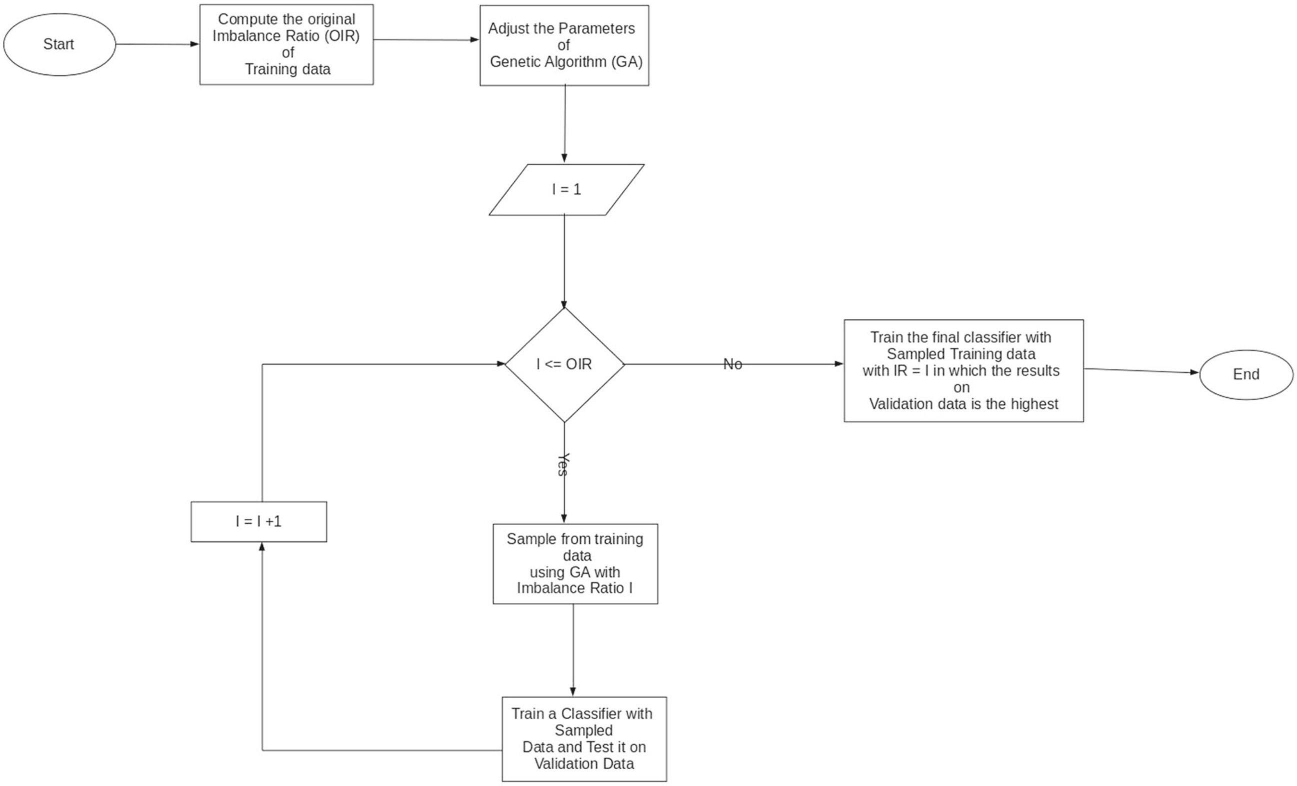
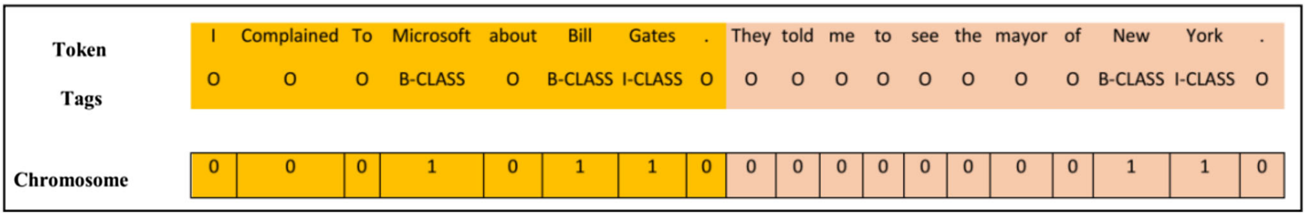

Start
Compute the original Imbalance Ratio (OIR) of Training data
Adjust the Parameters of Genetic Algorithm (GA)
I = 1
I <= OIR
No
Train the final classifier with Sampled Training data with IR = I in which the results on Validation data is the highest
End
Yes
Sample from training data using GA with Imbalance Ratio I
Train a Classifier with Sampled Data and Test it on Validation Data
I = I +1

Fig. 3 Research workflow



Token	I	Complained	To	Microsoft	about	Bill	Gates	.	They	told	me	to	see	the	mayor	of	New	York	.
Tags	O	O	O	B-CLASS	O	B-CLASS	I-CLASS	O	O	O	O	O	O	O	O	O	B-CLASS	I-CLASS	O
Chromosome	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	1	1	0

Fig. 4 Example for chromosome representation

Table 1 Statistics of corpuses used for the experiments

Corpus	# of used articles	# of sentences	# of NE's	Original IR
CoNLL				
Train	946	14,041	23,499	35
Development	216	3259	5942	–
Test	231	3453	5648	–
JNLPBA				
Train	2000	18,546	51,301	25
Development	–	–	–	–
Test	404	3856	8662	–

Individual sentences of training data and corresponding sub-array of a chromosome are separated by different colors in given example in Fig. 4. The final goal of the proposed method is to reach distribution of tokens within all sentences to be balanced in optimum sampling ratio, such that the trained classifier with such train data has maximum generalization performance on development data. Measurement used for performance evaluation (fitness value) is F-score [32] which is the harmonic metric based on combination of recall and precision. Since we do not want to lose any posi-

Table 2 Feature sets used for the experiments

Feature set	Actual features in the feature set
Space features	Has right space, has left space, has both right and left space
Context words	One token before and one token after current token
n-Gram affixes	*n*-Gram affixes (prefixes + suffixes) for n = 1:4 for each token
Word shapes	Word shape (number of uppercase, lowercase letters, digits, punctuations, greeks), digital word shape (word shape in digital format), Summarized word shape (combination of two aforementioned features)
Orthographic features	All upper case, has slash, has punctuation, has real number, start with digit, starts with upper case, has more than 2 uppercase letters
Token length	Number of characters in the token

Table 3 Values for GA's parameters

Parameter	Range	Step size	Selected value	
			CoNLL	JNLPBA
Population size	Pop size for both corpuses considered as 300			
Iteration	[50, 200]	25	100	100
Crossover rate	[0.5, 1]	0.1	0.6	0.8
Mutation rate	[0.0, 0.6]	0.1	0.4	0.2
Selection method	Both tournament and roulette wheel, by random			

tive sample, so in the initialization phase, all positive samples should be considered as 1 in all chromosomes and keep them unchanged until the end of process; only values of genes (bits of chromosomes) in which their initial values were 0 could be changed during the evolution process. The novelty of our method is applying under sampling on each sentence instead of considering all sentences together. Lack of considering sentences individually in this process can cause the unfair samples distributions through the training data, where some sentences would be remained unchanged, but in our method, it is guaranteed that all sentences would participate in sampling process and all of them will be under sampled based on their own original distribution of samples. During the implementation phase, we put this issue as a constraint to new solution generation process, such that all start/end positions of each sequence of tokens belong to each sentence are memorized at first, and then in sampling process to achieve desired IR, all individual sentences have taken into account instead of considering all tokens of all together. Both Tournament and Roulette Wheel selection methods are taken into account here, by choosing one of them randomly. Thresholds of other parameters have been selected by examining the results on development set of each corpus for different values in a specific range for each parameter, while other parameters have been kept unchanged. In the case of lack of individual development data set, *n*-fold cross validation can be used instead, or also it would be good idea if 30% of train data thrown randomly as development set. In the recent case, after finding the proper values for algorithm parameters and optimum desired sampling ratio (The ratio leads to the highest classification performance on development data), once again method should be applied on whole training data set.

After finding the best sampled training data using the proposed method, trained classifier with sampled data is applied on the test data to see effects of method proposed on the generalization performance of classifiers trained using sampled data.

4 Experiments

To investigate the efficiency of the proposed method on different domains for NER application, the experiments are done on two corpuses from different contexts. CoNLL [33] corpus for English texts, it is a kind of general topic data sets which is related to the news wire domain; interested entities here include name of persons, organizations, dates, etc. The second corpus that we used is JNLPBA [34] from biological domain which mentions of interest are from biomedical entities such as genes, proteins, DNA, and etc. Statistics about used data sets are given in Table 1.

As shown in Table 1, JNLPBA corpus does not have development data set; therefore, 30% of training data is randomly selected as development data set during experiments. Original imbalance ratio per each training set of each corpus is given in the last column. Classifiers' performances considered as fitness function to evaluate the quality of each solution; since conditional random fields (CRFs) [7] are the

Table 4 Effects of different under sampling methods on JNLPBA test data

DIR	S1			S2			S3			S4		
	R	*P*	*F*	*R*	*P*	*F*	*R*	*P*	*F*	*R*	*P*	*F*
3	75.71	59.37	66.55	79.19	49.78	61.13	75.01	65.46	69.9	78.56	48.39	59.88
4	74.24	62.12	67.64	77.79	55.61	64.86	73.88	68.76	71.23	76.72	55.29	64.21
5	73.32	63.37	67.98	76.96	62.9	69.22	67.28	68.56	67.74	74.9	64.68	69.41
6	73.34	64.88	68.85	73.5	60.98	66.66	69.48	70.01	69.72	75.22	68.44	71.67
7	71.75	64.73	68.06	**75.75**	**65.78**	**70.41**	69.73	72.54	71.1	73.24	71.13	72.15
8	72.8	65.81	69.13	75	65.73	70.06	65.92	72.76	69.14	73.34	72.14	72.73
9	72.79	66.15	69.31	73.75	66.26	69.8	59.74	70.37	64.35	68.2	69.73	68.95
10	70.87	65.92	68.31	74.02	66.94	70.3	66.54	72.57	69.39	69.25	72.4	70.76
11	69.85	64.56	67.1	73.51	66.31	69.72	**69.63**	**74.72**	**72.09**	58.73	64.72	60.88
12	70.03	66.5	68.22	70.3	64.94	67.51	66.56	72.07	69.2	69.31	71.41	70.19
13	71.87	66.34	68.99	73.12	67.49	70.19	67.53	72.88	70.09	68.51	74.26	71.22
14	68.17	65.46	66.79	73.76	67.17	70.31	69.04	74.67	71.74	69.99	75.15	72.47
15	70.79	66.88	68.78	72.86	67.33	69.99	65.11	72.38	68.51	64.41	68.57	66.41
16	71.78	66.87	69.24	72.68	67.22	69.84	69.13	71.72	70.25	68.29	72.39	70.27
17	69.81	66.79	68.27	73	67.78	70.29	68.98	74.83	71.78	69.79	71.13	70.22
18	71.31	66.84	69	72.5	67.85	70.1	65.3	70.54	67.81	68.85	73.47	71.08
19	70.53	67.14	68.79	72.49	67.72	70.02	68.89	74.51	71.59	70.73	75.27	72.92
20	71.62	66.36	68.89	72.97	67.91	70.35	67.18	72.6	69.78	69.62	71.2	70.29
21	71.02	66.5	68.69	70.74	65.04	67.77	67.93	74.08	70.87	**71.36**	**76.42**	**73.8**
22	71.91	66.71	69.21	72.82	67.85	70.25	68.82	74.89	71.73	67.16	72.36	69.66
23	**72.37**	**66.66**	**69.4**	72.57	67.25	69.81	68.21	74.31	71.12	69.05	71.25	70.03
24	69.68	66.93	68.28	73.21	67.34	70.15	70.17	73.12	71.61	68.93	72.83	70.79
25	72.04	66.76	69.3	71.8	68.41	70.06	70.18	73.42	71.76	67.18	73.44	70.16

Performance of different undersampling methods on JNLPBA: S1 (Random Under sampling), S2 (Random undersampling considering individual sentences), S3 (proposed method without considering individual sentences), S4 (proposed method) per Desired IRs (DIR) in terms of Recall (R), Precision (P), and F Score (F). The bold values show the best result of each individual method

mostly used algorithm for NER in different domains, it has been used for classifier creation processes by means of MALLET [35] toolkit. The first step after sentence boundary detection was the tokenization of sentences to the appropriate segments and converting the data into the sequences of ⟨Token, Label⟩ forms. We applied a rule-based tokenizer which described in [36] for tokenization. Like all other classification tasks, we needed to extract some features for data; Table 2 shows all the features that are extracted for data. These features are of commonly used features for NER problems in different domains.

As mentioned in Sect. 4, the binary representation of chromosomes is used here in which the length of chromosomes is equals to the total number of tokens from all sentences given in training data. In addition, the start/end index of each sentences in chromosomes is memorized to be used for sentence-based sampling. In addition, the positive samples from each sentence determined in the first step are kept unchanged during sampling process. In fact, the aim of the method is to choose from negative samples to take into account while ignoring all other negatives. Therefore, the final solution would consist of default positive samples plus those negatives which selected by genetic algorithm. To find the optimize values for the involved parameters in GA, we tried to fix them from a reasonable valid range per each parameter by keeping all other parameters unchanged. Table 3 gives information about selected values for algorithm parameters per each corpus. We used uniform crossover and random bit inversing method as mutation. While applying crossover and mutation, the corresponding bits for positive samples should be kept unchanged. To these experiments, a 2.5 GH Core i7 processor with 16 GB RAM is used.

4.1 Results

In this section, we report the results of our experiments on both CoNLL and JNLPBA corpuses. After preparing data sets in tokenized mode and extracting mentioned features, we tuned the parameters of genetic algorithm based on the approach explained in Sect. 4. Then, to find the optimum sampling ratio (desired IR), we applied our under sampling methods several times with different sampling ratios started

Table 5 Effects of different under sampling methods on CoNLL test data

DIR	S1			S2			S3			S4		
	R	*P*	*F*	*R*	*P*	*F*	*R*	*P*	*F*	*R*	*P*	*F*
3	92.39	81.75	86.74	87.27	85.7	86.47	92.39	81.75	86.74	92.17	85.65	88.79
4	92.24	83.84	87.84	87.72	88.63	88.17	92.24	83.84	87.84	92.44	88.05	90.19
5	92.17	85.15	88.52	86.08	87.73	86.89	92.17	85.15	88.52	92.71	90.03	91.35
6	91.91	85.84	88.77	88	90.55	89.26	91.91	85.84	88.77	92.59	90.54	91.55
7	91.66	86.15	88.82	87.03	89.75	88.37	91.66	86.15	88.82	92.51	91.13	91.81
8	91.6	86.46	88.96	87.58	91	89.26	91.6	86.46	88.96	92.64	91.69	92.16
9	91.6	86.56	89.01	87.43	90.89	89.13	91.6	86.56	89.01	92.49	91.86	92.17
10	91.58	86.96	89.21	86.52	90.43	88.43	91.58	86.96	89.21	92.34	92.01	92.18
11	91.38	87.01	89.15	87.11	90.98	89	91.38	87.01	89.15	92.29	92.08	92.18
12	89.8	85.62	87.66	86.63	90.59	88.56	89.8	85.62	87.66	92.24	92.12	92.18
13	91.28	86.85	89.01	86.95	91.17	89.01	91.28	86.85	89.01	92.23	92.27	92.25
14	91.05	87.13	89.05	85.76	90.24	87.94	91.05	87.13	89.05	92.24	92.38	92.31
15	90.99	87.18	89.04	85.82	90.34	88.02	90.99	87.18	89.04	92.1	92.32	92.2
16	91	86.99	88.95	86.42	90.98	88.64	91	86.99	88.95	92.24	92.41	92.32
17	91.22	87.28	89.21	85.42	89.73	87.52	91.22	87.28	89.21	92.11	92.45	92.27
18	91.16	87.13	89.1	87.29	91.84	89.51	91.16	87.13	89.1	92.18	92.43	92.3
19	91.04	87.2	89.08	85.91	90.07	87.94	91.04	87.2	89.08	92.15	92.46	92.3
20	91.15	87.19	89.12	86.73	91.16	88.89	91.15	87.19	89.12	92.1	92.48	92.29
21	90.88	87.48	89.14	86.1	90.19	88.1	90.88	87.48	89.14	92.07	92.47	92.27
22	91.1	87.41	89.21	86.37	90.6	88.44	91.1	87.41	89.21	92.12	92.35	92.24
23	91.4	87.12	89.2	85.55	89.7	87.58	91.4	87.12	89.2	92.03	92.44	92.24
24	90.93	87.38	89.12	85.3	89.87	87.53	90.93	87.38	89.12	92.1	92.44	92.27
25	89.09	84.64	86.8	87.21	91.82	89.46	89.09	84.64	86.8	92	92.48	92.24
26	91.2	87.33	89.22	86.79	91.42	89.04	91.2	87.33	89.22	92.07	92.45	92.26
27	86.63	90.59	88.56	89.45	91.78	90.6	**93.44**	**87.57**	**90.41**	90.94	92.3	91.62
28	86.95	91.17	89.01	**89.38**	**92.03**	**90.69**	93.73	87.15	90.32	94.99	89.39	92.1
29	85.76	90.24	87.94	89.4	91.84	90.6	93.6	87.12	90.24	94.96	90.56	92.71
30	85.82	90.34	88.02	85.04	87.15	86.08	93.66	87.1	90.26	**94.37**	**91.35**	**92.84**
31	86.42	90.98	88.64	88.49	90.89	89.67	93.52	87.1	90.2	94.43	91.22	92.8
32	85.42	89.73	87.52	89.3	91.83	90.55	94.24	86.48	90.19	93.77	90.02	91.86
33	**87.29**	**91.84**	**89.51**	80.55	82.29	81.41	93.98	86.68	90.18	93.84	90.54	92.16
34	85.91	90.07	87.94	89.76	85.5	87.54	91.66	82.94	87.08	93.77	90.66	92.19
35	86.73	91.16	88.89	89.66	87.29	88.46	93.64	86.42	89.89	93.71	90.86	92.26

Performance of different undersampling methods on ConLL : S1 (Random Under sampling), S2 (Random undersampling considering individual sentences), S3 (proposed method without considering individual sentences), S4 (proposed method) per Desired IRs (DIR) in terms of Recall (R), Precision (P), and F Score (F). The bold values show the best result of each individual method

from 3 to the original IR by step 1. During experiments, all evaluations have been done on development data sets of each corpus. After finding the optimum solution (under sampled training data), the classifier which trained with its corresponding training data has used for the test data. To compare and evaluate the method, some other under sampling methods are considered including: random under sampling considering all tokens of sentences together (S1), random under sampling considering sentences individually (S2), and, moreover, we ignored sentence-based concept of under sampling from our method (S3) to see the effect of consideration of individual sentences instead of looking at it in whole. Results of our proposed method have given as S4. Table 4 and 5 show the results per each approach in different sampling ratios using development data for each corpus. The results of using the best found classifiers with different under sampling methods on test data are also shown plus the base line systems' performances are depicted in Table 6.

Referring to Table 6, it is clear that applying under sampling techniques has positive effects on the overall per-

Table 6 Baselines performances on different test data

	JNLPBA				CoNLL			
	R	*P*	*F*	OIR	*R*	*P*	*F*	OIR
Baseline on development data	68.44	67.85	68.14	–	86.43	90.77	88.54	–
Base line on test data	69.65	69.5	69.57	–	88.32	90.91	89.60	–
Best of S1 on development data	72.37	66.66	69.4	23	87.29	91.84	89.51	33
Best of S1 on test data	73.57	67.85	70.59		87.86	92.79	90.26	
Best of S2 on development data	75.75	65.78	70.41	7	89.38	92.03	90.69	28
Best of S2 on test data	76.91	67.74	72.03		91.42	92.53	91.97	
Best of S3 on development data	69.63	74.72	72.09	11	93.44	87.57	90.41	27
Best of S3 on test data	70.75	75.28	72.94		93.78	90.83	92.28	
Best of S4 on development data	71.36	76.42	73.8	21	94.37	91.35	92.84	30
Best of S4 on test data	72.13	75.89	73.96		94.62	92.75	93.68	

formances of NER systems on both different data sets. To investigate the effect of under sampling considering individual sentences instead of whole data set, the differences between results of applying S_1 and S_2 also between S3 and S4 can be considered, and both S_2 and S_4 outperform S_1 and S_4, respectively. Generalization on test data using S_2 outperforms the using S1 about 1.5 in term of F-score for JNLPBA corpus and 1.71 points for CoNLL data set. In addition, applying S_4 on test data improves the results of S_3 more than 1 point for both corpuses. The main reason for this improvement is to considering the distribution of samples through sentences, such that after under sampling process, all contributed sentences in training data will have the same sampling ratio. Moreover, results show that using genetic algorithm for under sampling regardless of considering individual sentences or all sentences together is better than using random under sampling approaches. However, since random under sampling works based on random selection of negative samples, in each time running, its outcome can be different. To make results stable as much as possible, we applied random under sampling approaches (S_1 and S_2) 5 times and took their average, as shown in Tables 4 and 5. Considering the results of different methods, our proposed approach showed the better impacts on the overall performances of NER system in comparison with the others.

5 Conclusion

Named entity recognition tries to identify entities of interests in given text and the mostly used strategy to deal with it is the machine learning approach. With respect to the importance of NER systems in the different subtasks of information extraction, any improvement on the performances of NER systems would directly affect the quality of those subtasks. Since usually the number of important and desired named entities in a text is much less than other unimportant segments of text, class imbalance problem would be the innate property of NER tasks. In this research, we proposed a heuristic under sampling method using genetic algorithm to make balance between number of mentions of interest (positive samples) and other undesired segments of text (negative samples) as much as possible. The major difference between our method and other current under sampling methods is about whether given data taken into account in sampling process. The current methods mostly consider all sentences together to apply sampling processes on data, while our method looks at individual sentences and apply under sampling on each individual sentence by means of heuristic approach to the selection of negative samples for being in training data. Two different corpuses from different contexts are used to evaluate method proposed, namely, CoNLL from news wire domain and JNLPBA from biological context. Experiments shows that suggested approach has achieved better results than other commonly used undersampling methods in terms of F-score. Moreover, we illustrated that considering individual sentences in under sampling process instead of taking all of them together improve the performance of NER systems in comparison with the methods in which this concept is ignored.

References

1. Westergaard, D., Stærfeldt, H.-H., Tønsberg, C., Jensen, L.J., Brunak, S.: Text mining of 15 million full-text scientific articles. bioRxiv, 162099 (2017). https://doi.org/10.1101/162099
2. Holzinger, A., Schantl, J., Schroettner, M., Seifert, C., Verspoor, K.: Biomedical text mining: state-of-the-art, open problems and future challenges. In: Holzinger, A., Jurisica, I. (eds.) Interactive Knowledge Discovery and Data Mining in Biomedical Informatics, Lecture Notes in Computer Science, vol. 8401,pp. 271–300. Springer, Berlin (2014). https://doi.org/10.1007/978-3-662-43968-5_16
3. Munkhdalai, T., Li, M., Batsuren, K., Park, H.A., Choi, N.H., Ryu, K.H.: Incorporating domain knowledge in chemical and biomedical

named entity recognition with word representations. J. Cheminform. **7**(1), S9 (2015)

4. Marrero, M., Urbano, J., Sánchez-Cuadrado, S., Morato, J., Gómez-Berbís, J.M.: Named entity recognition: fallacies, challenges and opportunities. Comput. Stand. Interfaces **35**(5), 482–489 (2013)
5. Yang, Q., Wu, X.: 10 challenging problems in data mining research. Int. J. Inf. Technol. Decis. Mak. **5**(04), 597–604 (2006)
6. Akkasi, A., Varoğlu, E., Dimililer, N.: Balanced undersampling: a novel sentencebased undersampling method to improve recognition of named entities in chemical and biomedical text. Appl. Intell. 1–14 (2017). https://doi.org/10.1007/s10489-017-0920-5
7. Tang, B., Feng, Y., Wang, X., Wu, Y., Zhang, Y., Jiang, M., Xu, H.: A comparison of conditional random fields and structured support vector machines for chemical entity recognition in biomedical literature. J. Cheminform. **7**(S1), S8 (2015)
8. Nanni, L., Fantozzi, C., Lazzarini, N.: Coupling different methods for overcoming the class imbalance problem. Neurocomputing **158**, 48–61 (2015)
9. Lemnaru, E. C.: Strategies for dealing with real world classification problems. Doctoral dissertation, Technical University of Cluj-Napoca (2012)
10. Japkowicz, N.: The class imbalance problem: significance and strategies. In: Proceedings of the 2000 International Conference on Artificial Intelligence (IC-AI'2000): Special Track on Inductive Learning Las Vegas, Nevada (2000)
11. He, H., Ma, Y. (eds.): Imbalanced Learning: Foundations, Algorithms, and Applications. Wiley, New York (2013)
12. Zhu, B., Baesens, B., vanden Broucke, S.K.: An empirical comparison of techniques for the class imbalance problem in churn prediction. Inf. Sci. **408**, 84–99 (2017)
13. Longadge, R., Dongre, S.: Class imbalance problem in data mining review (2013). arXiv preprint arXiv:1305.1707
14. Chawla, N. V.: Data mining for imbalanced datasets: an overview. In: Data Mining and Knowledge Discovery Handbook, pp. 875–886. Springer (2009). https://doi.org/10.1007/978-0-387-09823-4_45
15. Tomek, I.: Two Modifications of CNN. IEEE Trans. Syst. Man Commun. SMC **6**, 769–772 (1976)
16. Kumar, R.R., Viswanath, P., Bindu, C.S.: Nearest neighbor classifiers: a review. Int. J. Comput. Intell. Res. **13**(2), 303–311 (2017)
17. Faris, H.: Neighborhood cleaning rules and particle swarm optimization for predicting customer churn behavior in telecom industry. Int. J. Adv. Sci. Technol. **68**, 11–22 (2014)
18. Kubat, M., Matwin, S.: Addressing the curse of imbalanced training sets: one-sided selection. In: Proceedings of the Fourteenth International Conference on Machine Learning, pp. 179–186. Morgan Kaufmann (1997)
19. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: synthetic minority over-sampling technique. J. Artif. Intell. Res. **16**, 321–357 (2002)
20. Han, H., Wang, W. Y., Mao, B. H.: Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. In: Advances in Intelligent Computing, pp. 878–887. Springer, Berlin (2005). https://doi.org/10.1007/11538059_91
21. Lim, P., Goh, C.K., Tan, K.C.: Evolutionary cluster-based synthetic oversampling ensemble (eco-ensemble) for imbalance learning. IEEE Trans. Cybern. **47**, 2850–2861(2016)
22. Braytee, A., Liu, W., Kennedy, P.: A cost-sensitive learning strategy for feature extraction from imbalanced data. In: International Conference on Neural Information Processing, pp. 78–86. Springer International Publishing (2016). https://doi.org/10.1007/978-3-319-46675-0_9
23. Chawla, N. V., Lazarevic, A., Hall, L. O., Bowyer, K. W.: SMOTEBoost: Improving prediction of the minority class in boosting. In: Knowledge Discovery in Databases: PKDD 2003, pp. 107–119. Springer, Berlin (2003). https://doi.org/10.1007/978-3-540-39804-2_12
24. Williams, G., Chen, H.: Stratified over-sampling bagging method for random forests on imbalanced data. In: Intelligence and Security Informatics: 11th Pacific Asia workshop. PAISI 2016, Auckland, New Zealand, April 19, 2016, Proceedings, vol. 9650, p. 63. Springer (2016). https://doi.org/10.1007/978-3-319-31863-9_5
25. Ahachad, A., Álvarez-Pérez, L., Figueiras-Vidal, A.R.: Boosting ensembles with controlled emphasis intensity. Pattern Recognit. Lett. **88**, 1–5 (2017)
26. Tomanek, K., Hahn, U.: Reducing class imbalance during active learning for named entity annotation. In: Proceedings of the Fifth International Conference on Knowledge Capture, pp. 105–112. ACM (2009). https://doi.org/10.1145/1597735.1597754
27. Gliozzo, A.M., Giuliano, C., Rinaldi, R.: Instance filtering for entity recognition. ACM SIGKDD Explor. Newsl. **7**(1), 11–18 (2005)
28. Mitchell, M.: An Introduction to Genetic Algorithms. MIT Press: Cambridge, MA (1998)
29. Dasgupta, D., Michalewicz, Z. (eds.): Evolutionary Algorithms in Engineering Applications. Springer Science & Business Media, New York (2013)
30. http://www.obitko.com/tutorials/genetic-algorithms/crossover-mutation.php. Accessed 12 Aug 2017
31. Sang, E.F., Veenstra, J.: Representing text chunks. In: Proceedings of the Ninth Conference on European Chapter of the Association for Computational Linguistics, pp. 173–179. Association for Computational Linguistics (1999)
32. Powers, D.M.W.: Evaluation: from precision, recall and f-measure to roc, informedness, markedness & correlation. J. Mach. Learn. Technol. **2**(1), 37–63 (2011)
33. Tjong Kim Sang, E.F., De Meulder, F.: Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In CoNLL-2003 (2003)
34. Kim, J.D., Ohta, T., Tsuruoka, Y., Tateisi, Y., Collier, N.: Introduction to the bio-entity recognition task at JNLPBA. In: Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications, pp. 70–75. Association for Computational Linguistics (2004)
35. McCallum, Andrew Kachites. MALLET: A Machine Learning for Language Toolkit (2002). http://mallet.cs.umass.edu. Accessed 5 Oct 2017
36. Akkasi, A., Varoğlu, E., Dimililer, N.: ChemTok: a new rule based tokenizer for chemical named entity recognition. BioMed Res. Int. **2016** (2016)