Mining Comparative Sentences and Relations

Nitin Jindal and Bing Liu

Department of Computer Science
University of Illinois at Chicago
851 South Morgan Street, Chicago, IL 60607-7053
{njindal, liub}@cs.uic.edu

Abstract

This paper studies a text mining problem, *comparative sentence mining*. A comparative sentence expresses an ordering relation between two sets of entities with respect to some common features. For example, the comparative sentence "*Canon's optics are better than those of Sony and Nikon*" expresses the comparative relation: (*better*, {*optics*}, {*Canon*}, {*Sony*, *Nikon*}). Given a set of evaluative texts on the Web, e.g., reviews, forum postings, and news articles, the task of comparative sentence mining is (1) to identify comparative sentences from the texts and (2) to extract comparative relations from the identified comparative sentences. This problem has many applications. For example, a product manufacturer wants to know customer opinions of its products in comparison with those of its competitors. In this paper, we propose two novel techniques based on two new types of sequential rules to perform the tasks. Experimental evaluation has been conducted using different types of evaluative texts from the Web. Results show that our techniques are very promising.

Introduction

One of the most important ways of evaluating an entity or event is to directly compare it with a similar entity or event. The objective of this work is to extract and to analyze comparative sentences in evaluative texts on the Web, e.g., customer reviews, forum discussions, and blogs. This task has many important applications. For example, after a new product is launched, the manufacturer of the product wants to know consumer opinions on how the product compares with those of its competitors. Extracting such information can help businesses in its marketing and product benchmarking efforts.

In recent years, there was a growing interest in analyzing evaluative texts on the Web (e.g., Pang, Lee & Vaithyanathan, 2002; Turney, 2002; Wilson, Yu & Hatzivassiloglou 2003; Wiebe & Hwa 2004; Hu & Liu, 2004; Popescu & Etzioni 2005; Carenini, Ng & Zwart 2005; Riloff & Wiebe 2005). The main focus has been on sentiment classification and opinion extraction (positive or negative comments on an entity or event). Comparisons are related to but also different from sentiments and opinions, which are subjective. Comparisons can be subjective or objective. For example, a typical opinion sentence is "*The picture quality of camera x is great*" A subjective comparison is "*the picture quality of camera x is better than that of camera y*." An objective comparison is "*car x is 2 feet longer than car y*". We can see that comparative sentences use different language constructs from typical opinion sentences (although the first comparative sentence above is also an opinion). In this paper, we study the problem of comparative sentence mining. It has two tasks:

1. Given a set of evaluative texts, identify comparative sentences from them, and classify the identified comparative sentences into different types (or classes).
2. Extract comparative relations from the identified sentences. This involves the extraction of entities and their features that are being compared, and comparative keywords. The relation is expressed with

   (<relationWord>, <features>, <entityS1>, <entityS2>)

   For example, we have the comparative sentence "*Canon's optics is better than those of Sony and Nikon.*" The extracted relation is:

   (better, {optics}, {Canon}, {Sony, Nikon})

Both tasks are very challenging. Although we see that the above sentences all contain some indicators i.e., "better", "long*er*", many sentences that contain such words are not comparatives, e.g., "*I cannot agree with you more*". The second step is a difficult information extraction problem.

For the first task, we present an approach that integrates *class sequential rules* (CSR) and *naïve Bayesian classification* to perform the task. This task is studied in detail in (Jindal & Liu 2006). We include it for completeness. For the second task, a new type of rules called *label sequential rules* (LSR) is proposed for extraction. Our results show that LSRs outperform Conditional Random Fields (CRF) (Lafferty, McCallum & Pereira 2001), which is perhaps the most effective extraction method so far (Mooney & Bunescu 2005).

Experimental evaluation has been conducted using Web evaluative texts, including consumer reviews, forum postings and news articles. The results show that the proposed methods are very promising. In summary, this paper makes the following two contributions:

1. It proposes the study of comparative sentence mining. To the best of our knowledge, there is no reported study

on this problem. Although linguists have investigated the semantics of comparative constructs, their work is mainly for human understanding and is not directly applicable to our classification and extraction tasks.

2. It proposes two new methods to identify comparative sentences and to extract comparative relations from these sentences based on two new types of rules, class sequential rules and label sequential rules.

Related Work

Researchers in linguistics have studied the syntax and semantics of comparative constructs for a long time (e.g., Moltmann 1997; Doran *et al*. 1994; Kennedy 2005). However, they have not examined computational methods for extracting comparative sentences and relations.

In text mining, we found no direct work on comparative sentence mining. The most related works are sentiment classification and opinion extraction, which as we pointed out are different from our work. Sentiment classification classifies opinion texts or sentences as positive or negative. Pang, Lee and Vaithyanathan (2002) examined several learning algorithms for sentiment classification of movie reviews. Turney (2002) proposed a method based on mutual information between document phrases and the words "excellent" and "poor" to find indicative words for sentiment classification. Dave, Lawrence, Pennock (2003) proposed another method. In (Hu and Liu 2004), some methods were proposed to extract opinions from customer reviews, i.e., identifying product features commented on by customers and determining whether the opinions are positive or negative. (Popescu & Etzioni 2005) and (Carenini, Ng & Zwart 2005) explore the issues further. Other related work on sentiment classification and opinion extraction includes (Morinaga *et al*. 2002; Yu & Hatzivassiloglou 2003; Yi *et al* 2003; Wilson, Wiebe & Hwa 2004; Kim & Hovy 2004; Riloff & Wiebe 2005). However, none of them extract or analyze comparisons.

Our work is also related to information extraction. Mooney & Bunescu (2005) gave a good survey and comparison of existing methods. Conditional random field (CRF) (Lafferty, McCallum & Pereira 2001) is perhaps the best method so far. We are not aware of any existing work specifically for extraction of comparative relations. We show that the LSR method outperforms CRF for our task.

Comparative Sentence Mining Problem

In linguistics, comparatives are based on specialized morphemes, *more*/*most*, *-er*/*-est*, *less*/*least* and *as*, for the purpose of establishing orderings of superiority, inferiority and equality, and *than* and *as* for making a 'standard' against which an entity is compared.

The coverage in linguistics is however limited because in practice many comparisons do not use these keywords, such as user preferences (e.g., "*I prefer Intel to AMD*") and comparatives expressed using other words (e.g., *lead*, *beat*, etc). We consider them in this work. Broadly speaking, comparatives can be classified into two main types: *gradable* and *non-gradable*. Gradable comparatives are based on such relationships as *greater or less than*, *equal to*, and *greater or less than all others*. Non-gradable comparatives express implicit comparisons (e.g., "*Toyota has GPS, but Nissan does not have*"). In this paper, we focus on *gradable* comparatives. Such a comparative sentence expresses an explicit ordering between entities.

Although linguists have studied both syntax and semantics of comparatives (Doran *et al* 1994; Moltmann 1997; Kennedy 2005), their work is mainly for human understanding and it provides limited help to our computational tasks of classification and extraction.

Part-of-Speech (POS) Tagging

We now give an introduction to part-of-speech (POS) tagging as it is useful to our subsequent discussion and also the proposed techniques. In grammar, part-of-speech of a word is a linguistic category defined by its syntactic or morphological behavior. Common POS categories are: noun, verb, adjective, adverb, pronoun, preposition, conjunction and interjection. Then there are many categories which arise from different forms of these categories. In this work, we use Brill's Tagger (Brill 1992). Important POS tags to this work and their categories are:

NN: Noun, *NNP*: Proper Noun, *PRP:* Pronoun, *VBZ*: Verb, present tense, 3rd person singular, *JJR*: Comparative Adjective, *JJS*: Superlative Adjective, *RBR*: Comparative Adverb, *RBS*: Superlative Adverb.

Although *JJR*, *JJS*, *RBR*, and *RBS* tags represent comparatives, many sentences containing such tags are not comparisons. Many sentences that do not contain any of these tags may be comparisons. Thus, we cannot solely use these tags to identify comparative sentences.

Problem Definition

In this paper we study the problem based on sentences. Thus we also state the problem based on sentences.

Definition (**entity and feature**): An *entity* is the name of a person, a product brand, a company, a location, etc, under comparison in a comparative sentence. A *feature* is a part or property of the entity that is being compared.

Types of Comparatives

1) *Non-Equal Gradable*: Relations of the type *greater* or *less than* that express a total ordering of some entities with regard to certain features. This type also includes user preferences.
2) *Equative*: Relations of the type *equal to* that state two entities as equal with respect to some features.
3) *Superlative*: Relations of the type *greater* or *less than all others* that rank one entity over *all* others.
4) *Non-Gradable:* Sentences which compare features of two or more entities, but do not explicitly grade them.

The first three types of comparative are called *gradable comparatives*. This work only focuses on these three types. For simplicity, from now on we use *comparative sentences* and *gradable comparative sentences* interchangeably. Note that in a long comparative sentence, there can be multiple

relations separated by delimiters such as commas (",") and conjunctions such as "and" and "but".

Definition (**comparative relation**): A *comparative relation* captures the essence of a comparative sentence and is represented with the following:

(relationWord, features, entityS1, entityS2, type)

where *relationWord*: The keyword used to express a comparative relation in a sentence.

features: a set of features being compared.

*entityS*1 and *entityS*2: Sets of entities being compared. Entities in entityS1 appear to the left of the relation word and entities in entityS2 appear to the right of the relation word.

type: *non-equal gradable*, *equative* or *superlative*, (not mention earlier for easy understanding).

Each of the sets can be empty. We do not claim that this relation representation captures everything about comparative sentences. Our experiences show that it covers a large portion of practically useful comparisons.

Our objective: Given a collection of evaluative texts, (1) identify the three types of comparative sentences, and (2) extract comparative relations from the sentences.

Two Types of Sequential Rules

We now start to present the proposed techniques, which are based on two types of sequential rules. Mining of such rules is related to mining of sequential patterns (SPM) (Agrawal and Srikant 1994). Given a set of input sequences, SPM finds all subsequences (called *sequential patterns*) that satisfy a user-specified minimum support threshold. Below, we first explain some notations, and then define the two new types of rules, *Class sequential rules* (CSR) used in classification of sentences, and *label sequential rules* (LSR) used in relation item extraction. For more details about these types of rules and their mining algorithms, please see (Liu 2006).

Let $I = \{i_1, i_2, ..., i_n\}$ be a set of items. A *sequence* is an ordered list of itemsets. An *itemset X* is a non-empty set of items. We denote a sequence s by $\langle a_1 a_2 ... a_r \rangle$, where a_i is an itemset, also called an *element* of s. We denote an element of a sequence by $\{x_1, x_2, ..., x_k\}$, where x_j is an item. An item can occur only once in an element of a sequence, but can occur multiple times in different elements. A sequence $s_1 = \langle a_1 a_2 ... a_r \rangle$ is a *subsequence* of another sequence $s_2 = \langle b_1 b_2 \ldots b_m \rangle$ or s_2 is a *supersequence* of s_1, if there exist integers $1 \le j_1 < j_2 < ... < j_{r-1} \le j_r$ such that $a_1 \subseteq b_{j_1}$, $a_2 \subseteq b_{j_2}$, ..., $a_r \subseteq b_{j_r}$. We also say that s_2 *contains* s_1.

Example 1: we have $I = \{1, 2, 3, 4, 5, 6, 7\}$. The sequence $\langle \{3\}\{4, 5\} \rangle$ is contained in $\langle \{6\}\{3, 7\}\{4, 5, 6\} \rangle$ because $\{3\} \subseteq \{3, 7\}$ and $\{4, 5\} \subseteq \{4, 5, 6\}$. However, $\langle \{3, 6\} \rangle$ is not contained in $\langle \{3\}\{6\} \rangle$ or vice versa.

Class Sequential Rules

Let S be a set of data sequences. Each sequence is labeled with a class y. Let Y be the set of all classes, $I \cap Y = \emptyset$. Thus, the input data D for mining is represented with $D = \{(s_1, y_1), (s_2, y_2), ..., (s_n, y_n)\}$, where s_i is a sequence and $y_i \in Y$ is its class. A *class sequential rule* (CSR) is an implication of the form

$$X \rightarrow y, \text{ where } X \text{ is a sequence, and } y \in Y.$$

A data instance (s_i, y_i) in D is said to *cover* the CSR if X is a subsequence of s_i. A data instance (s_i, y_i) is said to *satisfy* a CSR if X is a subsequence of s_i and $y_i = y$. The *support* (sup) of the rule is the fraction of total instances in D that satisfies the rule. The *confidence* (conf) of the rule is the proportion of instances in D that covers the rule also satisfies the rule.

Given a labeled sequence data set D, a minimum support (*minsup*) and a minimum confidence (*minconf*) threshold, CSR mining finds all class sequential rules in D.

Label Sequential Rules

A *label sequential rule* (LSR) is of the following form,

$$X \rightarrow Y,$$

where Y is a sequence and X is a sequence produced from Y by replacing some of its items with wildcards. A wildcard, denoted by a '*', matches any item. The definitions of support and confidence are similar to those above. The input data is a set of sequences, called *data sequences*.

Example 2: Table 1 gives a sequence database with 5 sequences and the minimum support of 30% and minimum confidence of 30%. We can find the following rule,

$\langle \{1\}\{3\}\{*, *\} \rangle \rightarrow \langle \{1\}\{3\}\{7, 8\} \rangle$ [sup = 2/5, conf = 3/4]

Data sequences 1, 2, and 4 contain $\langle \{1\}\{3\}\{*, *\} \rangle$, and data sequences 1 and 2 contain $\langle \{1\}\{3\}\{7, 8\} \rangle$.

	Data Sequence
1	$\langle \{1\}\{3\}\{5\}\{7, 8, 9\} \rangle$
2	$\langle \{1\}\{3\}\{6\}\{7, 8\} \rangle$
3	$\langle \{1, 6\}\{9\} \rangle$
4	$\langle \{1\}\{3\}\{5, 6\} \rangle$
5	$\langle \{1\}\{3\}\{4\} \rangle$

Table 1: An example sequence database

Such rules are useful because one may be interested in predicting some items in an input sequence, e.g., items 7 and 8 in the above example. The confidence of the rule gives us the estimated probability that the two '*''s are 7 and 8 given that an input sequence contains $\langle \{1\}\{3\}\{*, *\} \rangle$. This is exactly what we are interested in, i.e., to use such rules to predict and extract relation items.

Mining of the two types of rules is quite involved. Interested readers, please refer to (Liu 2006) for details.

Identify Gradable Comparative Sentences

We now introduce the integrated approach of CSRs and learning to identify gradable comparative sentences.

For gradable comparatives, it is easy to use a set of keywords to identify almost all comparative sentences, i.e., with a very high recall (98%). However, the precision is low (32% according to our data set). We thus designed the following 2-step strategy for learning:

- Use the keywords to filter out those sentences that are unlikely to be comparisons. The remaining set of sentences *R* forms the candidate comparative sentences.
- Work on *R* to improve the precision through supervised learning, i.e., to classify the sentences in *R* into *comparative* and *non-comparative sentences*.

Keywords: Apart from morphemes such as *–er*, *-est*, *more*, and *less*, there are many other indicative words for comparisons, e.g., *beat*, *exceed*, *outperform*, etc. We have compiled a list of 79 such keywords. Those words with POS tags of JJR, RBR, JJS and RBS are also good indicators. However, we do not use each individual raw word as a keyword. Instead, we use their POS tags, i.e., JJR, RBR, JJS and RBS, as four keywords only. There are 4 exceptions. *more*, *less*, *most*, and *least* are treated as individual keywords because their usages are diverse and we want to catch their individual usage patterns. All together, we have 83 keywords. The set is by no means complete. As the project progresses, it will be expanded.

Sequence data preparation: We use the keywords as *pivots* to form the sequence data set. For each sentence that contains at least one keyword, we use the keyword and the items that are within a radius of *r* to form a sequence. We used a radius of 3 for optimal results. For each keyword, we combine the actual keyword and its POS tag to form a single item. The reason is that some keywords have multiple POS tags depending upon their uses, which can be important in determining whether a sentence is a comparative sentence or not. For other words in the sentence, we simply use their POS tags. Finally, we attach a manually labeled class to each sequence, "*comparative*" or "*non-comparative*".

Example 3: Consider the comparative sentence "*this/DT camera/NN has/VBZ significantly/RB more/JJR noise/NN at/IN iso/NN 100/CD than/IN the/DT nikon/NN 4500/CD.*" It has the keyword *more*. The final sequence put in the database for rule generation is:

⟨{NN}{VBZ}{RB}{moreJJR}{NN}{IN}{NN}⟩ *comparative*

CSR generation: Based on the sequence data we generate CSRs. In rule generation, for rules involving different keywords, different minimum supports are used because some keywords are very frequent and some are very rare. The mechanism behind this is quite involved (see (Jindal & Liu 2006; Liu 2006) for details). In addition to automatically generated rules, we also have 13 rules compiled manually, which are more complex and difficult to find by current mining techniques.

Learning using a naïve Bayesian classifier: We used the naïve Bayesian (NB) model to build the final classifier using CSRs and manual rules as attributes. A new database for NB learning is created. The attribute set is the set of all sequences on the left-hand-side of the rules. The classes are not used. The idea is that we use rules to identify attributes that are predictive of the classes. A rule's predictability is indicated by its confidence. Each sentence forms an instance in the training data. If the sentence has a particular pattern in the attribute set, the corresponding attribute value is 1 and 0 otherwise. Using the resulting data, it is easy to perform NB learning. More details about this step can be found in (Jindal and Liu 2006)

Extract Comparative Relations

This step performs two tasks:

1) Classification of the comparative sentences obtained from the last step into one of the three types or classes, *non-equal gradable*, *equative*, and *superlative*. Note that it is possible to combine this task with the previous step to build a 4-class classifier. However, the results were poor. Breaking the task into two performed better.
2) Extraction of features, entities and relation keywords.

Classify Comparative Sentences into Three Types

For this classification task, the keywords alone are already sufficient. That is, we use the set of keywords as the attribute set for machine learning. The classes are *non-equal gradable*, *equative* and *superlative*. If the sentence has a particular keyword in the attribute set, the corresponding attribute value is 1, and otherwise is 0. The SVM learner gives the best result in this case.

Extraction of Relation Items

We now discuss how to extract relation entries/items. Label sequential rules are used for this task. In this work, we make the following assumptions:

1) There is only one relation in a sentence. In practice, this is violated only in a very small number of cases.
2) Entities and features are nouns (includes nouns, plural nouns and proper nouns) and pronouns. This covers a majority of cases. However, a feature can sometimes be a noun used in its verb form or some action described as a verb (e.g., "Intel costs more", "costs" is a verb and a feature). We leave this to our future work.

Sequence data generation: We create a sequence database for mining as follows: Since we are interested in predicting and extracting items representing *entityS1* (denoted by *$ES1*), *entityS2* (denoted by *$ES2*), and *features* (denoted by *$FT*), *non-entity-feature* (denoded by *$NEF*), which are called *labels*, we first manually label such words in each sentence in the training data. Note that *non-entity-feature* represents a noun or pronoun that is not a feature or entity. For example, in the sentence "*Intel/NNP is/VBZ better/JJR than/IN amd/NN*", the proper noun "Intel" is labeled with *$ES1*, and the noun "amd" is labeled with *$ES2*. We use the four labels as pivots to generate sequence data. For every occurrence of a label in a sentence, a separate sequence is created and put in the sequence database. We used a radius of 4 for optimal results. The following words are also added to keep track of the distance between two items in a generated pattern:

1) Distance words = {*l*1, *l*2, *l*3, *l*4, *r*1, *r*2, *r*3, *r*4}, where "*li*" means distance of *i* to the left of the pivot. "*ri*" means the distance of *i* to the right of pivot.
2) Special words *#start* and *#end* are used to mark the start and the end of a sentence.

Example 4: The comparative sentence "*Canon/NNP has/VBZ better/JJR optics/NNS than/IN Nikon/NNP*" has *$ES1* "*Canon*", *$FT* "*optics*" and *$ES2* "*Nikon*". The three sequences corresponding to two entities and one feature put in the database are (note that the keyword 'than' is merged with its POS tag to form a single item):

⟨{#start}{*l*1}{**$ES1,** NNP}{*r*1}{has, VBZ}{r2}{better, JJR} {*r*3}{$FT, NNS}{r4}{thanIN}⟩

⟨{#start}{*l*4}{$ES1, NNP}{*l*3}{has, VBZ}{*l*2}{better, JJR} {*l*1} {**$FT**, NNS}{*r*1}{thanIN}{*r*2} {entityS2, NNP}{*r*3} {#end}⟩

⟨{has,VBZ}{*l*4}{better, JJR}{*l*3}{$FT, NNS}{*l*2} {thanIN} {*l*1}{**$ES2,** NNP}{*r*1}{#end}⟩

LSR generation: After the sequence database is built, we first generate all frequence sequences (minsup = 1%)). We only keep sequences that have at least one label, and the label has no POS tag associated with it. For example, we keep ⟨{**$ES1**}{VBZ}⟩, and discard ⟨{**$ES1**, NN}{VBZ}⟩ and ⟨{NN}{VBZ}⟩. Only the remaining sequences are used to generate LSRs.

Note that an entity or a feature can be a noun (NN), a proper noun (NNP), a plural noun (NNS) or a pronoun (PRP). We now replace each occurrence of label {c_i} in a sequence with {c_i, NN}, {c_i, NNP}, {c_i, NNS} and {c_i, PRP} and generate 4 rules with separate confidences and supports. This specialization enables us to generate rules with much lower supports. If there is more than one class in a rule, this substitution is done for one label at a time. So, k labels in a sequence will give 4*k new rules.

For example, the sequence, ⟨{**$ES1**}{VBZ}⟩, generates the following 4 CSRs

⟨{*, NN}{VBZ}⟩ → ⟨{**$ES1**, NN}{VBZ}⟩
⟨{*, NNP}{VBZ}⟩ → ⟨{**$ES1**, NNP}{VBZ}⟩
⟨{*, NNS}{VBZ}⟩ → ⟨{**$ES1**, NNS}{VBZ}⟩
⟨{*, PRP}{VBZ}⟩ → ⟨{**$ES1**, PRP}{VBZ}⟩

The supports and confidences of the new rules are computed by scaning the training data once.

We then select a set of rules to be used for extraction based on the sequential covering algorithm in machine learning. However, in our case, it is more complex because each sentence contains multiple labels. If a rule covers a label in a sentence, the sentence cannot be removed. We need the training sentences to build the sequential cover. The algorithm works as follows:

1) Select the LSR rule with the highest confidence. Replace the matched elements in the sentences that satisfy the rule with the labels ({c_i}) in the rule.
2) Recalculate the confidence of each remaining rule based on the modified data from step 1.
3) Repeat step 1 and 2 until no rule left with confidence higher than the *minconf* value (we used 90%).

Example 5: We use an example to explain the algorithm. Consider the following sentence (converted as a sequence),

⟨{**$ES**1, NN}{is, VBZ}{preferred, JJ}{because, IN} {the, DT} {**$FT**, NN}{is, VBZ}{better, JJR}{thanIN}{**$ES2**, NN}⟩

Note: **$ES**1 is "*coke*", **$FT** is "*taste*" and **$ES2** is "*pepsi*"

Assume that we have the following three LSR rules,

R1: ⟨{*, NN}{VBZ}⟩ → ⟨{**$ES1**, NN}{VBZ}⟩
R2: ⟨{DT}{*, NN}⟩ → ⟨{DT}{**$FT**, NN}⟩
R3: ⟨{$FT}{VBZ}{JJR}{thanIN}{*, NN}⟩ →
⟨{$FT}{VBZ}{JJR}{thanIN}{**$ES2**, NN}⟩

with confidences 80%, 90% and 70% respectively. Let us apply R2 (as it has the highest confidence) to the sentence first and replace the occurrence of {$FT, *NN*} with {$FT}. The sentence then becomes,

⟨{**$ES**1, NN}{is, VBZ}{preferred, JJ}{because, IN} {the, DT} {**$FT**}{is, VBZ}{better, JJR}{thanIN} {**$ES2**, NN}⟩

The {$FT} label can now be used by rules that have label {$FT} in them, e.g., the third rule (R3). R1 will not match {$FT, NN} in the original sentence, and thus will not label it as {$ES1}. So, its confidence will increase as {$FT, NN} is not an entity. If R1 were applied first, {$FT, NN} would have been labeled {$ES1}, which causes an error. After several iterations we will be able to pick best rules which cover the data set.

The rules are then applied to match each comparative sentence in the test data to extract the components of the relation. The process is similar to the above sequential covering. The *relationWord* in the relation is the keyword that identifies the sentence as a comparative sentence.

Empirical Evaluation

This section evaluates our approaches. We first describe the data sets used and then present the experimental results.

Data Sets and Labeling

We collected data from disparate Web sources to represent different types of data. Our data consists of customer reviews, forum discussions and random news articles. The sentence distribution of each type is given in Table 2.

Non-equal	Equative	Superlative	Other sentences
285	110	169	2684

Table 2. Different types of sentences

Only 4% of the sentences had multiple comparisons in one sentence. The distribution of entities and features in the data set is given in Table 3. Nouns include different types.

Type	Total	Nouns	Pronouns	Neither
Entities S1	488	316 (65%)	92 (19%)	80 (16%)
Entities S2	300	228 (76%)	4 (1%)	68(23%)
Features	348	248 (71%)	0 (0%)	100 (29%)

Table 3. Distribution of entities and features in data

Not every sentence contained all three, i.e. entities S1, S2 and features. For superlatives, entityS2 is normally empty.

Labeling: The data sets were manually labeled by 2 human labelers. Conflicts were resolved through discussions.

Experimental results

We now give the precision, recall and F-score results. All the results were obtained through 5-fold cross validations.

Identifying gradable comparatives: NB using CSRs and

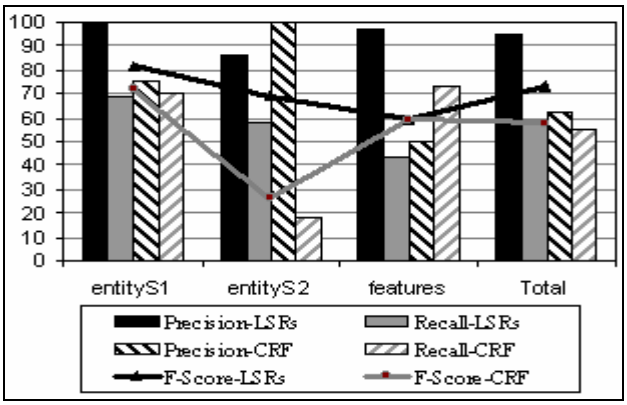

Figure 1. Precision-Recall and F-Score results of LSRs and CRF for extracting relation entries

manual rules as the attribute set gave a precision of 82% and a recall of 81% (F-score = 81%) for identification of gradable comparative sentences. We also tried various other techniques, e.g., SVM (Joachims 1999), CSR rules only, etc., but the results were all poorer. Due to space limitations, we are unable to give all the details. (Jindal & Liu 2006) has all the results using a larger dataset.

Classification into three different gradable types: For classification of only gradable comparatives, NB gave an accuracy of 87% and SVM gave an accuracy of 96%. When SVM was applied to sentences extracted from the previous step, the average F-score was 80%. Accuracy is not used as some sentences in the set are not comparisons.

Extraction of relations: Comparisons of precision, recall and F-score results of label sequential rules (LSRs) and conditional random fields (CRF) are given in Figure 1. We used the CRF system from Sarawagi (2004).

LSRs gave an overall F-score of 72%, while CRF gives an overall F-score of 58%. For entityS1, LSRs give 100% precision, because such entities have nice characteristics, e.g., occurring at the start of a sentence, before a verb phrase, etc. CRF performed reasonably well too, although not as good as LSRs. For entityS2, CRF performed poorly. Entities in entityS2 usually appear in the later part of the sentence and can be anywhere. LSRs covering a long range of items usually perform better than local contexts used in CRF. It is also interesting to note that LSRs gave consistently high precisions. The recalls of both systems were significantly affected by errors of the POS tagger.

Figure 1 does not include relation word extraction results since for such extraction we do not need rules. After classification, we simply find those keywords in the sentences, which give very accurate results:

Non-Equal Gradable:	Precision = 97%. Recall = 88%
Equative:	Precision = 93%. Recall = 91%
Superlative:	Precision = 96%. Recall = 89%

Using LSRs and keywords, we were able to extract 32% of complete relations and 32% of three-fourth relations (one of relation items was not extracted). Pronouns formed 16% of entityS1. In our current work, we do not perform pronoun resolution, which we plan to do in the future.

Conclusions and Future Work

This paper studied the new problem of identifying comparative sentences in evaluative texts, and extracting comparative relations from them. Two techniques were proposed to perform the tasks, based on class sequential rules and label sequential rules, which give us syntactic clues of comparative relations. Experimental results show that these methods are quite promising.

This work primarily used POS tags and keywords. In our future work, we also plan to explore other language features (e.g., named entities, dependency relationships of different constructs, etc) to improve the accuracy.

References

Agrawal, R. and Srikant, R. Mining sequential patterns, *ICDE'94*,

Brill, E. A Simple Rule-Based Part of Speech Tagger. *ANL* 1992.

Dave, K., Lawrence, S., and Pennock, D. Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews. *WWW'03*, 2003.

Doran, C., Egedi, D., Hockey, B. Srinivas, B & Zaidel, M. XTAG system-a wide coverage grammar for English. *COLING'1994*.

Carenini, G. Ng, R., and Zwart, E. Extracting knowledge from evaluative text. *ICKC'05*, 2005.

Hu, M., and Liu, B. Mining and summarizing customer reviews. *KDD'04*, 2004.

Joachims, T. Making large-scale SVM learning practical. *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf and C. Burges and A. Smola (ed.), 1999.

Jindal, N. and Liu, B. Identifying comparative sentences in text documents. *SIGIR-06*, 2006.

Kennedy, C. Comparatives, semantics of. *In Enclycopedia of Language and Linguistics*, Elsevier, 2005.

Kim, S and Hovy, E. Determining the sentiment of opinions. *COLING'04*, 2004.

Lafferty, J., McCallum, A., and Pereira, F. Conditional random fields: probabilistic models for segmenting and labeling or sequence data. *ICML'01*, 2001.

Liu, B. *Web Data Mining – Exploring hyperlinks, contents and usage data*. A forthcoming book. 2006/2007.

Moltmann, F., Coordination and Comparatives. Ph.D. dissertation. MIT, Cambridge Ma., 1987.

Mooney, R. J. and Bunescu, R. Mining Knowledge from Text Using Information Extraction. *SIGKDD Explorations*, 2005.

Morinaga, S., Yamanishi, K., Tateishi, K., and Fukushima, T. Mining Product Reputations on the Web. *KDD'02*.

Pang, B., Lee, L., and Vaithyanathan, S., Thumbs up? Sentiment classification using machine learning techniques. *EMNLP'02*.

Popescu, A-M and Etzioni. O. Extracting Product Features and Opinions from Reviews. *EMNLP-05,* 2005.

Riloff, E. and Wiebe, J. Learning extraction patterns for subjective expressions. *EMNLP'03*, 2003.

Sarawagi, S. CRF Project page, http://crf.sourceforge.net// 2004.

Turney, P. Thumbs Up or Thumbs Down? Semantic Orientation applied to Unsupervised Classification of Reviews. *ACL'02*.

Wilson, T., Wiebe, J., and Hwa, R. Just how mad are you? Finding strong and weak opinion clauses. *AAAI'04*.

Yi, J., Nasukawa, T., Bunescu, R., Niblack, W. Sentiment analyzer: Extracting sentiments about a given topic using natural language processing techniques. ICDM-2003.

Yu, H., and Hatzivassiloglou, V. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. *EMNLP'03*, 2003.