Multi Grain Sentiment Analysis using Collective Classification

S.Shivashankar [1] and **B.Ravindran** [2]

Abstract. Multi grain sentiment analysis is the task of simultaneously classifying sentiment expressed at different levels of granularity, as opposed to single level at a time. Models built for multi grain sentiment analysis assume fully labeled corpus at fine grained level or coarse grained level or both. Huge amount of online reviews are not fully labeled at any of the levels, but are partially labeled at both the levels. We propose a multi grain collective classification framework to not only exploit the information available at all the levels but also use intra dependencies at each level and inter dependencies between the levels. We demonstrate empirically that the proposed framework enables better performance at both the levels compared to baseline approaches.

1 Introduction

Sentiment analysis is one of the tasks which benefits from increasing web usage and reach in the form of forums, blogs and other websites that hold product reviews and discussions. In general, sentiment analysis is the task of identifying the sentiment expressed in the given piece of text about the target entity discussed. Depending on the target entity, the granularity of the analysis varies. The target entity can be the product itself, for example "Canon digital camera", in which case it is called as coarse-grained analysis. On the other hand the target entity can also be at finer granularity, capturing various features of a product, for example "clarity of the camera", in which case it is called as fine-grained analysis. This multi grain sentiment analysis can be achieved by performing analysis at various levels of granularity — document level, paragraph level/sentence level/phrase level — which basically captures product level, sub topic level or feature level target sentiments. The former refers to physical structure of the text taken for analysis, while the latter corresponds to logical level. We use this notion of physical and logical levels in granularity throughout the paper. There are models built at each level individually [1, 2, 3, 4, 5, 6] which are called as independent models [7]. Considering the nature of information available, the emphasis of recent approaches is towards exploring the intra dependencies at each level and inter dependencies between the levels. Intuition behind intra dependency between entities at a single level is explained using the following example. In the fragment of automobile review text given below :

"The manual gear shifter is rubbery."

If the sentiment about manual gear shifter is unknown, then the sentiment about similar features such as driving experience in *"..has an unpleasant driving experience.."* can help in disambiguating the sentiment.

Inter dependency between coarser and finer levels are also useful in predicting the unknown sentiments [7, 8]. For instance, if the sentiment of a document is known, then a majority of the sentences should have the same sentiment as the document and vice versa. This forms the basis of the proposed model, since we use these dependencies in the multi grain collective classification framework. Since the above mentioned approaches [7, 8] assume fully labeled corpus, which is not naturally available and huge amount of web data is partially labeled, we propose a multi grain collective classification algorithm for the semi-supervised environment. We focus on document level and sentence level analysis in this work.

2 Related Works

Pang and Lee [8], used the local dependencies between sentiment labels on sentences to classify sentences as subjective or not, and the top subjective sentences are used to predict the document level sentiment. It can be seen as a cascaded fine to coarse model and was shown to be better than other document level models. Following this work which gives the motivation to study these dependencies, McDonald, Ryan et al.,[7] proposed a joint structured model for sentence and document level sentiment analysis. In this work, they model document level sentiment using sentences level sentiment, sentence level sentiment using other sentences in the local context and document-level sentiment. This model was proven to be better than cascaded models (both fine to coarse and coarse to fine) and independent models at both the levels. Both the approaches assume fully labeled data. Minimum cut based approach [8] uses labeled sentences to predict document level sentiment, and the structured model [7] uses labeled documents and sentences to build the joint model. As the data available in web is naturally partially labeled, and it would involve human annotation to get fully labeled corpus, we do not assume the data to be fully labeled at any level. Secondly the above mentioned approaches capture the structural cohesion between sentences i.e., sentences occurring in physical proximity (next to each other) and not the logical cohesion, i.e., sentences discussing about similar features. The approach utilizing dependency between sentences based on logical cohesion captured using anaphora and discourse relations has been shown to perform better than other approaches [9]. Also it is a sentence level approach and not a multi grain approach as the proposed model. The issues with the existing approaches and the way proposed model differs from those approaches is briefly explained below:

- To the best of our knowledge, no framework has been proposed for multi grain sentiment analysis in a semi-supervised environment,

[1] Department of Computer Science and Engineering, Indian Institute of Technology Madras, Chennai, India, email: shivas@cse.iitm.ac.in
[2] Department of Computer Science and Engineering, Indian Institute of Technology Madras, Chennai, India, email: ravi@cse.iitm.ac.in

i.e., data is not fully labeled at any of the levels. Only a subset of documents are labeled with document level sentiment, and again only a subset of sentences in a document are labeled in the form of 'pros and cons'.

- In the above mentioned approaches, dependencies captured either at structural or logical level, expect the text to be written in an ideal manner for better performance. Structural cohesion based approaches expect sentences discussing related features to be written next to each other, and the logical cohesion based approaches captured using discourse graph [9] expect the sentences to have explicit anaphoric and discourse relations.
- Another disadvantage of the discourse graph based approach is that it needs anaphora resolution and discourse structure identification to be performed for all input documents. It also ignores background domain knowledge available in the form of domain taxonomy, knowledge bases like Wordnet, and fully relies on discourse graph construction methods. Also with the availability of huge amount of text it is possible to perform knowledge engineering and build a domain knowledge base which captures features of a domain and similarity between the features.

In this paper we propose an iterative classification algorithm which performs multi grain classification in a semi-supervised environment. Intra-dependencies at sentence level are captured using domain knowledge base, i.e., relation between features of a domain. The advantage is that it can be prebuilt, and instantiated for each document. This can be seen as adding domain knowledge to avoid sparsity that might arise in discourse graph based techniques. Since the construction of domain knowledge is not the focus of this paper, we do not discuss it in detail apart from briefly explaining the knowledge base used for this work in the experimental section.

3 Proposed Approach

We define the problem scenario and notation as follows. Let C be a corpus of web based review documents. A subset of review documents have a 'pros and cons' section, which has positive and negative sentences. The 'pros and cons' sentences have phrases which are generally comma or semi-colon separated. An example sentence taken from pros section of a laptop review in CNET website is given below

> *"Slim design; easy-to-use Intel Wireless Display built-in; speedy Core i5 processor."*

Since not all the sentences in the document are labeled, those documents are referred to as sentence level partially labeled data, S. Also a subset of documents contain overall sentiment label of the product, in the form of star rating, or scores between 1 and 10, or binary labels such as YES or NO. In this paper only binary labels are considered. Since only a subset of documents have document level sentiment label, they are referred to as document level partially labeled data, G. Documents that either have 'pros and cons' section or document level sentiment label or both are referred to as multi grain partially labeled data, M. Documents that do not have any labels are called as unlabeled data, U. In general, a document level sentiment label, can be seen as a function, Y_d of sentence level sentiment labels. It is stated formally as follows: D denotes a document containing a set of sentences, $D = \{s_1, s_2, s_3, ..., s_n\}$. Sentiment label is denoted by Ω, this yields the below formulation

$$\Omega(D) = Y_d(\Omega(s_i)), \forall s_i \text{ where } s_i \epsilon D$$

Each s_j in turn can be seen as a set of sentiment terms O_j, where $O_j \epsilon s_j$. Thus sentence level sentiment label, $\Omega(s_j)$ can be seen as a function, Y_{us} of sentiment term level labels. Sentiment terms are words that carry polarity, and the most commonly used class of words are adjectives. We refer to this as ***unigram based classification***. It is stated as below

$$\Omega(s_j) = Y_{us}(\Omega(O_j)), \forall O_j \text{ where } O_j \epsilon s_j$$

Precision issues in unigram based classification is due to the fact that lexicon W is not domain and context specific [3].

- **Domain adaptation** : This includes polarity mismatch of terms in the lexicon for different domains, and also expansion of the lexicon for evolving domain oriented sentiment terms. For example, it is not possible to predict the sentiment of the term *"rough"* in the sentence *"...surface is rough..."*, independent of the domain. It will be negative in the case of products like camera, and positive in the case of tyre. General lexicon might fail in this case depending on the domain. Also, we pose the problem of dealing with missing sentiment terms as one of improving precision, by adding the sentiment terms found in the corpus to the lexicon and assigning arbitrary labels to it. So the polarity has to be relearned using the evidences in multi grain partially labeled dataset M.

- **Context specificity issues** : Set of sentiment terms have different sentiments in different context, i.e., when they modify different target features. For example, the same sentiment term *"huge"* will be positive in *"huge win"*, and negative in *"huge loss"*. Since it is not possible to capture the context using the sentiment term alone, the unigram based classification faces precision issues in these cases.

Alternatively, s_j can be seen as a collection of constituent target feature term and sentiment term pairs, thus sentiment label at sentence level $\Omega(s_j)$ can be seen as a function, Y_{ts} of tuple's labels $\Omega(F_j, O_j)$. We refer to it as ***tuple based sentence classification***. It is given formally as below

$$\Omega(s_j) = Y_{ts}(\Omega(F_j, O_j)), \forall (F_j, O_j) \text{ where } (F_j, O_j) \epsilon s_j$$

Note that "tuple" and "phrase" have the same meaning. Tuple is used in the proposed model for convenient usage of the aforementioned pair notation. The same examples, *"huge win"* and *"huge loss"*, have different tuples $(win, huge)$ and $(loss, huge)$. And the classification is based on these tuples. Kaji and Kitsuregawa [4] extract 'pros and cons' phrases from huge collection of Japanese HTML documents whose polarity is computed using chi-square and PMI values. The phrases are used to classify sentences, and the method showed high precision but low recall because of sparsity, since it is tough to collect data for all possible phrases. Thus the proposed model must solve the domain adaptation, context specificity, and sparsity issues in order to build a framework which has proper balance between precision and recall. The proposed model employs tuple based sentence classification as the base classifier, and iteratively improves its recall without compromising much of the precision. It uses unigram evidence for sentiment classification when there is no evidence at tuple level. The model can be seen as an undirected graph which contains document (D), sentence(s) and tuple level(t) nodes, in which the tuple classification model combines evidences at tuple and unigram level. This is given by the backoff inference procedure in Algorithm 1. Henceforth, the notation of (F_j, O_j) is used to mention a tuple

t_j. The following section 3.1 explains the observations and intuitions behind the proposed model, which helps in bridging the gap between unigram and tuple based sentence classification techniques.

3.1 Intuition

The following are the observations that act as the base for proposed model. Note that we assume to have the background domain knowledge that has the set of features and similarity between the features. We denote this knowledge base as K_b, and the lexicon of sentiment terms (say for instance General Inquirer) is denoted as W. The effect of negating terms such as "not", "except" is handled using a hand compiled list. For example *"sound clarity is not good"* is captured as $(sound\ clarity, NOT\_good)$ and the label is reversed accordingly.

Observation 1 - Simple propagation This observation shows how more tuples can be learned compared to Kaji and Kitsuregawa's approach [4], which just uses phrases from 'pros and cons' sentences. A feature has the same polarity in a review. Due to the fact that the same feature can be mentioned in multiple sentences, this leads to more interesting cases where more tuples can be learned than just using 'pros and cons' sentences. Those cases are given below

- Case 1: In a document's 'pros and cons' section, the sentiment term used to modify a feature term will not always be identical to the one used in detailed review section. For example tuple in pros section is $(steering, quick)$, while the review contains $(steering, firm)$. This helps to infer that $(steering, firm)$ is positive.
- Case 2: The tuples in 'pros and cons' section may not be always complete. It can have only the feature term specified. For example $(touchscreen\ interface, NULL)$ in pros section, the review part has $(touchscreen\ interface, comfy)$. From this $(touchscreen\ interface, comfy)$ can be learned as positive. NULL denotes the absence of sentiment term in the tuple.

Observation 2 : Intra tuple label propagation Tuples will have same sentiment in a domain across review documents. So a tuple already seen as positive or negative can be used to label future occurrences of the tuple in the corpus. Not just identical tuples, but also the ones that share similar features. For example, $(navigation\ system, reachable)$ can be labeled if the tuple $(center\ stack, reachable)$ is already labeled. The similarity is not just based on parent-child relationships between features, it also includes synonyms or substitutable terms, such as "looks" and "appearance". This propagation is stated formally in the backoff model based inferencing procedure, which is referred to as backoff inference procedure. The lexicon with labeled tuples in the corpus is denoted as L_t, also note that the lexicon is updated through the process when labels of more tuples are learned.

Observation 3 : Top-down label propagation Not all sentiment terms require domain and context information for classification. Terms like "good", "wonderful" are positive independent of domain. So the general lexicon W, has a mix of domain independent and domain dependent sentiment terms. In order to avoid domain adaptation issues, the words' label must be relearned for the target domain using the multi grain partially labeled dataset M. The predominant label of the sentences and documents it occurs in, is taken as the sentiment terms' label. The evidences at sentence and document level are combined using linear interpolation, which is given below

$$\Omega(O_k) = argmax_l(w_s * P_{O_k ls} + w_d * P_{O_k ld})$$

$$P_{O_k ls} = \frac{N_{O_k ls}}{N_{O_k s}}$$

$$P_{O_k ld} = \frac{N_{O_k ld}}{N_{O_k d}}$$

where l denotes the label space, $l \epsilon \{+,-\}$, and w_s, w_d denote the weightage for sentence and document level evidence, also $w_s + w_d = 1$. $P_{O_k ls}$ is the sentence level evidence that the sentiment term O_k has the label l. $N_{O_k ls}$ denotes number of sentences containing O_k, and labeled l. $N_{O_k s}$ denotes number of sentences containing O_k. Similarly $P_{O_k ld}$, $N_{O_k ld}$ and $N_{O_k d}$ are at document level. As the number of sentences required to reliably predict a sentiment term's label is lesser compared to the number of documents, we treat sentence level evidence as more reliable than document level's. So the weightage is formulated as follows

$$N_{O_k ls} > 0 \rightarrow \{w_s = 1, w_d = 0\}$$
$$N_{O_k ls} = 0 \rightarrow \{w_s = 0, w_d = 1\}$$

This can be seen as top-down label propagation, where sentences and documents are used to relearn the polarity of sentiment terms. Note that relearning of polarity of words in W using M does not fully solve the domain adaptation issue. Since it depends on the amount of labeled data available. We denote the lexicon with labeled unigrams, which is built using W and M as L_u. Thus the unknown label of a tuple (F_q, O_q) can be inferred based on L_t and L_u using a back-off model. The intuition is that L_u can be used to infer sentiment of a tuple when it cannot be inferred using L_t. In the Algorithm 1, L_{lt} denotes the label l given by L_t for the input tuple (tuple based classification), similarly L_{lu} denotes the label l given by L_u for the input sentiment term (unigram based classification). If the tuple has evidence to be classified, then its label is committed. Also it is obvious that the label identified using committed labels of tuples is more reliable than uncommitted labels of tuples and prediction using unigrams. Thus we assign a confidence flag for each prediction. If the prediction is not confident then the labels are further refined using label smoothing, which is given in next observation.

Observation 4 : Label Smoothing The intuition is that similar features tend to have the same sentiment label in a single review document. Feature terms used in 'pros and cons' need not be identical to the ones given in detailed review section; it might have similar features. For example $(performance, good)$ in pros section, would imply that the tuple $(engine, high - revving)$ in detailed review section is positive, as engine and performance are similar features. This also applies to any labeled tuple t_k in a document, where the tuples t_i which have target feature terms F_i that are similar to F_k will have more probability for t_k's label. This along with observation 2 defines the intra-dependency at fine-grained level, where the neighborhood is defined by the domain knowledge base K_b, and is instantiated for each document. Note that the difference between intra tuple label propagation discussed in observation 2 and label smoothing is that, observation 2 expects F_i and F_k to be similar and O_i and O_k be identical. But it is not the case in observation 4, which only expects F_i and F_k to be similar. Also it is obvious that observation 2 applies to neighborhood between features within a document and between documents, while the neighborhood structure in observation 4 applies to a single document only. Also observation 1 can be called as a special case of observation 4, where the identity or repetition of a feature term is used to propagate labels.

Observation 5 : Bottom-up label propagation Unknown document

Algorithm 1 Backoff Inference Procedure(F_q,O_q)

Input: L_t, L_u, K_b
Output: Label of (F_q,O_q)
  $Label \leftarrow UNK$
  $Label_l \leftarrow 0$ $//count\ of\ neighbors\ with\ label\ l\ , whose\ labels\ are\ not\ committed$
  $CLabel_l \leftarrow 0$ $//count\ of\ neighbors\ with\ label\ l\ , whose\ labels\ are\ committed$
  $Confidence \leftarrow Low$
  if (F_q, O_q) in L_t AND committed **then**
    $Label \leftarrow L_{lt}((F_q, O_q))$
    $Confidence \leftarrow High$
  else
    $F_k \leftarrow$ Neighbors(F_q,K_b)
    for (F_k,O_k) in L_t **do**
      if O_q==O_k **then**
        if (F_k,O_k) is committed **then**
          $CLabel_l \leftarrow CLabel_l + L_{lt}((F_k, O_k))$
        else
          $Label_l \leftarrow Label_l + L_{lt}((F_k, O_k))$
        end if
      end if
    end for
    if $CLabel_l$ is NOT zero vector **then**
      $Label \leftarrow argmax_l(CLabel_l)$
      $Confidence \leftarrow High$
    else
      $Label \leftarrow argmax_l(Label_l)$
    end if
  end if
  if $Label == UNK$ **then**
    $Label \leftarrow L_{lu}(O_q)$
  end if
  Return Label, Confidence

level sentiment label is disambiguated using known sentence level labels. This follows from the intuition that a product or overall topic of discussion should carry the same sentiment as majority of its features. For example, if every aspect of a camera has positive feedback, then its obvious that the camera should also have positive feedback. Observations 3 and 5 define the inter-dependency between finer and coarser levels. The formulation given in section 3 is rewritten incorporating the above mentioned observations. It is stated as follows: unknown sentence level sentiment for s_j is given as a function of tuple level labels

$$\Omega(s_j) = Y_{ts}(\Omega(F_j, O_j)), \forall(F_j, O_j) \text{ where } (F_j, O_j)\epsilon s_j$$

where tuple level label is predicted using backoff inferencing procedure and tuples' labels with similar features in the same review document (label smoothing). The neighbors of the tuple t_j within a document are referred to as its local context, denoted by $Context(t_j)$. Since the backoff model relies on M to build both tuple based and unigram based classification, tuple level label $\Omega(t_j)$ is defined as a function Y_c of M and labels of $Context(t_j)$.

$$\Omega(t_j) = Y_c(M, \Omega(Context(t_j)))$$

Note that this $Context$ given by neighbors is different from context specific issues discussed earlier. And the document label is a function of sentence level labels, as given in section 3.

3.2 Collective Classification Framework

Given: Semi-supervised environment where the datapoints are partially labeled at both coarse and fine grained levels -document and sentence level respectively.
Target: Propagate labels and make M fully labeled at both coarse and fine grained levels.
Procedure: A document is seen as a function of sentence level labels, and a sentence can in turn be seen as a function of tuple level labels. So the entire corpus can be posed an undirected graph (V, E), where E is the set of edges and nodes in V corresponds to different entities - document, sentence and tuples. A tuple has target features and sentiment terms, and with the domain knowledge base a neighborhood structure is formed within a document and between documents as mentioned above. Since not all the nodes are labeled, the goal is to utilize the information available in the overall graph structure, and fully label the nodes. Example neighborhood structure for two documents D_1 and D_2 is given in Figure 1, we do not explain the example much since the idea is to show the graph structure. Classical learning algorithms train a classifier on the labeled datapoints, and use it to classify unlabeled datapoints. But they ignore the valuable neighborhood structure available. Collective classification is a framework which combines both classical learning method and neighborhood structure based classification in an iterative procedure [11]. One of the common inferencing procedures is relaxation labeling, which is a popular technique in image processing algorithms. The intuition behind the method is that pixel's probability to be assigned a label increases, given that its neighbors are assigned the same label. A similar intuition is employed here.

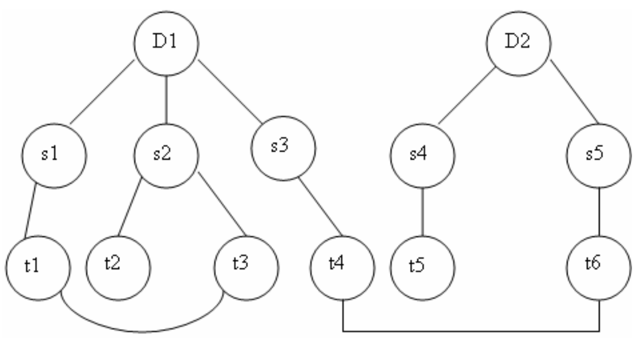

Figure 1. Example showing neighborhood for two documents

The notations for the iterative procedure are re-established and simplified as follows
l - Sentiment label space, which is $\{+, -\}$
$N_i(s)$ - Number of subjective sentences in the document D_i
$N_{il}(s)$ - Number of subjective sentences with label l in D_i
$N_j(t)$ - Number of tuples in the subjective sentence s_j
$N_{jl}(t)$ - Number of tuples in the subjective sentence s_j with label l
L_u - Lexicon with labeled sentiment terms from the lexicon W, which are relearned for the domain using M. L_u has the structure with 5 elements $[O, Flag, Label, c_+, c_-]$, where O denotes the sentiment term, Flag has commit and not-commit status which means whether the class label is committed or not. Label gives the class label, c_+ and c_- denote the count of occurrence of tuple in positive and negative contexts during the iteration i.e., number of times classified as positive and negative respectively. The querying that was mentioned in backoff model, $L_{lu}(O)$ which denotes the label returned

using L_u for the sentiment term O is done as follows.

$$If(Flag == commit) \rightarrow Return\ Label$$
$$Elseif(Flag == uncommit) \rightarrow Return\ argmax_l(c_l)$$

L_t - Lexicon with labeled tuples initialized using 'pros and cons' section in M. L_t has the structure with 6 elements $[F, O, Flag, Label, c_+, c_-]$, where F denotes the feature term, and other 5 tuples are similar to those in L_u. The querying procedure in backoff model and the way counts are updated during iterative procedure is similar to what is done for L_u.
$\Omega(t)$ - Sentiment label of a tuple
$(\Omega(t), Confidence)$ - Pair denotes sentiment label of a tuple and Confidence of inference given by backoff inference procedure
$\Omega(s)$ - Sentiment label of a sentence is defined as the function of labels of constituent tuples. The function used in this work is $maxlabel(x)$, where x is a collection of labels. $maxlabel(x)$ returns the label which occurs predominantly in the input x, i.e., a sentence is labeled with the maximum occurring label of its tuples as given below

$$\Omega(s_j) = maxlabel(\Omega(F_j, O_j)), \forall (F_j, O_j)$$
$$where\ (F_j, O_j) \epsilon s_j, \Omega(F_j, O_j) \epsilon l.$$

Note that any function can be used in place of $maxlabel$, for instance a linear classification model. Since the focus of this task is to show how the proposed model improves the accuracy of any classical learning algorithm and not to propose a new fine-grained or coarse-grained classifier, the simplest classification function is chosen.
$\Omega(D)$ - Sentiment label of a document is given by bottom up label propagation of the constituent sentences. $maxlabel$ is the function employed again, thus giving the following formulation

$$\Omega(D) = maxlabel(\Omega(s_i)), \forall s_i \text{ where } s_i \epsilon D$$

γ_{tli} - Number of neighboring tuples for a tuple t in a document D_i, with the sentiment label l.

3.3 Multi Grain Iterative Classification Algorithm

Multi grain iterative classification is a joint model that helps in predicting the unknown sentiment of sentences and documents to get completely labeled dataset, and also acquire more evidence for unigrams and tuples to update the lexicon L_u and L_t.
Initialization : L_u and L_t are initialized. Only subjective sentences in a document are taken for analysis, and the presence of adjectives is taken as an indicator for subjectivity [10]. Since the iterative procedure is shown to have same performance for any arbitrary ordering of nodes [11, 12], we choose an arbitrary ordering for document nodes and the natural order of occurrence within a document for sentences. Iterative procedure is given in Algorithm 2

Algorithm 2 Iterative procedure

```
repeat
  for Document D_i in M do
    for Sentence s_j in D_i do
      for Tuple t_k in s_j that is NOT committed do
        (Ω(t_k), Confidence) ← Backoff Model(t_k)
        if Confidence == High OR F_k is labeled in D_i then
          Propagate and commit labels, L_t ← L_t ∪ (t_k, Ω(t_k))
        end if
      end for
    end for
    repeat
      for Sentence s_j in D_i do
        for Tuple t_k in s_j that is NOT committed do
          Ω(t_k) ← argmax_l γ_tli
        end for
      end for
    until Convergence of labels
    Update counts in L_u, L_t
    for Sentence s_j in D_i do
      P(l|s_j) = N_jl(t) / N_j(t)
      Ω(s_j) = argmax_l P(l|s_j)
    end for
    P(l|D_i) = N_il(s) / N_i(s)
    Ω(D_i) = argmax_l P(l|D_i)
  end for
until Convergence
Commit the current labels in L_u, L_t
```

4 Experimental results

The review articles are taken from websites like CNET, Epinions and Edmunds which contain Automobile reviews. CNET's and Epinions' articles have both pros, cons section and document level label. Edmunds' test drives articles contain pros, cons section only and not document level label. 100 articles were chosen arbitrarily from each website , thus forming a dataset of 300 articles. The class distribution of document labels is — 120 positive and 80 negative documents among 200 documents which have document level label, and the remaining 100 have unknown labels. We do not use the tuple based approach using the 'pros and cons' phrases as a baseline approach, since Kaji and Kitsuregawa [4] have shown that the method has high precision but low recall. We define the baseline approaches below
Trivial Baseline : For any document D_i, which has document level label, propagate the label top-down such that all unknown labels of the sentences are labeled with the document's label. It is given as

$$\Omega(s_j) = \Omega(D_i), \forall s_j \text{ where } s_j \epsilon D_i$$

It was tested on 200 labeled documents among the 300 input documents. Since other baseline approaches and the proposed model are tested on 300 documents, the results of trivial baseline approach is presented separately. It can be seen as unrestricted top-down label propagation. The results are given in Table 1.

Class	P	R	F1
+	0.313	0.272	0.291
-	0.247	0.214	0.229

Table 1. Trivial classifier

Lexical classifier at document level : Using the words in W, classify documents as positive or negative depending on the count. Let C_p be the count of positive terms in the document, C_n be the count of negative terms in the document, then the documents are classified using $argmax_i(C_i)$.

Lexical classifier at sentence level : Using the words in W, classify sentences as positive or negative depending on the count using a method similar to the one mentioned above for document level.

Proposed model : The multi grain collective classification model is applied on the documents, after initializing L_u and L_t. Here we describe briefly the knowledge base used in this work. It has the domain terms list built using collection of review articles. Noun, noun phrases are considered as domain terms, and are put in partial hierarchy using Wordnet, since Wordnet is not complete with all the domain terms and jargon. The remaining terms are inserted at some node in the hierarchy based on distributional similarity computed using PMI of the second-order co-occurrence vector. The results comparing the methods for sentence and document classification is given in Table 2 and Table 3. P denotes precision, R denotes recall and F1 denotes F measure, which is the harmonic mean of precision and recall. The experimental evaluation is based on comparison of predicted labels against human labeled documents and sentences. The following are the observations of the results

Class	P	R	F1	Method
+	0.381	0.334	0.355	Lexical classifier
-	0.278	0.245	0.260	Lexical classifier
+	0.73	0.61	0.664	Proposed model
-	0.63	0.56	0.593	Proposed model

Table 2. Sentence level classification

Class	P	R	F1	Method
+	0.55	0.39	0.456	Lexical classifier
-	0.225	0.30	0.257	Lexical classifier
+	0.78	0.734	0.756	Proposed model
-	0.56	0.498	0.527	Proposed model

Table 3. Document level classification

- The main contribution of the proposed model is the use of the neighborhood structure for tuples within and between documents, by which the accuracy of any ordinary classifier can be improved. The classifier chosen in this work is an ordinary lexical classifier which takes the maximum labels of tuples. The cases where sentence level classification failed in the proposed model are those that needed other contextual evidences to be taken care of. An example case is given below,

  "The question is whether the car itself is as good as the wrapper it comes in"

  So, it indicates that with a better classifier (state of the art) in place, the additional strength provided by the framework should enable better performance.
- From the results where the proposed framework outperforms the baseline classifier, it is clear that it can improve any local classifier's performance when used in this framework. Though the final results obtained are on par with state of art methods [6, 13], it has to be noted that this method is not a fully labeled approach and uses only a small amount of labeled data at sentence level.

5 Conclusion and Future Work

In this paper, we have proposed a multi grain collective classification framework which uses partially labeled data at document and sentence level, and converts it into a fully labeled dataset. Though its transductive, the unigram and tuple lexicon learned in the iterative procedure can be used for inductive labeling on testset. The key contributions of the work include the following a) collective classification framework for multi grain sentiment analysis which requires very less labeled data, and improves the performance of any local classifier used in the iterative procedure b) utilizes the sentiment term lexicon and domain knowledge base to handle sparsity and improve recall without compromising precision. Future work is given as follows, a) In this work 0-1 neighborhood is used, where all the neighbors of a node are considered equally important, while practical knowledge about any domain shows that not all neighboring nodes would influence equally. For example "novelty of story" influences "quality of movie" the most in a movie review. So, potentials of the edges must be automatically learned and used in the iterative procedure. b)In this work a prebuilt domain knowledge base was used. In order to handle scaling domains, an automated way to build the domain knowledge base must be investigated.

ACKNOWLEDGEMENTS

This work was partially funded by GM R&D PO 910000121410.

REFERENCES

[1] Peter D. Turney. 2002. *Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews*, In Proc. of ACL.

[2] V. Sindhwani and P. Melville. 2008. *Document-word co-regularization for semi-supervised sentiment analysis*, In Proc. ICDM

[3] Peter D. Turney and Michael L. Littman. 2003. *Measuring praise and criticism: Inference of semantic orientation from association*, ACM Transactions on Information Systems.

[4] Nobuhiro Kaji and Masaru Kitsuregawa. 2007. *Building lexicon for Sentiment Analysis from Massive Collection of HTML Documents*, In Proc. EMNLP.

[5] B. Pang, L. Lee, and S. Vaithyanathan. 2002. *Thumbs up? sentiment classification using machine learning techniques*, In Proc. EMNLP.

[6] Melville, P., Gryc, W., and Lawrence, R. D. 2009. *Sentiment analysis of blogs by combining lexical knowledge with text classification*, In Proc. SIGKDD.

[7] McDonald, Ryan , Hannan Kerry , Neylon Tyler , Wells Mike , Reynar Jeffrey C. 2007. *Structured Models for Fine-to-Coarse Sentiment Analysis*, In Proc. ACL.

[8] B. Pang and L. Lee. 2004. *A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts*, In Proc. ACL.

[9] Somasundaran, S., Namata, G., Wiebe, J., and Getoor, L. 2009. *Supervised and unsupervised methods in employing discourse relations for improving opinion polarity classification.*, In Proc. EMNLP.

[10] Bing Liu. 2006. *Web Data Mining*, Springer.

[11] Neville, J. and Jensen, D. 2000. *Iterative Classification in Relational Data*, In Proc. AAAI

[12] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor *Collective Classification in Network Data*, Technical Report CS-TR-4905 and UMIACS-TR-2008-04

[13] Qiu, G., Liu, B., Bu, J., and Chen, C. 2009. *Expanding domain sentiment lexicon through double propagation*, In Proc. IJCAI