A Hybrid Approach for Entity Extraction in Code-Mixed Social Media Data

Deepak Gupta
Comp. Sc. & Engg. Deptt.
IIT Patna, India
deepak.pcs16@iitp.ac.in

Shweta
Comp. Sc.& Engg. Deptt.
IIT Patna, India
shweta.pcs14@iitp.ac.in

Shubham Tripathi
Electrical Engg. Deptt.
MNIT Jaipur, India
stripathi1770@gmail.com

Asif Ekbal
Comp. Sc. & Engg. Deptt.
IIT Patna, India
asif@iitp.ac.in

Pushpak Bhattacharyya
Comp. Sc. & Engg. Deptt.
IIT Patna, India
pb@iitp.ac.in

ABSTRACT

Entity extraction is one of the important tasks in various natural language processing (NLP) application areas. There has been a significant amount of works related to entity extraction, but mostly for a few languages (such as English, some European languages and few Asian languages) and doamins such as newswire. Nowadays social media have become a convenient and powerful way to express one's opinion and sentiment. India is a diverse country with a lot of linguistic and cultural variations. Texts written in social media are informal in nature, and perople often use more than one script while writing. User generated content such as tweets, blogs and personal websites of people are written using Roman script or sometimes users may use both Roman as well as indigenous scripts. Entity extraction is, in general, a more challenging task for such an informal text, and mixing of codes further complicates the process. In this paper, we propose a hybrid approah for enity extraction from code mixed language pairs such as English-Hindi and English-Tamil. We use a rich linguistic feature set to train Conditional Random Field (CRF) classifier. The output of classifier is post-processed with a carefully hand-crafted feature set. The proposed system achieve the F-scores of 62.17% and 44.12% for English-Hindi and English-Tamil language pairs, respectively. Our system attains the best F-score among all the systems submitted in Fire 2016 shared task for the English-Tamil language pairs.

CCS Concepts

•**Computing methodologies** → **Natural Language Processing;** •**Information System** → *Information Extraction;* •**Algorithm** → Conditional Random Field(CRF);

Keywords

Code-mixing, Entity Extraction, Named Entity Recognition, Conditional Random Field(CRF), Indian Language, Social Media data

1. INTRODUCTION

Code-mixing refers to the mixing of two or more languages or language varieties. Code switching and code mixing are interchangeably used by the peoples. With the availability of easy internet access to people, social media involvement has been increased a lot. Over the past decade, Indian language content on various media types such as blogs, email, website, chats has increased significantly. And it is observed that with the advent of smart phones more people are using social media such as whatsapp, twitter, facebook to share their opinion on people, products, services, organizations, governments. The abundant of social media data created many new opportunities for information access, but also many new challenges. To deals with these challenges many of the research is going on and it have become one of the prime present-day research areas. Non-English speakers, especially Indians, do not always use Unicode to write something in social media in Indian languages. Instead, they use their roman script or transliteration and frequently use English words or phrases through code-mixing and also often mix multiple languages in addition to anglicisms to express their thoughts. Although English is the principal language for social media communications, there is a necessity to develop mechanism for other languages, including Indian languages. According to the Indian constitution there are 22 official language in India. However Census of India of 2001, reported India has 122 major languages and 1599 other languages. The 2001 Census recorded 30 languages which were spoken by more than a million native speakers and 122 which were spoken by more than $10,000$ people. Language diversity and dialect changes instigate frequent code-mixing in India. Hence, Indians are multi-lingual by adaptation and necessity, and frequently change and mix languages in social media contexts, which poses additional difficulties for automatic social media text processing on Indian language. The growth of Indian language content is expected to increase by more than 70% every year. Hence there is a great need to process this huge data automatically. Named Entity Recognition (NER) is one of the key information extraction tasks, which is concerned with identifying names of entities such as people, location, organization and product. It can be divided into two main phases: entity detection and entity typing (also called classification)[7]. Recently, Information extraction over micro-blogs have become an active research topic [4], following early experiments which showed this genre to be extremely challenging for state-of-the-art algorithms[5, 2]. For instance, named entity recognition methods typically have 85-90% accuracy on longer texts, but 30-50% on tweets[16]. First, the shortness of micro-blogs (maximum 140 characters for tweets) makes them hard to

interpret. Consequently, ambiguity is a major problem since semantic annotation methods cannot easily make use of coreference information. Unlike longer news articles, there is a low amount of discourse information per microblog document, and threaded structure is fragmented across multiple documents, flowing in multiple directions. Second, microtexts exhibit much more language variation, tend to be less grammatical than longer posts, contain unorthodox capitalization, and make frequent use of emoticons, abbreviations and hashtags, which can form an important part of the meaning. To combat these problems, research has focused on microblog-specific information extraction algorithms (e.g. named entity recognition for Twitter using CRFs[16] or hybrid methods[18]. Particular attention is given to micro-text normalization[8], as a way of removing some of the linguistic noise prior to part-of-speech tagging and entity recognition. In literature primarily machine learning and rule based approach has been used for named entity recognition (NER). Machine learning (ML) based techniques for NER make use of a large amount of NE annotated training data to acquire higher level knowledge by extracting relevant features from the labeled data. Several ML techniques have already been applied for the NER tasks such as Support vector vector classifier[9], Maximum Entropy[3, 10], Markov Model(HMM)[1], Conditional Random Field (CRF)[12] etc. The rule based techniques have also been explored in the task by[6, 13, 19]. The hybrid approaches that combines different Machine learning based approaches are also used by Rohini et al.[17] by combining Maximum entropy, Hidden Markov Model and handcrafted rules to build an NER system. Entity extraction has been actively researched for over 20 years. Most of the research has, however been focused on resource rich languages, such as English, French and Spanish. However entity extraction and recognition from social media text on for Indian language have been introduced on FIRE-15 workshop[15]. The code-mixing entity extraction from social media text on for Indian mix language introduced in the FIRE-2016. Entity extraction from code-mixing social media text poses some key challenge which are as follows:

1. The released data set contains code mixing as well as uni-language utterance.
2. Set of entity are not limited to only traditional set of entity e.g. Person Name, Location Name, Organization Name etc. There are 22 different types of entities are there to extract from text.
3. There are lack of resources/tools for Indian languages. code-mixing makes problems more difficult for preprocessing tasks required for NER such as sentence splitter, tokenization, part-of-speech tagging and chunking etc.

2. PROBLEM DEFINITION

The problem definition of code-mixing entity extraction comprises two sub-problem entity extraction and entity classification. Mathematically the problem of code-mixing entity extraction can be described as follows: Lets S is code-mixing sentence having n tokens $t_1, t_2 \dots t_n$. E is the set of k pre-defined entity $E = \{E_1, E_2, \dots E_k\}$.

1. **Entity Extraction step:** Extract set of tokens $T_E = \{t_i, t_j \dots t_k\}$ from S whose characteristics is similar to any of the entity from entity set E.
2. **Entity classification step:** Classify each of the tokens of set T_E into one of the entity type from entity set E.

3. DATASET

There are two language pair data set available to evaluate the system performance. It was crawled from tweeter, mainly the crawled tweet are in English-Hindi and English-Tamil language mix. There are 22 types of entities present in the training data set in which the majority of entities are from 'Entertainment', 'Person' 'Location' and 'Organization'. The statistics of the training data set is shown in the Table-1. We have also shown some of the sample tweets from both Language pair in Table-2. English-Tamil language pair tweets contains some of the tweets from only Tamil language only. English-Hindi tweet data set contains total 2700 tweets from 2699 tweeter users. Similarly English-Tamil tweet data set contains total 2183 tweets from 1866 tweeter users.

Entities	**English-Hindi**	**English-Tamil**
	# Entity	# Entity
COUNT	132	94
PLANTS	1	3
PERIOD	44	53
LOCOMOTIVE	13	5
ENTERTAINMENT	810	260
MONEY	25	66
TIME	22	18
LIVTHINGS	7	16
DISEASE	7	5
ARTIFACT	25	18
MONTH	10	25
FACILITIES	10	23
PERSON	712	661
MATERIALS	24	28
LOCATION	194	188
YEAR	143	54
DATE	33	14
ORGANIZATION	109	68
QUANTITY	2	0
DAY	67	15
SDAY	23	6
DISTANCE	0	4
Total	**2413**	**1624**

Table 1: Training dataset statistics for both the language pair. ENTERTAINMENT type of entity has the maximum no. of entity in both the language pair.

3.1 Conditional Random Field(CRF)

Lafferty et al.[11] define the the probability of a particular label sequence y given observation sequence x to be a normalized product of potential functions, each of the form

$$exp(\sum_j \lambda_j t_j(y_{i-1}, y_i, x, i) + \sum_k \mu_k s_k(y_i, x, i)) \qquad (1)$$

Langauage Pair	Sample Tweet
English-Hindi	@YOUniqueDoc Nandu,muje shaq hai ki humari notice ke bagair tere ghar ke secret route ki help se you met kya. My intution is never wrong
	A RiftWood Productions presents to you the season finale of Le' Bill & Giddy,La Muje'r
English-Tamil	Ungala nenachu neengale romba proud ah feel panra vishayam enna ?! Post ur comments. Will be read on sun music at 5pm live ;)
	IruMugan will be a Class + Mass movie like Thani Oruvan. The biggest plus is the screenplay - Thambi Ramaiyah..

Table 2: Sample tweet from both language pair

where $t_j(y_{i-1}, y_i, x, i)$ is a transition feature function of the entire observation sequence and the labels at positions i and $i-1$ in the label sequence; $s_k(y_i, x, i)$ is a state feature function of the label at position i and the observation sequence; and λ_j and μ_k are parameters to be estimated from training data. When defining feature functions, we construct a set of real-valued features $b(x, i)$ of the observation to expresses some characteristic of the empirical distribution of the training data that should also hold of the model distribution. An example of such a feature is

$$b(x,i) = \begin{cases} 1 & \text{if the observation word at position } i \text{ is } \textit{'religion'} \\ 0 & \text{otherwise} \end{cases}$$

Each feature function takes on the value of one of these real-valued observation features $b(x, i)$ if the current state (in the case of a state function) or previous and current states (in the case of a transition function) take on particular values. All feature functions are therefore real-valued. For example, consider the following transition function:

$$t_j(y_{i-1}, y_i, x, i) = \begin{cases} b(x,i) & \text{if } y_{i-1} = \textit{B-ORG} \text{ and } y_i = \textit{I-ORG} \\ 0 & \text{otherwise} \end{cases}$$

This allows the probability of a label sequence y given an observation sequence x to be written as

$$P(y|x, \lambda) = \frac{1}{Z(x)} exp(\sum_j \lambda_j F_j(y, x)) \quad (2)$$

where $F_j(y, x)$ can be written as follows:

$$F_j(y, x) = \sum_{i=1}^{n} f_j(y_{i-1}, y_i, x, i) \quad (3)$$

where each $f_j(y_{i-1}, y_i, x, i)$ is either a state function $s(y_{i-1}, y_i, x, i)$ or a transition function $t(y_{i-1}, y_i, x, i)$.

4. FEATURE EXTRACTION

The proposed system uses an exhaustive set of features for NE recognition. These features are described below.

1. **Context word**: Local contextual information is useful to determine the type of the current word. We use the contexts of previous two and next two words as features.

2. **Character n-gram**: Character n-gram is a contiguous sequence of n characters extracted from a given word. The set of n-grams that can be generated for a given token is basically the result of moving a window of n characters along the text. We extracted character n-grams of length *one* (unigram), *two*(bigram) and *three* (trigram), and use these as features of the classifiers.

3. **Word normalization** : Words are normalized in order to capture the similarity between two different words that share some common properties. Each uppercase letter is replaced by 'A', lowercase by 'a' and number by '0'.

Words	Normalization
NH10	AA00
Maine	Aaaaa
NCR	AAA

4. **Prefix and Suffix**: Prefix and suffix of fixed length character sequences (here, 3) are stripped from each token and used as features of the classifier.

5. **Word Class Feature**: This feature was defined to ensure that the words having similar structures belong to the same class. In the first step we normalize all the words following the process as mentioned above. Thereafter, consecutive same characters are squeezed into a single character. For example, the normalized word *AAAaaa* is converted to *Aa*. We found this feature to be effective for the biomedical domain, and we directly adapted this without any modification.

6. **Word Position:** In order to capture the word context in the sentence, we have used a numeric value to indicate the position of word in the sentence. The normalized position of word in the sentence is used as a features. The feature values lies in the ranges between 0 and 1.

7. **Number of Upper case Characters:** This features takes into account the number of uppercase alphabets in the word. The feature is relative in nature and ranges between 0 and 1.

8. **Test Word Probability:** This feature finds the probability of the test word to be labeled the same as in training data. The length of this vector feature is the total number of labels or output tags, where each bit represents an output tag, It is initialized with 0. If the test word does not appear in training, every bits retain their initially marked value 0. Based on the probability value, we have two features:

   (a) **Top@1-Probability:** For the output tag with highest probability, its corresponding bit in the feature vector is set to 1. All other bits remain as 0.

   (b) **Top@2-Probability:** For the output tag with highest and second highest probability, their corresponding bit are set to 1 in the feature vector. All other bits remain as 0.

9. **Binary Features:** These binary features are identified after the through analysis of training data.

(a) **isSufficientLength:** Since most of the entity from training data have a significant length. Therefore we set a binary feature to fire when the length of token is greater than a specific threshold value. The threshold value 4 is used to extract the binary features.

(b) **isAllCapital:** This value of this feature is set when all the character of current token is in uppercase.

(c) **isFirstCharacterUpper:** This value of this feature is set when the first character of current token is in uppercase..

(d) **isInitCap**: This feature checks whether the current token starts with a capital letter or not. This provides an evidence for the target word to be of NE type for the English language.

(e) **isInitPunDigit**: We define a binary-valued feature that checks whether the current token starts with a punctuation or a digit. It indicates that the respective word does not belong to any language. Few such examples are *4u,:D*, *:P* etc.

(f) **isDigit**: This feature is fired when the current token is numeric.

(g) **isDigitAlpha**: We define this feature in such a way that checks whether the current token is alphanumeric. The word for which this feature has a true value has a tendency of not being labeled as any named entity type.

(h) **isHashTag**: Since we are dealing with tweeter data , therefore we encounter a lot of hashtag is tweets. We define the binary feature that checks whether the current token starts with # or not.

Input : Disease Name list as *DN*
Living things list as *LT*;
Special Days list as *SD*
List of pair obtained from CRF as L(W,C)
Output: Post-processed list of (token,label) pair
obtained after post-processing as PL(W,C')
PL(W,C')=L(W,C)
while *L(W,C) is non-empty* **do**
  if *DN contains* W_i **then**
    C=DISEASE;
    C'=C;
  else if *LT contains* W_i **then**
    C=LIVINGTHINGS;
    C'=C;
  else if *SD contains* W_i **then**
    C=SPECIALDAYS;
    C'=C;
  else
    C'=C;
end
return PL(W,C');

Algorithm 1: Post-processing algorithm for code mixed data set

5. EXPERIMENTAL SETUP

To extract the entity from code mixed data we have followed three step approach, which are described in this section. Fig-1 shows a architecture diagram of our proposed approach.

5.1 Pre-processing

Pre-processing stage is an important task before applying any classifier. The release data set was in raw text sentence having the list of entity. There are two step was performed as part of pre-processing.

1. **Tokenization:** Since the data set are crawled from Twitter therefore a suitable tokenizer which can deals with social media data need to be used. We used the CMU PoS tagger[14] for tokenization and PoS tagging of tweets.

2. **Token Encoding:** We used the IOB encoding for tagging token. The IOB format (Inside, Outside, Beginning) is a common tagging format for tagging tokens in a chunking task in computational linguistics (ex. Named Entity Recognition). The B- prefix before a tag indicates that the tag is the beginning of a chunk, and an I- prefix before a tag indicates that the tag is inside a chunk. An O tag indicates that a token belongs to no chunk.

5.2 Sequence Labeling

In literature primarily HMM, MEMM and CRF has been used for sequence labeling task. Here we used the CRF classifier for label the sequence of token. The features set described in section-4 were finally formulated to provide as input to our CRF classifier[11]. We used the CRF^{++}[1] implementation of CRF. The default parameter setting was used to carry out the experiment.

5.3 Post-processing

The rule and dictionary based post-processing was performed followed by labeling obtained from CRF classifier. The detailed explanation are as follows:

English-Hindi

For English-Hindi code mixed data we used dictionary of *Disease Name*, *Living Things* and *Special Days*. A list consist of 250 disease name was obtained from wiki page[2]. A manual list are created of 668 living things from different web page source. Similarly for list of special days, a manual list of 92 special days was obtained from this website[3]. The dictionary element was fired in the order as mentioned in Algorithm-1. At last regular expressions are formed to correct *PERIOD*, *MONEY* and *TIME* on post-processed output.

English-Tamil

Since none of our team member are native Tamil speaker so we could not do deep error analysis of CRF predicted output. We used the same dictionary which was used in English-Hindi data set, because those dictionary contains only English word. Finally regular expressions are formed to correct

[1]https://taku910.github.io/crfpp/
[2]https://simple.wikipedia.org/wiki/List_of_diseases
[3]www.drikpanchang.com/calendars/indian/indiancalendar.html

S. No.	Team	Run-1			Run-2			Run-3			Best-Run		
		P	R	F	P	R	F	P	R	F	P	R	F
1	Irshad-IIITHyd	80.92	59	68.24	NA			NA			80.92	59.00	68.24
2	**Deepak-IITPatna**	**81.15**	**50.39**	**62.17**	**NA**			**NA**			**81.15**	**50.39**	**62.17**
3	VeenaAmritha-T1	75.19	29.46	42.33	75	29.17	42.00	79.88	41.37	54.51	79.88	41.37	54.51
4	BharathiAmritha-T2	76.34	31.15	44.25	77.72	31.84	45.17	NA			77.72	31.84	45.17
5	Rupal-BITSPilani	58.66	32.93	42.18	58.84	35.32	44.14	59.15	34.62	43.68	58.84	35.32	44.14
6	SomnathJU	37.49	40.28	38.83	NA			NA			37.49	40.28	38.83
7	Nikhil-BITSHyd	59.28	19.64	29.50	61.8	26.39	36.99	NA			61.80	26.39	36.99
8	ShivkaranAmritha-T3	48.17	24.9	32.83	NA			NA			48.17	24.90	32.83
9	AnujSaini	72.24	18.85	29.90	NA			NA			72.24	18.85	29.90

Table 3: Official results obtained by the various teams participated in the CMEE-IL task- FIRE 2016 for code mixed English-Hindi language pair. Here P, R and F denotes precision, recall and F-score respectively.

S. No.	Team	Run-1			Run-2			Run-3			Best-Run		
		P	R	F	P	R	F	P	R	F	P	R	F
1	**Deepak-IITPatna**	**79.92**	**30.47**	**44.12**	**NA**			**NA**			**79.92**	**30.47**	**44.12**
2	VeenaAmritha-T1	77.38	8.72	15.67	74.74	9.93	17.53	79.51	21.88	34.32	79.51	21.88	34.32
3	BharathiAmritha-T2	77.7	15.43	25.75	79.56	19.59	31.44	NA			79.56	19.59	31.44
4	RupalBITSPilani-R2	58.66	10.87	18.20	58.71	12.21	20.22	58.94	11.94	19.86	58.71	12.21	20.22
5	ShivkaranAmritha-T3	47.62	13.42	20.94	NA			NA			47.62	13.42	20.94

Table 4: Official results obtained by the various teams participated in the CMEE-IL task- FIRE 2016 for code mixed English-Tamil language pair. Here P, R and F denotes precision, recall and F-score respectively.

the *PERIOD*, *MONEY* and *TIME* on post-processed output.

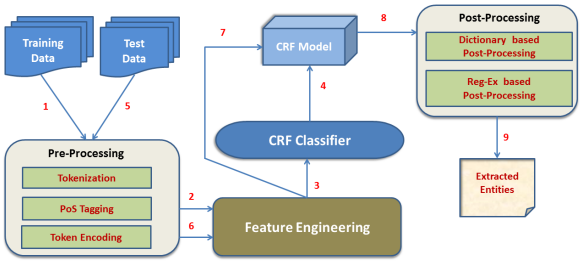

Figure 1: Proposed model architecture for Code-mixed entity extraction

6. RESULT & ANALYSIS

An entity extraction model for English-Hindi & English-Tamil language pair are trained by using CRF as base classifier. We test our system on the test data for the concerned language pair. The proposed approach was used to extract the entity from both the language pair data.

The developed entity extraction and identification system has been evaluated using the precision(P), recall (R) and F-measure (F). The organizers of the CMEE-IL task FIRE 2016, released the data in two phases: in the first phase, training data is released along with the corresponding NE annotation file. In the second phase, the test data is released and no NE annotation file is provided. The extracted NE annotation file for test data was finally sent to the organizers for evaluation. The organizers evaluate the different runs submitted by the various teams and send the official results to the participating teams.

The official results for English-Hindi language pair are shown in Table-3. Our system(Deepak-IITPatna) performance are shown in bold font. Our system got the highest Precision of 81.15% among all the submitted system. The proposed approach achieved F-score of 62.17% on English-Hindi language pair. Table-4 shows the official result for English-Tamil language pair data set. Our system(Deepak-IITPatna) performance are shown in bold font. Our system is the best performing system among all the submitted system. We achieved the 79.92%, 30.47% and 44.12% precision(p), recall(r) and F-score(f) respectively. The reason for lower F-score on Tamil-English could be the lack of good features which can help to recognize a Tamil word as named entity.

7. CONCLUSION & FUTURE WORK

This paper describes a code mixed named entity recognition from Social media text in English-Hindi and English-Tamil language pair data. Our proposed approach is a hybrid model of machine learning and rule based system. The experimental results show that our system is the best performer among the systems participated in the CMEE-IL task for code mixed English-Tamil language pair. For English-Hindi language pair system achieved the highest precision value 81.15% among all the submitted system. In future we would like to build a more robust code mixed NER system by using deep learning system.

8. REFERENCES

[1] D. M. Bikel, S. Miller, R. Schwartz, and R. Weischedel. Nymble: a high-performance learning name-finder. In *Proceedings of the fifth conference on Applied natural language processing*, pages 194–201. Association for Computational Linguistics, 1997.

[2] K. Bontcheva, L. Derczynski, and I. Roberts. Crowdsourcing named entity recognition and entity linking corpora. *The Handbook of Linguistic Annotation (to appear)*, 2014.

[3] A. Borthwick. *A maximum entropy approach to named entity recognition.* PhD thesis, Citeseer, 1999.

[4] A. E. Cano Basave, A. Varga, M. Rowe, M. Stankovic, and A.-S. Dadzie. Making sense of microposts (# msm2013) concept extraction challenge. 2013.

[5] L. Derczynski, D. Maynard, G. Rizzo, M. van Erp, G. Gorrell, R. Troncy, J. Petrak, and K. Bontcheva. Analysis of named entity recognition and linking for tweets. *Information Processing & Management*, 51(2):32–49, 2015.

[6] R. Grishman. The nyu system for muc-6 or where's the syntax? In *Proceedings of the 6th conference on Message understanding*, pages 167–175. Association for Computational Linguistics, 1995.

[7] R. Grishman and B. Sundheim. Message understanding conference-6: A brief history. In *Proceedings of the 16th Conference on Computational Linguistics - Volume 1*, COLING '96, pages 466–471, Stroudsburg, PA, USA, 1996. Association for Computational Linguistics.

[8] B. Han and T. Baldwin. Lexical normalisation of short text messages: Makn sens a# twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 368–378. Association for Computational Linguistics, 2011.

[9] H. Isozaki and H. Kazawa. Efficient support vector classifiers for named entity recognition. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics, 2002.

[10] N. Kumar and P. Bhattacharyya. Named entity recognition in hindi using memm. *Techbical Report, IIT Mumbai*, 2006.

[11] J. Lafferty, A. McCallum, and F. C. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. 2001.

[12] W. Li and A. McCallum. Rapid development of hindi named entity recognition using conditional random fields and feature induction. *ACM Transactions on Asian Language Information Processing (TALIP)*, 2(3):290–294, 2003.

[13] D. McDonald. Internal and external evidence in the identification and semantic categorization of proper names. *Corpus processing for lexical acquisition*, pages 21–39, 1996.

[14] O. Owoputi, B. O'Connor, C. Dyer, K. Gimpel, N. Schneider, and N. A. Smith. Improved part-of-speech tagging for online conversational text with word clusters. Association for Computational Linguistics, 2013.

[15] P. R. Rao, C. Malarkodi, and S. L. Devi. Esm-il: Entity extraction from social media text for indian languages@ fire 2015–an overview.

[16] A. Ritter, S. Clark, O. Etzioni, et al. Named entity recognition in tweets: an experimental study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1524–1534. Association for Computational Linguistics, 2011.

[17] R. Srihari, C. Niu, and W. Li. A hybrid approach for named entity and sub-type tagging. In *Proceedings of the sixth conference on Applied natural language processing*, pages 247–254. Association for Computational Linguistics, 2000.

[18] M. Van Erp, G. Rizzo, and R. Troncy. Learning with the web: Spotting named entities on the intersection of nerd and machine learning. In *# MSM*, pages 27–30, 2013.

[19] T. Wakao, R. Gaizauskas, and Y. Wilks. Evaluation of an algorithm for the recognition and classification of proper names. In *Proceedings of the 16th conference on Computational linguistics-Volume 1*, pages 418–423. Association for Computational Linguistics, 1996.