Mobile phone name extraction from internet forums: a semi-supervised approach

Yao, Yangjie; Sun, Aixin

2015

Yao, Y., & Sun, A. (2016). Mobile phone name extraction from internet forums: a semi-supervised approach. World Wide Web, 19(5), 783-805.

https://hdl.handle.net/10356/83415

https://doi.org/10.1007/s11280-015-0361-1

World Wide Web Journal (Draft)

Mobile Phone Name Extraction from Internet Forums: A Semi-supervised Approach

Yangjie Yao · Aixin Sun

Received: date / Accepted: date

Abstract Collecting users' feedback on products from Internet forums is challenging because users often mention a product with informal abbreviations or nicknames. In this paper, we propose a method named GREN to *recognize* and *normalize* mobile phone names from domain-specific Internet forums. Instead of directly recognizing phone names from sentences as in most named entity recognition tasks, we propose an approach to generating candidate names as the first step. The candidate names capture short forms, spelling variations, and nicknames of products, but are not noise free. To predict whether a candidate name mention in a sentence indeed refers to a specific phone model, a Conditional Random Field (CRF)-based name recognizer is developed. The CRF model is trained by using a large set of sentences obtained in a *semi-automatic* manner with minimal manual labeling effort. Lastly, a rule-based name normalization component maps a recognized name to its formal form. Evaluated on more than 4000 manually labeled sentences with about 1000 phone name mentions, GREN outperforms all baseline methods. Specifically, it achieves precision and recall of 0.918 and 0.875 respectively, with the best feature setting. We also provide detailed analysis of the intermediate results obtained by each of the three components in GREN.

Keywords Mobile phone · Name recognition and normalization · Internet forum

An extended abstract of this work [25] was presented at the SIGIR Symposium on IR in Practice, Gold Coast, July 2014.

Y. Yao
School of Computer Engineering, Nanyang Technological University, Singapore
E-mail: yyao002@e.ntu.edu.sg

A. Sun
School of Computer Engineering, Nanyang Technological University, Singapore
E-mail: axsun@ntu.edu.sg

1 Introduction

Reading relevant discussions and reviews has become a common practice for many users before they purchase consumer products like mobile phones and digital cameras. Many users also seek for recommendations by posting questions on Internet forums and social networking sites (*e.g.,* Facebook, Google+, and Twitter). Such questions, review comments, and discussions are important resources for product providers to better understand consumers' concerns or requirements and to further improve their products or marketing strategies. Because social networking sites often limit data crawling for privacy or other reasons, our discussion in this paper will be focusing on publicly accessible Internet forums.

Crawling product reviews from dedicated review sites (*e.g.,* Epinions, Amazon, and CNET reviews) is relatively straightforward because review reports about one specific product are usually well organized. Collecting user feedback (*e.g.,* questions, comments, and comparison with other products) about *a specific product* (*e.g.,* a given mobile phone model) from Internet forums is much more challenging. One reason is that user feedback about a specific product often spreads in different discussion threads in forums. More importantly, users often mention the same product with a large number of name variations containing informal abbreviations, misspellings, and nicknames. In this paper, we focus on mobile phones, a kind of consumer products that are widely discussed in Internet forums. As an example, Table 1 lists 25 name variations of *"Samsung Galaxy SIII"* (both LTE and non-LTE models), each used by at least 10 users in an Internet forum used in our experiments. We also observe more than 6 forms of misspells of "Ericsson" when users mention Sony Ericsson phones. On the other hand, we argue that user discussions about a mobile phone in Internet forums are at least equally important as user reviews from review websites. A user may choose not to buy a phone (hence will

Table 1 Name variations of *"Samsung Galaxy SIII"*

Name variation	#users	Name variation	#users
1. galaxy s3	553	14. lte s3	46
2. s3 lte	343	15. galaxy s3 lte	45
3. samsung galaxy s3	284	16. s3 non lte	32
4. s iii	242	17. samsung galaxy siii	32
5. galaxy s iii	225	18. sgs 3	27
6. samsung s3	219	19. samsung galaxy s3 lte	22
7. sgs3	187	20. sg3	21
8. siii	149	21. gsiii	16
9. samsung galaxy s iii	145	22. samsung galaxy s3 i9300	15
10. i9300	120	23. samsung i9300 galaxy s iii	13
11. gs3	82	24. s3 4g	11
12. galaxy siii	61	25. 3g s3	11
13. i9305	52	–	

not write a review report) after reading recommendations and discussions from other forum users. Understanding the reason behind helps a phone manufacturer in many aspects ranging from product design to marketing strategies.

In our study, we aim to recognize and normalize phone name mentions in Internet forums. Table 2 lists 4 example sentences, taken from a forum. Our task is to recognize the phone name mentions (highlighted in boldface) and link the recognized names to their corresponding formal names (shown in brackets). Note that, different products may follow very different naming conventions depending on the characteristics of the products (*e.g.,* camera lenses vs mobile phones). We choose mobile phone as the product of interest for two reasons. First, mobile phones, particulary smart phones, have a very large user base, leading to massive relevant discussions/comments in forums. Second, because of the strong competition in the market, mobile phones are released and updated at a very fast pace. Efficient and effective mining of user feedback about mobile phones is a major challenge for all phone makers.

In our problem setting, we assume two sets of inputs: (i) a collection of formal/official mobile phone names, and (ii) a collection of discussion threads from an Internet forum that are relevant to mobile phones. Note that, discussions in an Internet forum are usually organized into a few *discussion boards*; each discussion board consists of a number of threads and each thread has one or more post messages. Here, we assume the threads are collected from an Internet forum that is dedicated to mobile phone discussions like xda-developers forum[1] or from a specific discussion board that is about mobile phones, *e.g.,* the board titled "Mobile Communication Technology" in HardwareZone forum.[2]

Our proposed solution for mobile phone name recognition and normalization, named GREN, consists of three main components, namely, *candidate name Generator*, *CRF-based name REcognizer*, and *rule-based name Normalizer*.

[1] http://forum.xda-developers.com/

[2] http://forums.hardwarezone.com.sg/

Candidate Name Generator. As the name suggests, this component generates candidate names that might be used to mention a mobile phone. Examples are the names listed in Table 1. The generation consists of two phases. The first phase is to find name variations of phone brands including their common misspells and abbreviations (*e.g.,* bberry for BlackBerry). The next phase is to generate phone name variations by assuming that (i) a phone name is noun phrase, and (ii) a phone name shall either contain a brand variation or appear after a brand variation at least once. As the result of the second phase, we obtain a relatively large collection of candidate names. However, not all candidate names obtained are truly mobile phone names. For example, the word "battery" appears more than once after a brand (*e.g.,* Sony battery); then "battery" will be considered as a candidate name.

CRF-based Name Recognizer. The name recognizer is a classifier based on the linear chain CRF (Conditional Random Field) model. Given a sentence which contains at least one candidate name mention, the classifier predicts whether the mention indeed refers to a mobile phone or not. The prediction is based on the local context derived from the sentence and global context associated with the mentioned candidate name. To learn the classifier, we use three types of features including lexical features, grammatical features, and features derived from candidate names and candidate name mentions. We highlight that a large number of training instances are semi-automatically labeled in our implementation with minimal human annotation effort.

Rule-based Name Normalizer. The last component of GREN maps a recognized name variation to its formal name (*e.g.,* "ssg3" is mapped to "Samsung Galaxy SIII"). The normalization is mainly based on lexical rules. The confidence of a mapping is measured by the co-occurrence of the candidate name and the formal name in selected forum threads.

Although GREN is evaluated in the setting of mobile phone name recognition and normalization in forums, the proposed approach is generic. Candidate name generator is

Table 2 Mobile phone name mentions (highlighted in boldface), and their formal names [in brackets] in 4 example sentences

1.	True, **Desire** [HTC Desire] might be better if compared to **X10** [Sony Ericsson Xperia X10] but since I am using **HD2** [HTC HD2], it will be a little boring to use back HTC ...
2.	I just wanna know what problems do users face on the **OneX** [HTC One X]... of course I know that knowing the problems on **one x** [HTC One X] doesn't mean knowing the problems on **s3** [Samsung Galaxy SIII]
3.	Still prefer **ip 5** [Apple iPhone 5] then **note 2** [Samsung Galaxy Note II]...
4.	oh, the mono rich recording at **920** [Nokia Lumia 920] no better than stereo rich recording at **808** [Nokia 808 PureView].

probably the only component among the three that needs to be modified to reflect the naming conventions of products in a different domain. To summarize, we make the following contributions in this paper.

- We propose a novel solution named GREN for mobile phone mention recognition and normalization in Internet forums. Instead of running named entity recognizer on sentences directly, we generate candidate names and then predict whether a candidate name mention indeed refers to a mobile phone. The recognized names are then normalized to formal names.
- We propose a semi-automatic approach to generating training examples for CRF-classifier learning. A large number of training examples are obtained with little manual effort. More specifically, 33,072 sentences are obtained as training examples by manually labeling only 500 candidate names. A candidate name is a word or a phrase, which is easy to label.
- We conducted extensive experiments to evaluate the proposed GREN method and compare it against baseline methods. Our results show that GREN significantly outperforms all baseline methods. For mobile phone name recognition and normalization, with the best feature setting, GREN achieves precision and recall of 0.918 and 0.875 respectively.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 defines the research problem and presents an overview of GREN. The three components of GREN are detailed in Sections 4, 5, and 6 respectively. The experiments are reported in Section 7. After discussing limitations in Section 8, we conclude this paper in Section 9.

2 Related Work

Our work is related to both Named Entity Recognition (NER) and Named Entity Normalization (NEN). In general, NER detects name mentions in sentences, then NEN links a detected name to its formal name or the entity it refers to.

Named Entity Recognition. The task of NER is to extract and classify information units like person, organization, location, and other entity types [5, 14, 17, 21]. Relatively good recognition accuracy has been achieved for NER on formal text (*e.g.,* news article). For instance, F_1 of 0.908 is reported by LBJ-NER on CoNLL03 data [21]. Nevertheless, NER on user-generated content, particular short texts like Twitter messages, remains challenging. Next, we briefly review three work representing three different approaches to NER on tweets. Oliveira *et al.* [4] propose a FS-NER system which employs five lightweight filters to exploit nouns, terms, affixes, context and dictionaries for NER on tweets. Each filter employs a different recognition strategy and is independent of grammar rules. In their experiments, FS-NER outperforms CRF-based baseline by 3%. Liu *et al.* [12] combine K-Nearest Neighbors (KNN) and CRF for NER on tweets. KNN is first conducted to label tweets in word level; CRF is then utilized to recognize named entities, based on the results of KNN and gazetteers. Li *et al.* [8, 9] propose to segment the tweets in a batch and then recognize named entities from the segments by using part-of-speech tagger or a random walk algorithm. While the language usage in Internet forums is similar to that of tweets, the approaches proposed in aforementioned work are for named entities of all types (*e.g.,* person, location, organization). In our work, we aim to detect mobile phone names which follow certain naming conversions.

Named Entity Normalization. There are a large number of studies on normalizing and disambiguating various types of name mentions in formal text [2,3,7,16]. However, similar to NER, the performance of NEN is beset by the characteristics of user-generated content like tweets. Liu *et al.* [11] propose a collective inference method to link name mentions in tweets to the entries they refer to in Wikipedia. Specifically, they integrate three similarity functions into their method: similarity between a mention and an entity, similarity between entities, and similarity between mentions. Meij *et al.* [15] extract all possible n-grams from a tweet. A ranked list containing candidate Wikipedia entries is further built for each n-gram. The selection of the correct Wikipedia entry for linking is done by a supervised learning method. In our proposed solution, we normalize mobile phone name variations mainly based on lexical rules.

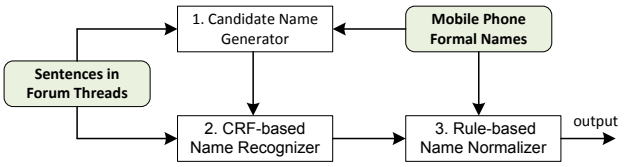


Fig. 1 Overview of GREN

Table 3 Notations and semantics

$\mathcal{T}$	a thread in a discussion board
f	a formal name of a mobile phone
$\mathcal{L}^f$	list of name variations for a formal name f
c	a candidate mobile phone name
C	collection of candidate names, $c \in C$
$\mathcal{W}_b$	word cluster containing a phone brand word
$\mathcal{W}_m$	word cluster containing a word used to name a phone model

Consumer PRODucts Contest. Wu *et al.* [24] achieve the best accuracy in the contest organized by ICDM 2012.[3] The task is to detect name mentions from web pages or forum messages and link the name mentions to product catalog. In their solution, name mentions are extracted by three models: (i) lexical match, (ii) rule based match, where the rules are handcrafted based on authors' observations and assumptions, and (iii) CRF-based model. For name normalization, all product names which contain a detected name mention are first retrieved. Words in the retrieved product names form a word set. A name mention is extended to a product name using words in this set and then linked to the product catalog. Different from [24] and other submissions to the contest, instead of recognizing product names directly from forum messages, we generate candidate phone names and then use CRF-based classifier to predict whether a candidate name mention is a mobile phone name. The generation of candidate phone names makes it feasible to obtain a large number of training examples with minimal manual labeling effort. In other words, our proposed solution is a semi-supervised method with no training examples provided directly, while the method reported in [24] relies on the large number of training examples provided by the contest organizers. Another key difference between our work and the solutions for this context is that we focus on mobile phones and the rules are crafted specifically based on the naming conventions for mobile phones. The rules listed in [24] are not designed specifically for mobile phones and are not directly applicable in our problem setting. As for the similarity, both [24] and our proposed solution are partially based on the CRF model, one of the most widely used models for named entity extraction.

3 Overview of the GREN method

In this section, we first define our research problem and then give an overview of the proposed GREN method.

Problem Definition. Let $\mathcal{T}$ be a thread in a discussion board relevant to mobile phones in an Internet forum. Let $S \in \mathcal{T}$ be a sentence from a post message in $\mathcal{T}$. Our task is to recognize from S a span of words $s = \langle w_1 w_2 \dots w_n \rangle$ $(n \geq 1)$ that refers to a mobile phone and map s to its formal name f. The collection of formal (or official) names of mobile phones is provided as an input to our task. The notations used in this paper are listed in Table 3.

Take the first sentence in Table 2 as an example, our task is to recognize mobile phone name mentions *Desire*, *X10*, and *HD2*, from the sentence and map these mentions to their formal names *HTC Desire*, *Sony Ericsson Xperia X10*, and *HTC HD2* respectively.

Method Overview. Figure 1 gives an overview of the proposed GREN method. Shown in the figure, candidate name generator takes two inputs: a collection of sentences and a set of formal names of mobile phones. The output of the candidate name generator includes mobile phone name variations like the examples listed in Table 1, and phrases that are wrongly taken as candidate phone names. Examples of the latter include *software*, *samsung service center*, and *iphone price*. We denote the set of candidate names by C. We will detail the candidate name generator in Section 4 and explain why some phrases are wrongly included as candidate names.

Given a sentence, if a span of words in the sentence matches any candidate name $c \in C$ (*i.e.,* a candidate name mention), then the name recognizer is utilized to predict whether the mention is truly a mobile phone name or not. The prediction is made by a CRF-based classifier with an array of features detailed in Section 5.

If a mention is predicted to be a true mobile phone name, then the last component in GREN, name normalizer, maps the phone name to its formal name. For example, *X10* is normalized to *Sony Ericsson Xperia X10*. With name normalization, one can easily retrieve all sentences mentioning a specific phone, regardless of the mention is through which name variation of this phone. The name normalization is presented in Section 6.

4 Candidate Name Generation

The formal name of a mobile phone usually contains two components: *brand* and *model name*. Some phones (about 22% in our dataset) have *model numbers* as the examples shown in Table 4. Note that a mobile phone may have more than one model number indicating small specification differences (*e.g.,* I9300 and I9305 for "Samsung Galaxy SIII"). A

[3] http://icdm2012.ua.ac.be/content/contest

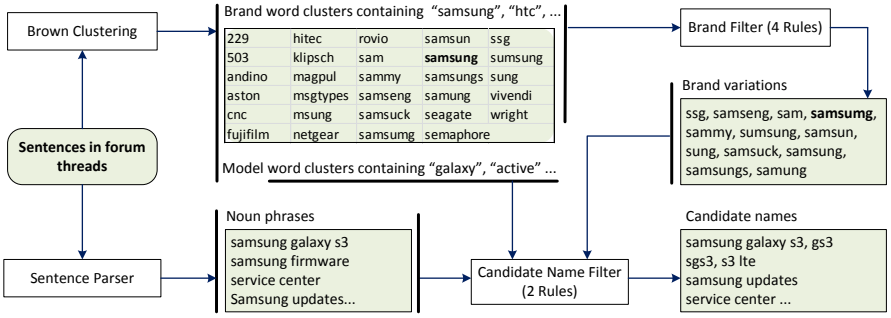


Fig. 2 Overview of candidate name generation using brand "Samsung" as an example

Table 4 Example mobile phone formal names

Brand	Model Name	Model Number
Samsung	Galaxy SIII	I9300, I9305
Nokia	Lumia 920	–
Nokia	8210	–
Sony	Xperia S	LT26

model number is often in the form of alphanumeric starting with a letter then followed by numerical digits, and is *independent* from a phone name. That is, a phone can be identified either by its *brand and model name* or by its *brand and model number*. For example, a model like "Samsung Galaxy SIII" can be identified without using its model number. Numbers like 920, 8210 shown in Table 4 are considered parts of model names but not model numbers. In the following discussion, we mainly focus on model names unless stated otherwise.

Figure 2 gives an overview of the candidate name generation process. A major challenge dealing with Internet forum data is the informal abbreviations and misspellings. To partially address this challenge, we adopt the Brown Clustering algorithm, a hierarchical clustering algorithm which groups the words with similar meaning and syntactical function together [1]. The intuition of the algorithm is that similar words have similar contexts in which they occur. The clustering is then conducted by maximizing the mutual information of the bi-gram language model [10]. Given all sentences from relevant threads, by applying the Brown clustering algorithm, we obtain a collection of clusters. Each word belongs to exactly one word cluster. To avoid misspellings and names used by very few users, we only consider the words that appear at least 10 times in the dataset when conducting Brown clustering[4]. Using the Brown clustering results, we obtain brand variations and then generate candidate phone names. Note that, the purpose of applying a clustering algorithm here is to group the words by their contexts of usage. That is, the clustering of words is based on their contextual or structural similarity derived from sentences. Although it is possible to apply other clustering algorithms like K-means or K-medoids for the same purpose, the computation of contextual similarity between two words is non-trivial.

[4] We consider a name variant is more meaningful and useful if the name has been used by a reasonable number of users, which is 10 in the entire paper. Given a name variant has at least 10 users, each word contained in this name variant must appear at least 10 times.

4.1 Brand Variation

Compared to the number of different phone models, the number of brands is much smaller. Because of their clear context in word usage, almost all variations of the same brand are grouped into the same word cluster through Brown clustering. An example word cluster containing brand "Samsung" is shown in Figure 2. The cluster has 29 words and many of them are spelling variations of "Samsung" and its nicknames (*e.g., sam* and *sammy*). If a word cluster contains a mobile phone brand, we call it a *brand word cluster*.

Because of the relatively good quality of word grouping for mobile phone brands, we apply 4 rules as *brand filter* to identify brand variations from a brand word cluster. Let $\mathcal{W}_b$ be the word cluster containing brand b. We consider a word $w \in \mathcal{W}_b$ is a brand variation of b if w satisfies one of the following 4 rules:

(i) The phonemic edit distance between w and b is 0. That is, the two words have the same pronunciation. The phonemic edit distance is measured by using Metaphone algorithm [18].
(ii) The first and the last characters in w and b are the same. The character comparison is case-insensitive.
(iii) The first three characters in w and b are the same. The character comparison is case-insensitive.
(iv) Brand b contains more than one upper-case character (*e.g.,* "BlackBerry") and the prefix of w matches all upper-case characters in b in sequence. The match is case-insensitive. For example "bb", "bbry", and "bberry" are variations of "BlackBerry" matched by this rule.

Although simple and even seem to be ad-hoc, the four rules give surprisingly good results. For brands with easy spelling (*e.g.,* Apple, HTC, and LG), no variations are found. More variations are obtained for brands with more than 5 characters (*e.g.,* BlackBerry, Motorola, Samsung, and Sony

Erricson). Figure 2 lists the 12 brand variations for "Samsung". Note that, for "Sony Erricson" the two words "Sony" and "Erricson" belong to two different word clusters. The brand variations are obtained by combining the matches from both clusters.

4.2 Candidate Mobile Phone Name

Recall that a mobile phone name contains *brand* and *model name*. A word cluster containing a model word may have higher chance containing model variations. However, many words that are used to name phone models are valid English words (*e.g.,* galaxy, active, one, desire), and some of them are widely used in very different contexts. Moreover, the same model word may appear in many different phone models. For instance, there are many different phone models from Samsung carrying the word "galaxy" in their formal names. Our clustering results show that there are 21 words in the cluster containing "galaxy" and among these 21 words 5 are model words (*i.e.,* word contained in at least one phone name). It is interesting to observe that the cluster containing the word "galaxies" has 87 words, among which 48 are model words. Many users have used the word "galaxies" to refer to the series of Samsung phones when compared with other phone models. In this sense, a rule based approach to recognizing model variations becomes less practical.

Based on the assumption that a mobile phone name is a noun phrase, we filter noun phrases to obtain candidate phone names. Let $\mathcal{W}_m$ be a *model word cluster* which contains at least one word used to name a phone model. A noun phrase is a candidate mobile phone name if it satisfies both conditions:

(i) The phrase contains a brand variation, or the phrase appears after a brand variation at least once in the whole dataset; and
(ii) At least one word in the phrase appears in a model word cluster, and all the remaining words appear in either model word clusters or brand word clusters.

Because of the first condition, only the sentences that contain at least one brand variation need to be parsed.

Shown in Figure 2, with the two rules as candidate name filter, we capture most phone name variations such as "sumsang galaxy s3" and the other names listed in Table 1. However, due to the noisy word usage contexts of model names like "one" and "active", the collection of candidate names contains a lot of noise. For example, "service center" and "samsung updates" are listed as candidate names because they satisfy both conditions.

5 CRF-Based Name Recognition

We now have a set of candidate names that might be used to refer to mobile phones in forums. Given a sentence mentioning at least one candidate name, our task is to predict whether this mention indeed refers to a specific phone model, *i.e.,* a binary classification task. Many classifiers have been developed in past decades and been used in text domain, *e.g.,* Naïve Bayes, K Nearest Neighbor, Support Vector Machines, etc. [23]. However, most of these classifiers cannot utilize the local contextual information in a sentence that is useful for the prediction in our task. In this study, we adopt linear-chain CRF model. CRF is a discriminative model that has been widely used in sequence labeling tasks, particularly named entity recognition [6, 19]. Next, we detail how to obtain a large number of training examples in a semi-automatic manner and the features used in building the CRF classifier.

Training Examples. Obtaining a reasonable number of high quality training examples is always a major challenge in training a classifier. In most classification tasks, training examples are manually annotated, which is costly and time consuming. We propose to generate training examples for our CRF model in a semi-automatic manner with minimal manual labeling effort.

Before we create labeled sentences for CRF training, we create two sets of mobile phone names: a set of positive names ($\mathcal{P}$) and a set of negative names ($\mathcal{N}$). In our problem definition (see Section 3), a set of formal names is given as one input. Each formal name (brand and model name, without model number) is inserted into $\mathcal{P}$. If a formal name contains Roman number like "ii" and "iii", a separate copy of this formal name is inserted into $\mathcal{P}$ by mapping the Roman number into Arabic number. To further enlarge the list of positive names, if a model name has more than one word (*e.g.,* "galaxy note"), then the model name is inserted into $\mathcal{P}$. For example, given a formal name "Samsung Galaxy SIII", the following four names are considered positive names and are inserted into $\mathcal{P}$: "Samsung Galaxy SIII", "Samsung Galaxy S3", "Galaxy SIII", and "Galaxy S3". Note that, the key consideration here is to ensure the high quality or correctness of the selected examples. Because many mobile phone names contain valid English names (*e.g.,* desire, butterfly, active, one), mentioning of any one of such words does not mean that the sentence contain a positive phone name mention. Considering model names with at least two words (*e.g.,* galaxy S5) addresses this issue.

Populating the set of negative names $\mathcal{N}$, however, is through manual annotation. In Section 4, we get the set of candidate names C, and we note that many of its entries are not truly mobile phone names. A number of entries from C that can be easily identified not phone names are manually selected to form $\mathcal{N}$. Example entries in $\mathcal{N}$ include "debug", "service center", "retail prices", "toolbar", "firmware", and "update". We argue that annotating words/phrases that are not phone names is much easier and less time-consuming compared to annotating sentences directly.

Table 5 Original and rewritten sentences with labels for "ip_5"

Original sentence:	Still	prefer	ip 5	then	note 2
Rewritten sentence:	Still	prefer	ip_5	then	note_2
"ip_5" (w_i):	w_{i-2}	w_{i-1}	w_i	w_{i+1}	w_{i+2}

With sets $\mathcal{P}$ and $\mathcal{N}$, we select sentences satisfying the following two conditions to be training examples: (i) The sentence contains at least one entity in either set $\mathcal{P}$ or set $\mathcal{N}$; and (ii) The sentence does not contain any entry that appears in C but not in $\mathcal{N}$ or $\mathcal{P}$ (*i.e.,* any entry in $C \setminus (\mathcal{P} \cup \mathcal{N})$). The second condition is set because the entries in $C \setminus (\mathcal{P} \cup \mathcal{N})$ are not manually annotated.

Features. Because our task is to predict whether a candidate name mention is truly a mobile phone name, we consider each candidate name as "*a single token*" in the sentence. More specifically, we rewrite the sentence by adding underscores to the words contained in a candidate name, as shown in Table 5. The rewriting enables us to take surrounding words of a candidate name to be its context in a more natural manner.

In the example shown in Table 5, after rewriting, candidate names *ip_5* and *note_2* each becomes one word. We propose to use 6 features in our CRF model (listed in Table 6), mostly considering the current word w_i, and its surrounding two words on both sides: $w_{i-2}w_{i-1}$ to its left-hand side and $w_{i+1}w_{i+2}$ to its right-hand side.

The two lexical features include the word and the surrounding words from w_{i-2} to w_{i+2} (L_1) and the surface forms of the words (L_2). The two grammatical features include the POS taggers of the 5 words from w_{i-2} to w_{i+2} (G_1) and the path prefixes obtained from Brown clustering (G_2). Note that, the path prefixes of the words are obtained by running Brown clustering on the rewritten sentences. The resultant word clusters therefore are different from the word clusters obtained for candidate name generation in Section 4. We argue that by rewriting each candidate name mention to a single word, word usage context patterns, particularly the usage patterns surrounding candidate names, can be better captured by Brown clustering algorithm on the rewritten sentences than the original sentences.

For the candidate name features, the first feature (N_1) is to flag the candidate names. The corresponding feature is assigned a value of 1 if the word is rewritten from a candidate name (*e.g., ip_5*) and assigned 0 otherwise. The last feature (N_2) is the *brand entropy* of a candidate name. Recall that in candidate name generation (Section 4.2), a candidate name either contains a brand variation or appears after a brand variation at least once. By intuition, a mobile phone name should appear after only one brand (or its variations) but not appear after many different brands. If a candidate name appears after many different brands, then it is unlikely a true phone name. The brand entropy feature is computed as following:

- If a word in a sentence is not a candidate name, then feature N_2 is set to 0.
- If a word is a candidate name (*i.e.,* the rewritten form) that contains a brand variation, then feature N_2 is set to 1.
- If a word is a candidate name that does not contain a brand variation, then we compute the probability of this candidate name appears after a brand (or its variation). Let m be the number of times a candidate name c appears after any brand variation, and m_b be the number of times c appears after a brand variation of brand b. Then the probability of c that follows brand b is $P(c, b) = \frac{m_b}{m}$. Let B be the set of brands that c appears after at least once, then the brand entropy of c is $H(c) = -\sum_{b \in B} P(c, b) \log P(c, b)$. If a candidate name, for example "galaxy siii", appears only after one brand "Samsung", then $H(c) = 0$. If a candidate name (*e.g.,* "software", "phone", or "battery") appears after many different brands, then $H(c)$ is a much larger value. The value of feature N_2 is set to 1 if $0 \leq H(c) \leq 1$ and set to 2 if $H(c) > 1$.

Sentence Labeling. CRF model is a very flexible model, allowing different kinds of labeling schemes. In our proposed solution, we use "YNO" (Yes-No-Out) scheme. Given a rewritten sentence as a training example, if a word is rewritten from a positive name in $\mathcal{P}$, then the word is labeled "Y"; if a word is rewritten from a negative name in $\mathcal{N}$, the word is labeled "N". The remaining words in the sentence are labeled "O".

6 Rule-based Name Normalization

After we have the CRF classifier, if a candidate name is predicted to be a phone name variation, our next task is to map this name to its formal name. For example, all the 25 variations listed in Table 1 shall be mapped to "Samsung Galaxy SIII". We adopt a rule-based approach to creating the mapping.

Because of the way we train our CRF classifier, most phone name variations detected are originated from the candidate name set C. That means we can pre-normalize candidate names in C to their corresponding formal names with some degree of noise because not all candidate names are true mobile phone names. For each formal name f, we build a list of its possible variations from C with confidence scores, denoted by $\mathcal{L}^f$. A candidate name in list $\mathcal{L}^f$ is normalized to formal name f if detected to be true by the CRF classifier.

Given a formal name f containing brand f_b, model name f_m, and model number f_r, we select its name variations from C in two steps, shown in Algorithm 1.

Step 1 (Lines 2 - 6): Given a candidate name c, if all its characters are contained in the brand and model name of

Table 6 Feature description for lexical features (L_1, L_2), grammatical features (G_1, G_2), and name features (N_1, N_2)

L_1	The current word and its surrounding two words $w_{i-2}w_{i-1}w_iw_{i+1}w_{i+2}$, and their lowercased forms.
L_2	Word surface feature of the current word: Initial capitalization, all capitalization, containing capitalized letters, all digits, containing digits and letters.
G_1	POS tagger of the current word and its surrounding two words.
G_2	Path prefixes of length 4, 6, 10, 20 (*i.e.,* maximum length) of the current word by Brown clustering.
N_1	Flags to indicate whether the current word and its surrounding two words are candidate phone names
N_2	The brand entropy of the current word and its surrounding two words.

Algorithm 1: Name variation generation for formal name f

Input: f, a formal name consisting of brand f_b, model name f_m, and model number f_r;
C, set of candidate names
Output: $\mathcal{L}^f$, list of name variations of f

1 $\mathcal{L}^f \leftarrow \Phi$;
2 **forall the** $c \in C$ **do**
3 **if** *sequenceContainment*(f_b, f_m, c) **then**
4 $\mathcal{L}^f \leftarrow \mathcal{L}^f \cup \{c\}$;
5 **if** *containsModelNumber*(c, f_r) **then**
6 $\mathcal{L}^f \leftarrow \mathcal{L}^f \cup \{c\}$;
7 **forall the** $f' \in \mathcal{L}^f$ **do**
8 **forall the** $c \in C \setminus \mathcal{L}^f$ **do**
9 **if** *wordMatch*(c, f') **then**
10 $\mathcal{L}^f \leftarrow \mathcal{L}^f \cup \{c\}$;
11 **forall the** $f' \in \mathcal{L}^f$ **do**
12 $f'.conf \leftarrow$confidenceScore(f', f);
13 **if** $f'.conf == 0$ **then**
14 remove f' from $\mathcal{L}^f$;
15 **return** $\mathcal{L}^f$;

the formal name, and are arranged in the same sequence, then we consider c as a name variation of f (Lines 3 and 4). For example, all characters in candidate name "SGS III" are contained in "Samsung Galaxy SIII" and are in the same sequence. The match is case-insensitive, and Roman numbers match Arabic numbers. As the result, *sequenceContainment* function returns true for "s3", "galaxy s3", "sgs3", "siii", etc. If a candidate name c contains the model number of f, then the candidate name is also added to $\mathcal{L}^f$, leading to the addition of "i9300", "i9305", "s3 i9300", and "samsung i9300 galaxy s iii" as examples. If a phone formal name does not contain a model number, then the *containsModelNumber* function (Lines 5 and 6) is skipped.

Step 2 (Lines 7 - 10): From step 1, we get the initial set of name variations in $\mathcal{L}^f$. Next, we try to learn from this set of name variations on how users name this phone. For this purpose, we tokenize all names currently in $\mathcal{L}^f$ to get all words people use to refer to this phone. The brand variations and single-character words are ignored. Example words obtained include "s3", "galaxy", and "sgs". Then, if any word in a candidate name that is currently not in $\mathcal{L}^f$ matches one of these words, then the candidate name is added to $\mathcal{L}^f$ (Lines 7 - 10). With this step, we get candidate names like "s3 lte", "s3 phone", "s3 4g", and "s3 pebble blue".

Through the above two steps, each formal name f is now associated with a list of candidate names stored in $\mathcal{L}^f$. However, one candidate name may appear in multiple $\mathcal{L}^f$'s. For example, "SGS II" appears in the list of "Samsung Galaxy SII" and the list of "Samsung Galaxy SIII", because the function *sequenceContainment* returns true for both formal names. Then which is the correct normalization? To address this issue, we compute a confidence score.

Confidence Score. An Internet forum contains a large number of threads. Each thread has a title, a description (*i.e.,* the first post message), and the replying posts. Recall that in Section 5, to train the CRF classifier, for each formal name, we generate a few positive names (*e.g.,* "Samsung Galaxy SIII" leads to the generation of "Samsung Galaxy SIII", "Samsung Galaxy S3", "Galaxy SIII", and "Galaxy S3"). To compute the confidence score, for each formal name, we obtain all the threads such that the title and description of the thread contain only the formal name, or any positive names derived from it, but not any other candidate names. We assume such thread has clear topical focus on the phone with the given formal name. Accordingly, when users post messages in this thread, name variations of this phone have higher chance to surface. We count the frequency of candidate names that appear in all these threads obtained for a formal name. The confidence score of a candidate name is the relative frequency with respect to the most frequent candidate names in these threads (Lines 11 and 12). If the confidence score of a candidate name is 0, this means that the candidate name never appears in any discussion threads specific to the phone model in the whole forum, then the candidate name is removed from $\mathcal{L}^f$ (Lines 13 - 14).

After computing the confidence score, if a candidate name appears in multiple $\mathcal{L}^f$'s, only the instance with the highest confidence score is kept in the corresponding $\mathcal{L}^f$. For example, "SGS II" has the highest confidence score with

"Samsung Galaxy SII"; it is removed from the list for "Samsung Galaxy SIII".

Remark. The candidate names that are assigned to a formal name might not be a true mobile phone name variation because of the noise in candidate name generation. Only the names that are predicted true by the CRF classifier will trigger the normalization rule. On the other hand, the normalization rule of a phone may not cover all name variations of the phone. For example, some users use codename "Nozomi" to refer the phone "Sony Xperia S", but the codename "Nozomi" is not included in $\mathcal{L}^f$ for this phone. In practical applications, manually crafted rules may be added to the phone's normalization list to capture such cases.

7 Experiment

We conducted extensive experiments to evaluate the proposed GREN method and compared it with baseline methods. We further study the results obtained by the different components in the GREN method.

7.1 Dataset

Forum Data. We collected forum data from the discussion board titled "Mobile Communication Technology" in HardwareZone forum, probably the most popular forum in Singapore.[5] Most discussions in this board are about mobile phones, and a few user interest groups are formed for a few brands: BlackBerry, Nokia, Samsung, Motorola, Sony, LG, HTC, and Apple's iPhone. In total, 25,251 discussion threads were collected, containing 1,026,190 post messages. The time duration of the discussion is from March 15, 2002 to May 2, 2013 in our data collection.

Mobile Phone Formal Names. The formal names of mobile phones were crawled from GSMArena.com.[6] GSMArena lists all phone models released by all phone makers in the global market. However, not all phone markers release their products in Singapore. Hence, we only consider the mobile phone models from the 8 brands where user interest groups are found in HarewareZone: BlackBerry, Nokia, Samsung, Motorola, Sony (Sony Ericsson), LG, HTC, and Apple's iPhone. For these 8 brands, 2,623 formal mobile phone names were obtained from GSMArena. Again, only a subset of these 2,623 phone models were released in Singapore, and those released before March 2002 are not covered by our dataset.

Ground Truth Labeling. To evaluate the accuracy of name recognition and normalization, we select 20 mobile phones (the first column in Table 7) which are relatively popular in the forum. For each phone, we randomly selected a thread with about 100 post messages, then we manually labeled the mobile phone name mentions in these posts and mapped to their formal names. In our manual annotation process, we observed that some of the names were used to refer to a series of phones. For example, in this sentence "I still prefer iphone than any android phones", "iphone" here does not refer to any specific model (*e.g.,* iPhone 4s or iPhone 5). That means we cannot even manually normalize this mention to any formal name, and thus we ignore such cases in our evaluation. Names like "s3 lte" and "3g s3" were labeled as positive names because each name indeed refers to a specific phone model with emphasis on specification variants.

In total, we labeled 4,121 sentences within which there were 946 phone name mentions. The 946 name mentions include name variations for the 20 selected mobile phones as well as name variations of other phone models, because users often compare phones with many different models in their discussion. The number of distinct phone name variations for each of the 20 phones is listed in the second column in Table 7.

7.2 Implementation and Intermediate Results

Before we jump to the results for name recognition and normalization, we report the software packages used in GREN implementation and the intermediate results obtained, *e.g.,* the number of obtained candidate names and the number of sentences used for training the CRF classifier.

Candidate Name Generation. For pre-processing, we adopted Stanford Tokenizer[7] for the sentence splitting task. Brown Clustering was conducted using Liang's code.[8] The number of clusters was set to 1000, ignoring the words that appeared fewer than 10 times in the dataset. On average, each resultant word cluster contains 32 words. After obtaining the brand variations for the 8 brands (see Table 8), we generated the candidate phone names. The sentences were parsed by using Stanford Parser with careless English model.[9]

In total, 4,258 candidate names were obtained, *i.e.,* $|C| = 4,258$. Each of the 4,258 candidate names has been used by at least 10 users. We consider a candidate name to be less interesting if it is used by very few users only. Many such candidate names are results of typos.

It is infeasible to directly evaluate the recall of C unless all the 1,026,190 post messages are fully manually labeled. We therefore partially evaluate the recall of C using the phone variations obtained from the 20 manually labeled phones. In total, we have 86 distinct name variations for the 20 selected

[5] http://forums.hardwarezone.com.sg/

[6] http://www.gsmarena.com/

[7] http://nlp.stanford.edu/software/tokenizer.shtml

[8] https://github.com/percyliang/brown-cluster

[9] http://nlp.stanford.edu/downloads/lex-parser.shtml

Table 7 The 20 mobile phones, the number of distinct name variations in threads ($\mathcal{T}$), and the number of distinct name variations covered in the normalization list ($\mathcal{L}^f$); Columns titled "Re ($\mathcal{L}^f$)" and "AP ($\mathcal{L}^f$)" report the recall and average precision of $\mathcal{L}^f$ respectively; The last column "AP($\mathcal{L}^f_r$)" reports the average precision of $\mathcal{L}^f_r$ after removing negative entries from $\mathcal{L}^f$ by the name recognizer.

Mobile Phone	$\mathcal{T}$	$\mathcal{L}^f$	Re($\mathcal{L}^f$)	AP($\mathcal{L}^f$)	AP($\mathcal{L}^f_r$)
apple iphone 4	3	3	1.000	0.643	1.000
apple iphone 4s	3	2	0.667	0.284	0.500
apple iphone 5	7	3	0.429	0.742	0.600
blackberry bold 9650	4	2	0.500	0.600	0.500
blackberry bold 9700	7	5	0.714	0.692	0.800
blackberry z10	4	3	0.750	0.800	1.000
htc desire hd	2	2	1.000	0.833	1.000
htc hd2	2	2	1.000	0.750	1.000
htc one x	4	3	0.750	0.549	1.000
lg nexus 4	3	3	1.000	0.700	1.000
motorola milestone 2	7	3	0.429	1.000	1.000
nokia e72	1	1	1.000	1.000	1.000
nokia lumia 900	5	4	0.800	0.714	1.000
nokia lumia 920	6	5	0.833	0.778	1.000
samsung galaxy note ii	8	7	0.875	0.666	1.000
samsung galaxy s iii	9	8	0.889	0.840	0.800
samsung omnia ii	4	3	0.750	0.875	1.000
sony ericsson xperia arc	2	2	1.000	0.361	1.000
sony xperia s	2	2	1.000	0.214	0.667
sony xperia z	3	3	1.000	0.507	1.000
Average	4.3	3.3	0.819	0.677	0.893

Table 8 Brand variations for 8 brands

Brand	**Brand variation**
Apple, HTC, LG	–No brand variations–
Nokia	nokia, nokie, nk
BlackBerry	blackberry, bbry, blackbery, bb, bberry
Motorola	motorola, moto, motorolla, mot
Samsung	ssg, samseng, sam, samsumg, sammy, sumsung, samsun, sung, samsuck, samsung, samsungs, samung
Sony Erricson	sony erricson, sony ericsson, sony ericson, sony ericcson, sonyericsson, sony ericssion, sn, sony, sonyeric

Table 9 The number of distinct name variations in the 20 manually labeled threads $\mathcal{T}$'s and the candidate name set C, with user frequency of 10 and above

Mobile Phone	No. in $\mathcal{T}$'s	No. in C	Recall
The 20 selected phones	74	67	0.905
Other phones mentioned	93	85	0.914
All phones in the 20 threads	167	152	0.910

phones, listed in the second column in Table 7. Among them, there are 74 name variations each has been used by at least 10 users in the whole dataset. Because C only includes candidate names with user frequency equal or greater than 10, we compute the recall to be the ratio of these 74 name variations that are also included in C, which is 0.905. Because users often compare different phones, we also get another 93 phone name variations (not for the 20 selected phones), each of which has user frequency not lower than 10. The recall for these 93 phone name variations is 0.914, computed in a similar manner. Table 9 summarizes the results. Overall, the recall of C, evaluated using the 167 distinct phone name variations is 0.910, which is a very high value.

CRF-based Name Recognition. The CRF classifier for name recognition was implemented using the CRF++.[10]

Next, we select the positive phone names and negative candidate names to generate training sentences. Using the 2,623 formal mobile phone names from GSMArena, we identified 412 positive names from the candidate name set C. We then randomly selected 500 negative names from the remaining 3,846 names in C, which were clearly not phone names. Following the rules stated in Section 5, we first selected 21,246 sentences each of which contained at least one positive name mention. In total, there were 22,884 positive phone name mentions. There were 250,644 sentences satisfy the condition with negative name mentions. However, using all these sentences leads to significant imbalance between positive and negative examples. We therefore randomly selected a subset of 11,826 sentences such that the total number of negative name mentions were the same as positive name mentions. In total, 33,072 sentences were used to construct the CRF-based name recognizer. Note that, in the process of

[10] https://code.google.com/p/crfpp/

generating the 33,072 training sentences, only the 500 negative names were selected manually from C with very low cost of manual annotation.

Regarding the CRF features (see Table 6), the POS tagger information was extracted by Stanford POS Tagger with careless English model[11], and the Brown clustering was again conducted by using Liang's code on the rewritten sentences. The performance of the CRF-based name recognizer will be reported in Section 7.3 in comparison with other baseline methods.

Rule-based Name Normalization. For each formal name f, we have a list of candidate names $\mathcal{L}^f$. The candidate names in $\mathcal{L}^f$ are originated from C and will be normalized to f if they are recognized to be true names by the CRF classifier (see Section 6). Now, we evaluate the precision and recall of $\mathcal{L}^f$ using the labeled data for the 20 selected phones.

In Table 7, the second column lists the number of distinct name variations for each of the 20 phones in the labeled data. Column 3 lists the number of distinct name variations that are included in the list $\mathcal{L}^f$ for each phone. The recall is computed by taking the ratio of the two numbers for each phone, reported in column 4. Observe that many phones get 100% recall, and the overall recall average is 0.819. Note that this value is lower than the recall of C for the 20 phones (*i.e.,* 0.905) reported in Table 9. One reason is that the computation of the recall of C only uses name variations with user frequency not smaller than 10. In Table 7, among the 86 distinct name variations for the 20 phones, 74 phone name variants have user frequency equal or greater than 10. Because all candidate names from $\mathcal{L}^f$ are originated from C, then the 12 phone name variations with user frequency below 10 degrades the recall value of $\mathcal{L}^f$ on average.

Next, we evaluate the average precision of list $\mathcal{L}^f$. Recall that the candidate names are ranked by confidence scores in $\mathcal{L}^f$. Using the ground truth for the 20 phones, we compute the average precision of each list and report the value in Table 7, column titled "AP($\mathcal{L}^f$)". Average precision of a ranked list of items is computed by averaging all precision values obtained at each rank position of the list from the top-ranked item to the bottom-ranked item [13]. The mean average precision is 0.677 for the 20 phones. We note that, the list $\mathcal{L}^f$ is generated from C, and some entries in $\mathcal{L}^f$ may not be recognized as true phone name variations by the CRF classifier. If we remove the entries that are always predicted negative by the CRF classifier, then the mean average precision becomes 0.893, reported in the last column titled "AP($\mathcal{L}^f_r$)" in Table 7. Particularly, 14 out of the 20 mobile phones have perfect average precision of 1.0 for $\mathcal{L}^f_r$.

Next, we report the accuracy of name recognition and normalization of GREN and compare it with baseline methods.

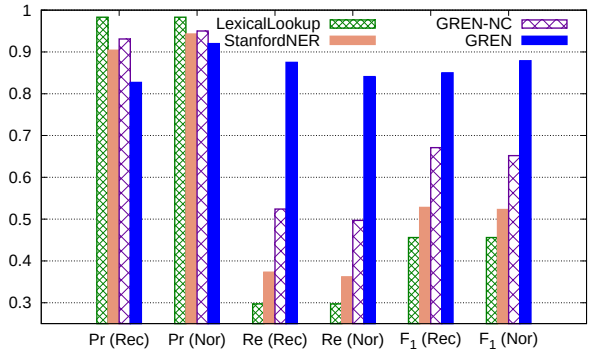
[11] http://nlp.stanford.edu/downloads/tagger.shtml

7.3 Name Recognition and Normalization

Methods. We compare the proposed GREN method with other 3 baselines: GREN-NC, StanfordNER, and LexicalLookup. In the following, we detail the four methods.

- **GREN**: This is our proposed method with candidate name generation, CRF-based name recognition, and rule-based normalization. The training sentences were obtained by using 412 positive phone names and 500 negative phone names selected from the set of candidate names C reported in Section 7.2. The features used to train the CRF model are listed in Table 6. Each candidate name mention was rewritten to a single word in both training and test sentences. We used YNO (Yes-No-Out) labeling scheme in CRF training: "Y" for a word rewritten from a positive name mention, "N" for a word rewritten from a negative name mention, and "O" for other words in the sentence (see Section 5).
- **GREN-NC**. In GREN, every candidate name mention is identified and rewritten to a single word. To evaluate the effectiveness of the pre-identification of candidate names, we propose GREN-NC method. In this method, no candidate name mention is identified, and the sentences are kept in their original form. In training CRF, we used BILO (Begin-In-Last-Out) scheme. It is reported that BILOU (Begin-In-Last-Out-Unit) outperforms BIO (Begin-In-Out) scheme in named entity recognition [20, 21]. However, we cannot adopt the BILOU scheme because of the automatically generated training sentences. Recall in Section 5, the positive phone names are selected such that the model name (without brand) must contain at least two words. For this reason, no words in any sentences will be assigned label U (Unit). Therefore, we used BILO (Begin-In-Last-Out) scheme without the U label. The same set of features in Table 6 were used except the last two features N_1 and N_2, because without sentence rewriting, a word by itself cannot represent a candidate name. In short, GREN-NC was trained using the same set of sentences as GREN with similar set of features except that GREN-NC did not take candidate names as an input.
- **StanfordNER**. This is one of the most widely used package for named entity recognition. We adopted the default features provided by the package[12] and trained the CRF classifier using the same set of labeled sentences as in GREN and GREN-NC. Same as GREN-NC, the BILO labeling scheme was used. Note that, StanfordNER allows to use a gazetteer as external knowledge. However, the candidate name set C cannot be used as a gazetteer because many of its entries are not true mobile phone names. The

[12] http://nlp.stanford.edu/nlp/javadoc/javanlp/edu/stanford/nlp/ie/crf/CRFClassifier.html



(a) Pr(Rec) and Pr(Nor) denote precision for name recognition and name normalization, respectively; the same applies to Re and F_1

Method	Name recognition			Name normalization		
	Pr	Re	F_1	Pr	Re	F_1
LexicalLookup	**0.983**	0.297	0.456	**0.983**	0.297	0.456
StanfordNER	0.904	0.373	0.528	0.943	0.362	0.523
GREN-NC	0.931	0.524	0.671	0.950	0.497	0.652
GREN	0.827	**0.875**	**0.850**	0.920	**0.841**	**0.879**

(b) Pr, Re, and F_1 of the 4 methods with best results in boldface

Fig. 3 Pr, Re, and F_1 of name recognition and name normalization for the 4 methods

only difference between StanfordNER and GREN-NC is the features used for model training.

– **LexicalLookup**. We stored the 412 formal names that were used as positive names for sentence labeling in a dictionary. Phone name mentions were recognized by lexical lookup in this dictionary. Normalization is not necessary here because the names in the dictionary are formal names.

Evaluation Metric. We used the widely adopted *Precision* (Pr), *Recall* (Re), and F_1 to evaluate the accuracy of name recognition and the accuracy of name normalization respectively. The three measures for the name recognition subtask are defined as follows:

– Precision is the ratio of true phone name mentions among all mentions that are predicted positively.
– Recall is the ratio of correctly recognized name mentions among all phone name mentions annotated in the ground truth data.
– F_1 is the harmonic mean of precision and recall.

For the name normalization task, a phone name mention is correctly normalized only if it is first correctly detected by the name recognizer and then correctly mapped to its formal name by the name normalizer.

Experimental Results. The precision, recall, and F_1 of the four methods for name recognition and name normalization are plotted in Figure 3(a) and listed in 3(b) for easy comparison.

We discuss the results on name recognition first. Observe from Figure 3, all the 4 methods achieve very high precision in name recognition. LexicalLookup gets nearly perfect precision as expected because that all matched name mentions are formal names.[13] Both GREN-NC and StanfordNER achieve precision value higher than 0.9. Although the proposed GREN method has the lowest precision value of 0.827 among all methods, GREN significantly outperforms the other methods in recall. More specifically, GREN achieves recall of 0.875, which is 67% of improvement over the second best recall by GREN-NC, 134% improvement over StanfordNER, and nearly 200% improvement over LexicalLookup. Because of the much higher recall, GREN achieves F_1 of 0.850, followed by GREN-NC of 0.671 and StanfordNER of 0.528.

From this set of results, we argue that both candidate name generation and sentence rewriting are main factors contributing to the significant improvement of F_1 for GREN. Recall that the positive names used to generate training sentences are either formal phone names or model names (without brand) that contain at least two words. As the result, the name mentions seen by StanfordNER and GREN-NC in the training data do not contain any informal abbreviations or misspells. It becomes reasonable that StanfordNER and GREN-NC achieve better precision than GREN but much poorer recall. The better recall by GREN-NC over StanfordNER attributes to the Brown clustering feature (G_2 in Table 6) which has been reported effective in NER from user-generated content like tweets [21, 22]. With candidate name generation and sentence rewriting, GREN is able to "focus" more on the surrounding context of a candidate name mention, and at the same time to ignore the detection of candidate name boundaries because of the YNO labeling scheme. Nevertheless, not all candidate names are true phone name variations. The noises in candidate names result in slightly poorer precision for GREN, compared with StanfordNER and GREN-NC.

We now discuss the results on name normalization. There is no change in results of LexicalLookup because the names recognized are formal names, according to the method definition. For the rest three methods (*i.e.,* StanfordNER, GREN-NC, and GREN), improvement in precision and degradation in recall are observed compared to their corresponding results on name recognition. In our evaluation for name normalization, a positive instance refers to a name mention that is correctly recognized and correctly normalized to its formal name. Many of the mentions that are wrongly recognized as phone names by the name recognizer cannot be normalized to any formal names using the normalization rules. Thus, they are now considered negative instances. On the one hand, the normalization process removes errors from the results of name recognition, leading to the increase in precision. On the other hand, due to the incompleteness of the rules, some

[13] The precision is not 1.0 because of a special case involving a space character. LexicalLookup recognizes "HTC Touch Diamond" from sentences containing word span "HTC Touch Diamond 2", and the correct recognition of the latter should be "HTC Touch Diamond2" (the formal name). "HTC Touch Diamond" is a different product.

Table 10 The name normalization accuracy after removing one or two features. Best results are highlighted in boldface

Method/Feature	Pr	Re	F_1
GREN	0.920	0.841	0.879
GREN-L_1	0.916	0.836	0.874
GREN-L_2	0.918	**0.875**	**0.896**
GREN-G_1	0.920	0.839	0.877
GREN-G_2	0.899	0.728	0.804
GREN-N_1, N_2	**0.961**	0.791	0.868
GREN-NC	0.950	0.497	0.652
GREN-NC-L_2	0.953	0.447	0.609

of the correctly recognized names cannot be normalized to their corresponding formal names, *e.g.,* "Nozomi" is not normalized to "Sony Xperia S", leading to degradation in recall. Considering both precision and recall, GREN increases its F_1 from 0.850 to 0.879 after name normalization. Slightly worse F_1's are observed for both GREN-NC and StanfordNER. Nevertheless, in real applications, more rules may be added to address the special cases in name normalization.

Feature Analysis. We now study the impact of the features listed in Table 6. Table 10 reports the precision, recall, and F_1 of name normalization using all features in GREN and after removing one or two features (*e.g.,*-L_1, -N_1, N_2). Surprisingly, we observe that much better F_1 is achieved by removing the word surface feature L_2 from GREN (*i.e.,* GREN-L_2). Removing L_2 feature leads to the improvement of F_1 from 0.879 to 0.896, about 2%. For comparison, we include results of GREN-NC and GREN-NC-L_2 in the last two rows in the table. Observe that removing L_2 from GREN-NC adversely affects F_1 with a big drop in recall. While word surface features are effective in most NER tasks, our results suggest that such features derived from the artificially created words (*i.e.,* the single words rewritten from candidate names in GREN) confuse the CRF model. Among the other features, features from Brown clustering are the most effective features; removing G_2 results in the largest degradation in F_1 comparing GREN-G_2 against GREN. The two features derived from the rewritten candidate names (N_1 and N_2) are the second most effective features. Removing L_1 or G_2 each leads to marginal degradation in F_1 measure. Relatively, they are less effective compared to the features from Brown clustering and the features from the rewritten candidate names.

8 Limitations

In our approach, Brown clustering is considered in the first step. Even though satisfied performance is obtained, Brown clustering only works on existing data. Due to this reason, the proposed approach is limited to the existing mobile phone names, while variations of newly released phones are not to be recognized. Thus, a re-generation process of candidate names is necessary. On the other hand, as a series of mobile phones follow similar naming conventions, some rules may be designed to capture the patterns which partially solve the problem.

As the normalization component of GREN is mainly based on lexical rules, it may not cover all name variations of a mobile phone. As mentioned earlier, some users use codename "Nozomi" to refer the phone "Sony Xperia S", but the codename "Nozomi" is not included in $\mathcal{L}^f$ for this phone. Adding additional rules to cover such cases is one possible solution, but it requires lots of manual efforts. A possible improvement to the normalization component is to consider not only lexical rules, but also additional information such as the profiles of mobile phones obtained from Wikipedia and the websites that provide specific information of mobile phones. The similarity between the context of a name variant and the mobile phone profile may help in the normalization.

Shown in the experimental results of recognition and normalization of mobile phone names, normalization improves the results generated by recognition in terms of precision. However, as recognition and normalization are conducted separately, there is no feedback from the normalization process to the recognition process, even though the normalization process realizes that a name mention is not a mobile phone name variation. A possible direction of future research is to conduct recognition and normalization jointly, such that the feedback from each component helps to improve the other component.

9 Conclusion

In this paper, we propose a novel method for mobile phone name recognition and normalization in Internet forums. Different from most NER approaches where named entities are recognized from sentences directly, we generate possible name variations based on word usage patterns and mobile phone naming conventions. The pre-generation of candidate names also enables us to obtain a large number of training examples at very low cost for manual annotation. Through extensive experiments, we show that our proposed GREN method significantly outperforms baseline methods, particularly in the recall measure for name recognition and normalization. The GREN method is flexible in the sense that both the candidate names and the name normalization rules can be easily modified to accommodate special cases in practical applications. Although the proposed method cannot be directly applied to detect names for many other products, we believe that the overall concepts of candidate name generation and candidate name mention prediction can be adopted in product name detection and normalization for products with common naming conventions in user generated content. An example of such products is camera lens which is often named by its focal length and the key feature, *e.g.,* "17-

85IS" is a name variant of Canon EF-S 17-85mm f/4-5.6 IS USM Lens.

Acknowledgement

This work was partially supported by Singapore MOE AcRF Tier 2 Grant MOE2014-T2-2-066.

References

1. P. F. Brown, P. V. deSouza, R. L. Mercer, V. J. D. Pietra, and J. C. Lai. Class-based n-gram models of natural language. *Computational Linguistics*, 18:467–479, 1992.
2. A. M. Cohen. Unsupervised gene/protein named entity normalization using automatically extracted dictionaries. In *ACL-ISMB Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics*, pages 17–24. ACL, 2005.
3. S. Cucerzan. Large-scale named entity disambiguation based on Wikipedia data. In *EMNLP-CoNLL*, pages 708–716. ACL, 2007.
4. D. M. de Oliveira, A. H. Laender, A. Veloso, and A. S. da Silva. FS-NER: A lightweight filter-stream approach to named entity recognition on twitter data. In *WWW (Companion)*, pages 597–604, 2013.
5. J. R. Finkel, T. Grenager, and C. Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *ACL*, pages 363–370. ACL, 2005.
6. J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289. Morgan Kaufmann Publishers, 2001.
7. C. Li, A. Sun, and A. Datta. TSDW: two-stage word sense disambiguation using wikipedia. *JASIST*, 64(6):1203–1223, 2013.
8. C. Li, A. Sun, J. Weng, and Q. He. Exploiting hybrid contexts for tweet segmentation. In *SIGIR*, pages 523–532. ACM, 2013.
9. C. Li, A. Sun, J. Weng, and Q. He. Tweet segmentation and its application to named entity recognition. *IEEE Trans. Knowl. Data Eng.*, 27(2):558–570, 2015.
10. P. Liang. Semi-supervised learning for natural language. Master's thesis, Massachusetts Institute of Technology, 2005.
11. X. Liu, Y. Li, H. Wu, M. Zhou, F. Wei, and Y. Lu. Entity linking for tweets. In *ACL*, pages 1304–1311. ACL, 2013.
12. X. Liu, S. Zhang, F. Wei, and M. Zhou. Recognizing named entities in tweets. In *ACL*, pages 359–367. ACL, 2011.
13. C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.
14. A. McCallum and W. Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *HLT-NAACL (Volume 4)*, pages 188–191. ACL, 2003.
15. E. Meij, W. Weerkamp, and M. de Rijke. Adding semantics to microblog posts. In *WSDM*, pages 563–572. ACM, 2012.
16. R. Mihalcea and A. Csomai. Wikify!: Linking documents to encyclopedic knowledge. In *CIKM*, pages 233–242. ACM, 2007.
17. D. Nadeau and S. Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30:3–26.
18. L. Philips. Hanging on the metaphone. *Computer Language Magazine*, 7(12):39–44, 1990.
19. L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, pages 257–286, 1989.
20. L. Ramshaw and M. Marcus. Text chunking using transformation-based learning. In *Proc. of the 3rd Workshop on Very Large Corpora*, pages 82–94. ACL, 1995.
21. L. Ratinov and D. Roth. Design challenges and misconceptions in named entity recognition. In *CoNLL*, pages 147–155. ACL, 2009.
22. A. Ritter, S. Clark, Mausam, and O. Etzioni. Named entity recognition in tweets: An experimental study. In *EMNLP*, pages 1524–1534. ACL, 2011.
23. F. Sebastiani. Machine learning in automated text categorization. *ACM Comput. Surv.*, 34(1):1–47, Mar. 2002.
24. S. Wu, Z. Fang, and J. Tang. Accurate product name recognition from user generated content. In *IEEE ICDM Workshops*, pages 874–877, 2012.
25. Y. Yao and A. Sun. Product name recognition and normalization in internet forums. SIGIR Symposium on IR in Practice (SIGIR Industry Track), July 2014.