Identifying Complaints from Product Reviews: A Case Study on Hindi

Raghvendra Pratap Singh[†], Rejwanul Haque, Mohammed Hasanuzzaman[‡], and Andy Way

ADAPT Centre, [†]School of Computing, Dublin City University, Dublin, Ireland
[‡]Cork Institute of Technology, Cork, Ireland
raghvendra.singh6@mail.dcu.ie
{rejwanul.haque,mohammed.hasanuzzaman,andy.way}@adaptcentre.ie

Abstract. When an expectation does not meet reality in a real-world situation, the difference is usually expressed and communicated via an act which is complaint. Customers often post reviews on the products or services they purchase on the retailer websites and different social media platforms, and the reviews may reflect complaints about the products or services. Automatic recognition of customers' complaints on products or services that they purchase can be crucial for the organisations, multinationals and online retailers since they can exploit this information to fulfil the customers' expectations including managing and resolving the complaints. In this work, we present the supervised and semi-supervised learning strategies to identify users' complaints from the language they use to post their reviews. In other words, we automatically identify complaints from the opinionated texts (reviews) about products posted in Hindi. For this, first we automatically crawled the Hindi reviews on different products from the the websites of the retail giant Amazon and the popular social media platform YouTube, and prepared a gold-standard data set via a systematic manual annotation process. We use state-of-the-art classification algorithms for the complaints identification task and our classification models achieve reasonable classification accuracy ($F_1 = 68.38\%$) on a gold-standard evaluation test set.

Keywords: Random walk · LSTM · fastText · Dice coefficient · SMOTE

1 Introduction

Text classification is an active field of natural language processing (NLP) and data mining. Almost all online retailers allow users to freely express their opinions and thoughts on products via their websites and the relevant social media platforms. The customers who intend to purchase a product may take purchasing decisions based on the reviews of the product. Accordingly, the commercial and retail companies considers product reviews as an important source of information, and could exploit this information to build their marketing tool and strategy, and to resolve any issues in relation to the product. This could also

benefit the users with the suggestions on the quality of the products or services that they want to purchase. As for the number of reviews of a product posted by the users, they could range from several hundreds to tens of thousands. The e-commerce companies and online retailers want to identify complaints given the reviews of a product for their own benefit. Likewise, the customers who want to buy a product or service may need such information while avoiding having to go through thousands of reviews about the product.

In this context, Gupta et al. [5] identified the relationship between users' purchase intent from their social media forums such as Quora[1] and Yahoo! Answers.[2] They primarily carried out text analysis to detect purchase intent from user-generated content (UGC). Wang et al. [13] investigated the problem of identifying purchase intent. In particular, the authors proposed a graph-based learning approach to identify intent tweets and classify them into six categories. For this, they retrieved tweets with a bootstrap method, with using a list of seed intent-indicators (e.g. 'want to'), and manually created training examples from the collected tweets. Haque et al. [6] extends the work of Wang et al. [13] while increasing the coverage of the purchase intent indicators with the distributed vector representation of words using the continuous skip-gram model [10].

Recently, Preotiuc-Pietro et al. [11] automatically identify complaints from the tweets posted by the social media users and the potential customers. To the best of our knowledge, the most relevant works to ours come from Preotiuc-Pietro et al. [11]. In fact, to a certain extent, our proposed methods can be viewed as the extension of Preotiuc-Pietro et al. [11] with mainly the following additions as far as this task is concerned: (i) we considered product reviews instead of tweets as in Preotiuc-Pietro et al. [11] and (ii) we explore a resource-poor and less-explored language, Hindi, for our investigation, (iii) we applied a state-of-the-art sampling strategy [2] in order to encounter class imbalance problem in the training data, and (iv) we explore applying a semi-supervised classification algorithm in the complaint identification task. Moreover, one of the key contributions to this work is creation of the gold-standard dataset in Hindi.

The remainder of the paper is organised as follows. In Section 2, we detail how we semi-automatically created training data for our experiments. In Section 3, we present our experimental methodology and setups. Section 4 presents our evaluation results, with some discussions. Section 5 concludes and provides avenues for further work.

2 Dataset Creation

This section details the creation of training data that has been used in this task.

[1] www.quora.com
[2] www.answers.yahoo.com

2.1 Collecting Hindi Reviews

To the best of our knowledge, there is no existing (freely available) annotated review data (complaint and non-complaint) for Hindi. For this task we needed an annotated review dataset for Hindi. In order to create an annotated review dataset for Hindi, we first collected Hindi reviews posted online. The reviews were taken from two different sources: (i) the websites of the retail giant Amazon, and (ii) YouTube. The users usually post their Hindi reviews on these two platforms.

In order to collect the reviews from Amazon, we used *amazon-reviews-scraper Python library*[3] which takes a product name as input and provides the reviews about the product across the different languages. Similarly, in order to collect the reviews from YouTube, we used *youtube-comment-downloader Python library*.[4] This script provided us reviews on the products across the different languages.

In order to remove noise (e.g. HTML tags, special characters) from reviews, we applied a number cleaning scripts including a language identifier.[5] We also removed emojis from the review texts. Each of the collected clean reviews is manually tagged with a particular category, namely complaint or non-complaint. The annotation scheme and results are presented in the next section.

Table 1. Sample Dataset. A1–3: three annotators.

Reviews	A1	A2	A3	Final Annotation
सोनी का साउंड बहुत बेहतरीन होता इसलिए मुझे ये बहुत पसंद है....	0	0	0	0
प्रयागराज मे जियो बहुत ही घटिया चल रहा है	1	1	1	1
फ़ोन का कैमरा बहुत अच्छी है डिस्पले भी और बैटरी तो तबाही हैं	0	0	0	0
पृष्ठ संख्या के बाद के पृष्ठ उल्टे लगाए गए हैं	1	1	1	1
सब इधर उधर का चोरी किया हुआ लिखा है...मकसद फिल्मों में स्क्रिप्ट लिखने का काम करना...	1	1	1	1
माउस काम नहीं कर रहा सर्विस बेकार है	1	1	1	1
क्या खूब लिखा है मजा आ गया सत्य भैया ऐसे ही लिखते रहे	0	0	0	0
काफी रोमांचक है और कुछ हट के भी	0	0	0	0
कहानी का शानदार आग़ाज़ बेहतरीन अंदाज़ आप आगे पढ़ने के लिए मजबूर होते हैं	0	0	0	0
चार्जिंग करते समय बहुत गरम हो जाता है, फटने का दर है	1	1	1	1

2.2 Annotation

The annotation task is performed in-house by our three undergraduate students. Each reviews is presented to the annotators. The annotators are expected to

[3] `https://github.com/philipperemy/amazon-reviews-scraper`. Accessed on August 2020

[4] `https://github.com/egbertbouman/youtube-comment-downloader`. Accessed on August 2020.

[5] `https://pypi.org/project/pycld2/`

answer two questions for a given review. While the first question is related to the decision as to whether a review is *complaints* or *non-complaints*, the motivation behind the second question is to collect a more fine-grained (book, phone, tv etc.) gold standard dataset.[6]

To have a concrete idea about the agreement between annotators, we calculated the majority class for each review in our dataset. A review belongs to a majority class k if the most frequent annotation for the review was selected by at least k annotators. As a consequence, a large percentage of review belonging to high majority classes are symptomatic of good inter-annotator agreement. Similar to earlier studies, we consider all annotations with a majority class greater than 2 as reliable. In this case, for the *complaints* or *non-complaints* annotation scheme, over 88% of the review were annotated identically by the majority of annotators, while for fine-grained annotation scheme,[7] over 85% of the annotations fell into this case. As such, we can be confident that the annotation process was successful and the dataset is reliable. A sample of our annotated dataset is presented in Table 1, and statistics about the dataset are presented in the next section.

2.3 Data Statistics

We report the statistics of the our gold standard data set in Table 2. The table shows the numbers of the complaint and non-complaint reviews which are listed according to different product types.

Table 2. The collected Hindi reviews listed according to product types.

Category	Non-complaints	Complaints
Book	2,722	117
Phone	335	339
Headphone	30	18
Watch	26	17
Misc.	32	75
Total	3,145	566

We divided the annotated set of reviews (i.e. 3,711 reviews) into train, development and test sets. The test and development set reviews were randomly sampled from the all reviews. The statistics about the train, development and test set reviews are shown in Table 3. The training, test and development data sets have been released publicly and can be downloaded from `https://github.com/MrRaghav/Complaints-mining-from-Hindi-product-reviews`.

[6] The details of the annotation guidelines are out of the scope of this paper.

[7] Annotation of reviews tagged with 'complaints' into product types such as phone, book, tv etc.

Table 3. Statistics of the train, development and test sets reviews.

	Reviews	Words	Complaints
Train set	2,967	105,322	452
Dev. set	372	13,966	57
Test set	372	14,038	57
Total	3,711	133,326	566

3 Methodology

3.1 The LSTM Network

Nowadays, recurrent neural network (RNN), in particular with long-short term memory (LSTM) [7] hidden units, has been proved to be an effective model for many classification tasks in NLP, e.g. sentiment analysis [14], text classification [8, 15]. RNN is an extension of the feed-forward neural network (NN), which has the gradient vanishing or exploding problems. LSTM deals with the exploding and vanishing gradient problems of RNN. An RNN composed of LSTM hidden units is often called an LSTM network. A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. More formally, each cell in LSTM can be computed as follows:

$$X = \begin{bmatrix} h_{t-1} \\ x_t \end{bmatrix} \tag{1}$$

$$f_t = \sigma(W_f \cdot X + b_f) \tag{2}$$

$$i_t = \sigma(W_i \cdot X + b_i) \tag{3}$$

$$o_t = \sigma(W_o \cdot X + b_o) \tag{4}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot tanh\ (W_c \cdot X + b_c) \tag{5}$$

$$h_t = o_t \odot tanh\ (c_t) \tag{6}$$

where $W_i, W_f, W_o \in \mathbb{R}^{d \times 2d}$ are the weighted matrices and $b_i, b_f, b_o \in \mathbb{R}^d$ are biases of LSTM, which need to be learned during training, parameterising the transformations of the input, forget and output gates, respectively. σ is the sigmoid function, and $\odot$ stands for element-wise multiplication. x_t includes the inputs of LSTM cell unit. The vector of hidden layer is h_t. The final hidden vector h_N represents the whole input review, which is passed to *softmax* layer after linearising it into a vector whose length is equal to the number of class labels. In our work, the set of class labels includes complaint and non-complaint categories.

3.2 fastText

We used *fastText* [8], a fast and efficient classifier, in our task. fastText is often on par with the state-of-the-art deep learning classifiers in terms of accuracy. It is also found to be faster for training and evaluation. fastText has many advance features, e.g. hierarchical softmax, hidden states shared among features and classes, and bag of n-grams features.

3.3 Graph-based Classifier

We also used a semi-supervised classification algorithm, i.e. *random walk* on graph. A graph, G is a pair of vertices V and edges E. We use a weighted graph for the classification. A weighted graph is a graph where each edge, E_i, is linked to a numeric value W_{ij} where i, $j \in V$. For our approach, it is necessary that

$$W_{\mathrm{ij}} = W_{\mathrm{ji}} \tag{7}$$

Random walk is based on the concept of randomly determined processes. It considers a transition probability of the random walker that it will reach from a vertex V_i to the other vertex V_j with each step being either +1 or -1 with equal probability. As for implementing random walk for this task, we follow the approach and python implementation of the algorithm described in [12].

3.4 Classical Supervised Classification Models

Furthermore, we compare the deep learning model (the LSTM network), fastText and semi-supervised classification model (random walk) presented above with the classical supervised classification models. We employ the following classical supervised classification techniques in our task:

- Logistic Regression (LR)
- Decision Tree (DT)
- Random Forest (RF)
- Naïve Bayes (NB)
- Support Vector Machine (SVM)

These classical learning models (LR, DT, RF, NB and SVM) can be viewed as the baselines in this task. Thus, we obtain a comparative overview on the performances of different supervised and semi-supervised classification models including the LSTM network.

3.5 Training Setup

In order to build LR, DT, RF and NB classification models, we use the well-known *scikit-learn* machine learning library,[8] and performed all the experiments

[8] https://scikit-learn.org/stable/

with default parameters set by scikit-learn. As for the representation space, each review was represented as a vector of word unigrams weighted by their frequency in the reviews.

For the classifiers based on the neural networks, we use a 300-Dimensional word embeddings from *fastText*. We use *sigmoid* activation function with Adam optimizer [9] and binary cross entropy loss function. The size of input layer of the NN is 300. We employ layer normalisation [1] in the model. Dropout [4] between layers is set to 0.10. The size of embedding and hidden layers are 300. The learning-rate is set to 0.0003, and the training examples were reshuffled for each epoch.

As for the random walk algorithm, we use the similarity between the sentences as the transition probability. Again, we create a network with the training examples. We take a set of labelled vertices L and unlabelled vertices U, where $(L, U) \subseteq G$ (G : set of all the labels). The random walk algorithm considers the vertices of the network as the states of a Markov chain. We calculate the similarity measure between the data points with the Dice coefficient (DC) [3] and reach to the final decision according to the probability. The DC measure has been widely used for evaluating the degree of association between words.

3.6 Handling Class Imbalance

We recall Tables 2 and 3 where we can see the presence of class imbalance in the training data. In order to encounter the class imbalance problem in our training data, we followed Chawla et al. [2] who presented a combination of over-sampling of minority class with under-sampling of majority class. This is also called *SMOTE* which is a kind of synthetic sampling technique and found to be effective in our problem.

4 Results and Discussion

We evaluate the performance our classifiers on the gold-standard test set (cf. Table 3) and report the evaluation results in this section. In order to measure classifier's accuracy on the test set, we use three widely-used evaluation metrics: precision, recall and F_1 measures. The results obtained are reported in Table 4. The first five rows of Table 4 represent our baseline classifiers (i.e. the classical supervised classification models). The next row represents the random walk algorithm. We see from the table that these classifiers performs below par and SVM is the best-performing method among them (SVM: a 47.94 F_1 score)) according to the F_1 scores.

As for NN-based classifiers, the LSTM network trained on fastText embeddings performed reasonable as we see from Table 4 that it produces a moderate F_1 score (68.38 F_1) on the test set. The fastText classifier also performs reasonably well; however, it could not surpass the LSTM network.

In Table 5, we show the F_1 scores of the best-performing classifiers of the complaint identification task presented in [11]. As can be seen from Table 5,

Table 4. Performance of the classifiers on the evaluation test set.

	Precision	Recall	F_1
NB	24.82	63.15	35.64
LR	40.44	63.15	49.31
DT	36.25	50.87	42.33
SVM	39.32	61.40	47.94
RF	52.63	35.08	42.10
Random walk	43.84	23.06	30.22
LSTM	66.67	70.18	68.38
fastText	72.34	59.64	65.38

their best complaint identification models produce F_1 scores in the range of 78–79 which are more than 10 F_1 points higher than the F_1) score of our best-performing complaint identification model (cf. second last row of Table 4). Note that, unlike us, Preotiuc-Pietro et al. [11] carried out experiments on English and the data sets including the evaluation test set are different too. Moreover, the nature of data is also different to us. Preotiuc-Pietro et al. [11] focused on identifying complaints on tweets and we focused on identifying the same on the product reviews posted on Amazon and YouTube. Naturally, the scores presented in Preotiuc-Pietro et al. [11] cannot be directly compared to those presented in this paper. Given the fact that English is a high resource language and there are a plenty of linguistic tools and resources freely available in English, Preotiuc-Pietro et al. [11] exploited many linguistic resources, models and tools in their task for feature engineering, e.g. sentiment and emotion analysis, temporal model, part-of-speech information. In Hindi, it is difficult to obtain such resources and standard tools, and many such tools are not available to use. Moreover, unlike English, Hindi is a morphological complex and highly inflected language. Therefore, we believe that identifying complaints on Hindi texts is to be more challenging in comparison to that in English or other high resource languages.

Table 5. The best-performing complaint identification classifiers (English) presented in Preotiuc-Pietro et al. [11].

	F_1
Best model (distant supervision)	79.0
Second best model (LR)	78.0

5 Conclusion

In this paper, we presented supervised and semi-supervised learning models to identify customers' complaints from the review data in Hindi, a low-resource, less-

explored, and morphological rich highly inflected Indic language. There is no publicly available gold-standard training data as far as identifying complaints from user generated Hindi texts about products is concerned. Accordingly, we semi-automatically created a gold-standard dataset for complaints identification on Hindi. We conducted our experiments with state-of-the-art LSTM classifier and classical supervised classification models. With our LSTM classifier, we achieved a competent accuracy (a F_1 score of 68.38) on the gold-standard evaluation test set. There are limited linguistic resources and tools freely available for research in Hindi in comparison to many high-resource languages including English. We believe that this work would add an additional value to the social media analytics research in low-resource scenarios.

In future, we intend to test our method on different low-resource and non-English languages. We also plan to investigate applying more sophisticated and linguistic features in our model, e.g. part-of-speech information. We removed emojis from reviews at the time data prereprocessing. Since emojis may capture polarity and encode information regarding users' experiences and complaints, in future, we aim to carry out experiments while keeping emojis in the reviews. Our classifiers were trained on the data which is a mixture of reviews from different product types. We intend to train classifiers on reviews from specific product type, i.e. book or phone reviews. By this, we can compare the classifiers trained on the data from individual product type to the one trained on the data consisting reviews from all product types.

References

1. Jimmy Lei Ba, Jamie Ryan Kiros and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.0645*.
2. Kevin W. Bowyer, Nitesh V. Chawla, Lawrence O. Hall and W. Philip Kegelmeyer. 2002. SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 16:321–357.
3. Lee R. Dice. 1945. Measures of the Amount of Ecologic Association Between Species. *Journal of Ecology*, 26(3):297–302.
4. Yarin Gal and Zoubin Ghahramani. 2016. A Theoretically Grounded Application of Dropout in Recurrent Neural Networks. In *Advances in neural information processing systems*, pages 1019–1027, Barcelona, Spain.
5. Vineet Gupta, Devesh Varshney, Harsh Jhamtani, Deepam Kedia and Shweta Karwa. 2014. Identifying Purchase Intent from Social Posts. In *Proceedings of the Eighth International AAAI Conference on Weblogs and Social Media*, pages 180–186, Ann Arbor, MI.
6. Rejwanul Haque, Arvind Ramadurai, Mohammed Hasanuzzaman and Andy Way. 2019. Mining Purchase Intent in Twitter. *Computación y Sistemas*, 23(3):871–881
7. Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, pages 9(8):1735–1780. MIT Press.
8. Armand Joulin, Edouard Grave, Piotr Bojanowski and Tomas Mikolov. 2017. Bag of Tricks for Efficient Text Classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431, Valencia, Spain.

9. Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*.
10. Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, Lake Tahoe, NV.
11. Daniel Preotiuc-Pietro, Mihaela Gaman and Nikolaos Aletras. 2019. Automatically Identifying Complaints in Social Media. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistic*, pages 219–224. Los Angeles, CA.
12. Martin Szummer and Tommi Jaakkola. 2002. Partially labeled classification with Markov random walks: Advances in neural information processing systems. In *Advances in neural information processing systems*, pages 945–952, Vancouver, BC.
13. Jinpeng Wang, Gao Cong, Wayne Xin Zhao and Xiaoming Li. 2015. Mining user intents in twitter: a semi-supervised approach to inferring intent categories for tweets. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 318–324. Austin, TX.
14. Yequan Wang, Minlie Huang, Xiaoyan Zhu and Li Zhao. 2016. Attention-based LSTM for aspect-level sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 606–615, Austin, TX.
15. Peng Zhou, Zhenyu Qi, Suncong Zheng, Jiaming Xu, Hongyun Bao and Bo Xu. 2016. Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling. *arXiv preprint arXiv:1611.06639*.