Two-Tier Performance Based Classification Model for Low Level NLP tasks

S Sameen Fatima R Krishnan

Abstract

An error in classification can occur due to an error of omission, statistically known as a false negative or an error of commission, statistically known as a false positive. In order to build a perfect classifier, the false negatives and false positives have to be zero. With this in mind, we propose a two-tier model for the classifier. The first tier will reduce false negatives to zero and pass the results to the second tier. The second tier will reduce false positives to zero. We demonstrate the working of this model for the task of classifying sentences in Hindi as passive formations. The first tier will consist of a simple pattern matching system for filtering out sentences with likely passive formations without committing errors of omission. This will reduce the size of the corpus considerably. The second tier will work on the reduced corpus and make a complete grammatical analysis of these filtered sentences in order to reduce the false positives to a zero. The Anusaraka System [Bharati 1995] is a very good example of such a system. This paper concentrates on building the first tier. A hill climbing algorithm is proposed, where the start state is a list of patterns commonly found in passive formations. Each step up the hill will update the list of patterns such that the next state will bring down the number of false negatives, thereby reducing errors of omission. The hill climbing algorithm terminates when the false negatives are zero.

Keywords : Natural Language Processing, Automated Language Processing

0. Motivation

In continuation of our effort to establish stylistic variation as a basis for genre-based text classification for English language [Fatima, 2001], research has been extended to the Indian languages. As Hindi is spoken by majority of the one billion Indian population, and as it is the official Indian language as well, the choice was restricted to Hindi language. While for English language, software is readily available to identify various linguistic features, to our knowledge no such readily off the shelf software is available in Hindi. Hence, before extrapolating our studies of English language texts for establishing stylistic variation to Hindi language texts, there was a need to build the necessary software. One of the features that directly and indirectly identifies style is the usage of passive sentences. The current work is a description of a model that can be used to classify sentences based on the voice as passive or active formations.

1. Passive Voice and Its Usage

Voice is that form of a verb which shows whether what is denoted by the subject does something (Active Voice) or has something done to it (Passive Voice). In the active voice, the agent (i.e., doer of the action) is made prominent, whereas in the passive voice, the object (i.e., person or thing acted upon) is made prominent [Wren 1936]. Although passive sentences are harder to comprehend than active sentences, they are sometimes needed.

With reference to automatic abstracting, Borko and Chatman [Borko, 1967] have advanced the view that it seems possible to make stylistic distinctions between informative and indicative abstracts. The informative abstract 'discusses the research' and the indicative abstract 'discusses the article which describes the research'. Distinctions in terms of form, in the usage of voice, tense, and the focus of the abstract have been used to differentiate between informative abstracts and indicative abstracts.

The passive voice in Hindi language is generally preferred [Mohanlal]

to emphasize the object of the sentence; as,

dhan paanii ki taraha bahaayaa jaa rahaa hei.

to avoid naming an unimportant character; as,

patra bhej diyaa gayaa hei.

To assert authority; as,

apraadhii ko pesh kiyaa jaae.

When the doer refers to a sabha, samaj or the government; as,

aaryasamaaja dwaaraa kii antarjaatiiya vivaah karaae jaate heiM.

In official language; as,

binaa TikeT safara karane vaaloM ko daNDit kiyaa jaayegaa.

to express inability to do something

mujha se khaanaa khaayaa nahii jaataa.

1.1 How can we identify passives ?

To identify passive sentences in Hindi, the following observations were made:

Two words in Hindi are popularly found in a passive formation namely ‘dwaara’ and ‘se’. For example:

yaha upanyaas muMshii premacanda dwaaraa likhaa gayaa hei.

mujhse kheera khaayii gayii.

Of the two ‘dwaara’ is more likely an indicative of a passive formation than ‘se’. This is because of the simple reason that ‘se’ can be used both in the sense of ‘by’ and ‘with’, whereas ‘dwaara’ is mostly used in the sense of ‘by’ only. Whenever the usage of ‘dwaara’ or ‘se’ is an indication of the instrumental case, then it is surely not an indication of a passive formation. For example:

caakuu se seba kaaTo.

mei dillii TRena dwaaraa jaa rahaa huuM.

Hence, ‘dwaara‘ and ’se’ by themselves are not an indication of passive voice.

In passive formations along with the head verb, the conjugation of the verb ‘jaana’ is used. For example: ‘jaana’ takes the form: ‘jaa saktaa’ or ‘jaayegaa’, where ‘saktaa’ and ‘gaa’ are auxiliary verbs or modals. The modals give information about gender, number and the tense (‘saktaa’ for future and ‘gaa’ for past). The modals may be

separated from the verb ‘jaa’, like ‘jaa saktaa/jaa saktii/jaa sakte/jaa rahaa/jaa rahii/jaa rahe’ is used in:

khaana khilaayaa jaa saktaa hei.

bacciiyaaM douDaayii jaa rahii thii.

kai khela khilaaye jaa sakte heiM.

or may combine with the verb ‘jaa’ like ‘jaaega/jaaegii/jaaenge’, ‘jaataa/jaatii/jaate’, ‘jaauuMgaa/jaauuMgii/ jaayenge’, is used in

khaana khilaayaa jaayegaa.

paaThshaalaa mein paDaayaa jaataa hei.

mei douDaayaa jaauuMgaa.

Passive formations also use the forms ’gayaa/gaii/gayii/gae/gaye’ of the verb ‘jaana’ as in:

mujhse kheera khaaii gayii /mujhse kheera khaaii gaii

kaii Teliphona kiye gaye./ kaii Teliphona kiye gae.

In addition to information consisting of the verb ending in modals, it can also be noted from the above examples that the vibhakti ‘yaa/yii/ye’ (which is the karma karak) is also used in passive formations. Collectively the vibhakti and the modals for a verb give information about tense, aspect and modality (TAM), and is, therefore, also called the TAM label. TAM labels for passive formations are purely syntactic and are determined from the combination of the verb (‘jaanaa’) with the modals and the vibhakti (‘yaa/yii/ye’).

The above discussion can be extrapolated to include negation. Negation is indicated by the word ‘nahii’ in between the vibhakti and the conjugation of the verb ‘jaanaa’. The following passive formations show the usage of negation:

mujhse khaayaa nahii gayaa

kii khela khilaaye nahii gaye

2. NLP approaches for Identifying Passive Formations

When considering a technology to support high-precision text classification, natural language processing (NLP) is one of the first things that comes to mind. Work done to date on NLP systems since early days have varied widely in their approach to analyzing texts. At one end of the spectrum were systems that processed a text using traditional NLP techniques. At the other extreme lie systems that use keyword/ pattern matching techniques and little or no linguistic analysis of the input text. [Cardie, 97]

The traditional approach to sentence analysis in natural language processing works on the assumption that a system must assign a complete constituent analysis to every sentence it encounters. The methods used to attempt this are drawn from mathematics, with context-free grammars playing a large role in assigning syntactic constituent structure. Using this approach, the following steps are taken for identifying passive formations:

? Tokenization: the input text is divided into sentences and words.

? Tagging: a dictionary or lexicon is looked up, and associated grammatical information (like parts of speech) is retrieved and used for tagging the words.

? Sentence-analysis: Noun groups and verb groups are identified. Verb groups are more important from the point of view of identifying passive formations.

The Anusaraka System described in Bharati, 1995 is a very good example of a complete system built on these principles. A sentence is first processed by a morphological analyzer (morph). The morph considers a word at a time, and for each word it checks whether the word is in the dictionary of indeclinable words. If found, it returns its grammatical features. It also uses the word paradigms to see whether the input word can be derived from a root and its paradigm. The output of morph is given as input to the local word grouper. Its main task is to group function words with the content words based on local information such as postposition markers that follow a noun, or auxiliary verbs following a main verb. This grouping (or case endings in case of inflectional languages), identifies vibhakti of nouns and verbs. The vibhakti of verbs is also called TAM (tense-aspect-modality) label. These TAM labels are used to identify passive formations.

One of the biggest challenges in natural language processing is the need to make NLP systems robust by providing it linguistic sophistication, while at the same time not trading off speed of processing. If a preprocessing step for filtering out likely passive formations precedes the complete analysis of the sentences, we would not have to grammatically process each and every sentence in the text. Hence, it was felt that if passive sentences can be mined for frequently occurring patterns, a simple pattern matching technique can be used for filtering likely passive sentences. This would reduce the corpus size. On the reduced corpus a complete analysis of the sentences can be carried out.

Currently, general pattern matching techniques have become the technique of choice for the extraction phase of an information extraction system [MUC-6 1995]. A number of researchers have investigated the use of corpus-based methods for learning information extraction patterns. The learning methods vary along a number of dimensions: the class of patterns learned, the training corpus required, the amount and type of human feedback required, the degree of preprocessing necessary, the background knowledge required, and the biases inherent in the learning algorithm [Cardie, 1997]

Sentence

"Witnesses conform that the twister occurred without warning at approximately 7:15 pm and destroyed two mobile homes."

Concept-Node Definition

Concept = Damage
Trigger = "destroyed"
Position = direct-object
Constraints = ((physical-object))
Enabling Condition = ((active voice))

Instantiated Concept Node

Damage = "Two mobile homes"

Figure 1: Concept node for extracting damage information

Information extraction is a subfield of natural language processing that is concerned with identifying predefined types of information from text. For example, an information extraction system designed for a terrorism domain might extract the names of perpetrators, victims, physical targets, weapons, dates, and locations of terrorist events. One of the earlier systems for acquiring extraction patterns was AUTOSLOG [Riloff 1996]. AUTOSLOG learns extraction patterns in the form of domain-specific concept nodes. It uses a small set of heuristic rules to decide what expression should activate the case frame, and from which syntactic constituent the slot should be filled. Figure 1 for example shows the concept node for extracting "two mobile homes" as damaged property, from a given sentence, using the keyword "destroyed" as a trigger.

4. Two-Tier Performance based Model for Classifiers

In order to build a true model of a classifier it should be bereft errors. An error is a misclassification: the classifier is presented a case, and it classifies the case incorrectly. Depending on the application, distinctions among different types of errors turn out to be important. For example, the error committed in diagnosing someone as healthy when one has a life-threatening illness (known as a false negative decision) is usually considered far more serious than the opposite type of error-of diagnosing someone as ill when one is in fact healthy (known as a false positive).

In order to distinguish between different types of errors, it is important to build a contingency matrix. The contingency matrix lists the correct classification against the predicted classification for each class. The number of correct predictions for each class falls along the diagonal of the matrix. All off-diagonal are the errors for a particular type of misclassification error. Table 1 is an example of such a matrix for two classes. The label Positive denotes the class of sentences with passive formations and the label Negative denotes the class of sentences with non-passive formations.

Performance of Classifier		**True labels**	
		Positive (Passive)	Negative (¬ Passive)
Predicted labels	Positive (Passive)	True positives (TP)	False Positives (FP)
	Negative (¬ Passive)	False negatives (FN)	True Negatives (TN)

Table 1: Sample contingency matrix for a two-class classification problem

A classic metric for reporting errors of omission and commission are sensitivity and specificity. Sensitivity is a measure of the errors due to commission and specificity is a measure of the errors due to omission. Using the notation in Table 1 sensitivity and specificity can be expressed as:

$$Sensitivity ? \frac{Truepositives}{Allpositives} ? \frac{TP}{TP ? FN} ? \text{Re}\, call$$

$$Specificity ? \frac{Truenegatives}{All\,\text{ln}\, egatives} ? \frac{TN}{TN ? FP}$$

$$\text{Pr}\, ecision ? \frac{Truepositives}{Allpredictedpositives} ? \frac{TP}{TP ? FP}$$

In the evaluation of information retrieval systems, the most widely used performance measures are *recall* and *precision*. However, S*ensitivity* and *specificity* are used to bring out the difference in the error types (errors of commission and errors of omission). *Sensitivity* is the accuracy among positive instances and *specificity* among negative. *Recall and precision* are mostly utilized in situations where *TP* is small when compared with *TN*. A perfect classifier can be described as one in which both false negatives and false positives are zero or in other words in which both the sensitivity and specificity would equal one [Baldi 2000].

Based on the above, a two-tier model is proposed in figure 2. The main goals of each tier are as follows:

Tier 1: This aims at reducing the false negatives to a zero. It achieves this by using a pattern matching filtering system.

Tier 2: This aims at reducing the false positives to a zero. It achieves this by using a complete natural language processing system.

At the end of the first tier, errors of omission are dangerous, however errors of commission can be tolerated. The second tier will take care of errors of commission. As Anusaraka [Bharati 1995] is available for complete grammatical analysis at Tier 2, this paper concentrates on building tier 1.

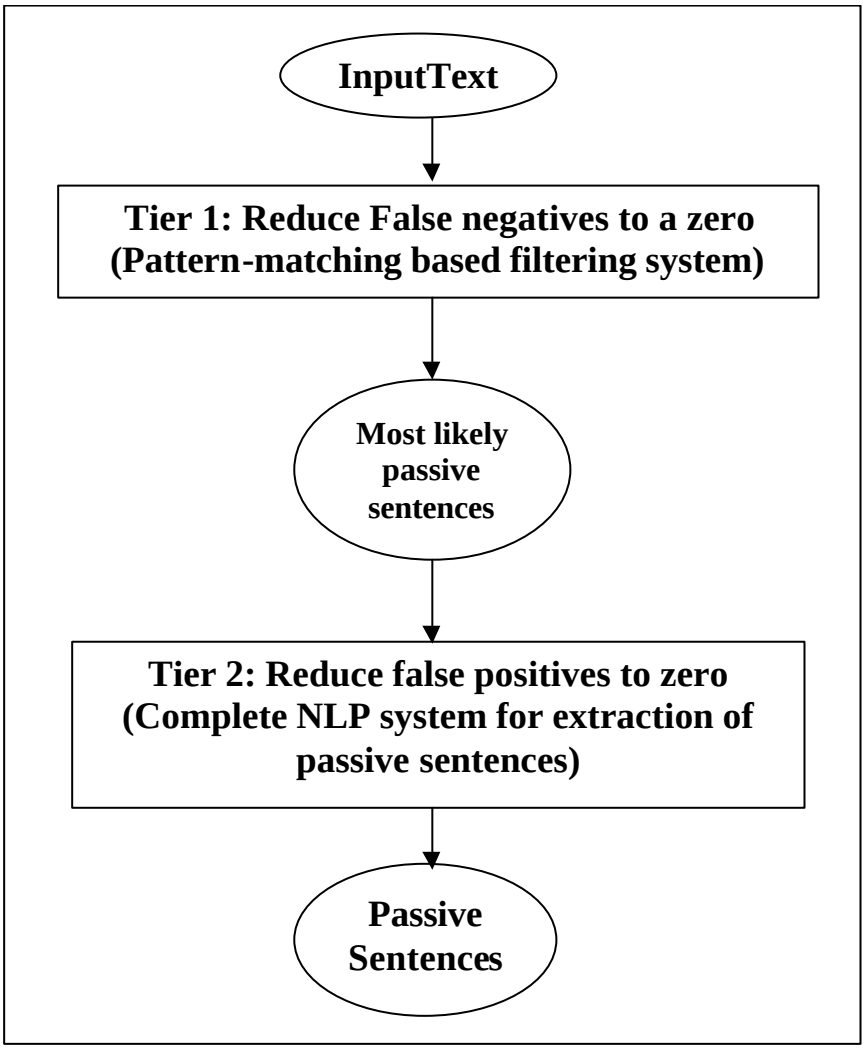

Figure 2: Two-tier performance based model for identifying passive sentences

4. Design of TIER 1

4.1 Hill Climbing Algorithm

In building tier 1, our focus was on the problem of distinguishing sentences as belonging to one class (passive) from another (not passive) – a binary supervised classification problem. An iterative improvement algorithm like hill climbing is used which improves sensitivity by decreasing the false negatives. It works in 2 phases:

Generate Phase: A list of patterns L+, commonly found in the Positive (passive) class of sentences was generated

Test phase: The performance of the list L+ was tested on a corpus of Hindi editorials, C. The number of passive sentences that are not filtered by the classifier were found out, which gave the number of false negatives, FN. If FN was zero we quit, otherwise we return to step 1, that is the Generate Phase

4.2 The State Space

The state space representation used in the hill climbing algorithm is described below:

State : A state corresponds to a list of patterns L+, for identifying sentences with passive formations.

Initial State Generation: Several passive sentences from common usage were taken as input to manually identify string patterns commonly found in passive sentences. This list of patterns in the initial state were generated with the help of a linguist, our background knowledge of Hindi and a corpus. The corpus was chosen as editorials of Hindi newspapers. The initial state was generated manually as a pre-tagged or pre-parsed training corpus for identifying passive formations was not available. Although appealing, the development of annotated corpora is still tedious and labor intensive [Srinivas 2001].

Next State Generation: The passive sentences in a corpus of editorials, C were filtered using list L+. A list of passive sentences not filtered by the help of the existing list L+ were identified. Each pattern in the list L+ was examined and compared against the sentences that were not filtered by the classifier. The list was refined with the help of feedback from linguists, from our background knowledge of Hindi and at times with the help of functions to identify the minimal common pattern between the patterns available in the list L+. As a result the existing patterns in the list were being deleted or updated, and new patterns were identified for inclusion wherever necessary in the list L+. The new list was the next state. The heuristic used was: decrease the number of false negatives FN.

Local Maxima: Each uphill move may involve a number of unsuccessful attempts (i.e., visits to nodes which increase the number of false negatives). A solution is called a local maximum if no uphill moves can be performed starting from the current state. If hill climbing reaches a local maximum, then we backtracked in search of a better solution in other areas of the state space. The cases not accounted for were again incorporated.

Goal State: The list of patterns for which the number of false negatives, FN was zero.

5. Experiments and Results

Our string pattern generation for identifying passive formations was guided by two main objectives: On the one hand, we give preference to the generation of short patterns (simplicity principle), and on the other hand, we attempt to cover every passive sentence by at least one pattern (comprehensive principle).

5.1 Generation of Goal State

After several iterations of the Generate and Test phase, two lists evolved: List L1 and List L2. Both the lists had zero false negatives, which was the aim of tier 1. However, list L2 had lesser false positives than list L1.

List L1 was a shorter list consisting of the string patterns shown in Table 2.

1. aa ga
2. aa jaa
3. ii ga
4. ii jaa
5. e ga
6. e jaa

Table 2: Goal State - List L1

A corpus consisting of 1626 sentences taken from editorials of Daily Hindi Milap and Vaartha were filtered based on the list L1. The results were analyzed and a contingency matrix shown in Table 3 was created

Performance of Filter		**True labels**	
		Passive	¬ Passive
PredictedLabels	Passive	337	61
	¬ Passive	0	1228

Table 3: Contingency matrix for the filter for classifying sentences as passive

$$Sensitivity ? \frac{337}{337 ? 0} ? 1$$

$$Specificity ? \frac{1228}{61 ? 1228} ? 0.95$$

As can be observed from Table 3, the number of false negatives are zero. This means that the list covered all patterns necessary for identifying passive formations, and hence sensitivity is one. Thus the filter at Tier 1 is perfect. However, there are 61 false positives, that is 61 sentences that were not passive were also filtered. This will be taken care of by Tier2. The number of sentences filtered (as likely passive sentences) are 398 (= 337 + 61), which will be fed to Tier 2. Tier 2 will perform a complete grammatical analysis on the 398 sentences filtered by Tier 1. At the end of tier 1 the size of the corpus reduced from 1626 to 398, which was considerable.

The second list was a larger list. The first four string patterns from List 1 were taken as it is. However, string patterns 5 and 6 from List 1 were replaced by string patterns 5 to 10 in List L2, with a view to improve specificity as shown in Table 4.

1. aa ga
2. aa jaa
3. ii ga
4. ii jaa
5. ie ga
6. ie jaa
7. ye ga
8. ye jaa
9. (dwaaraa Ú se) Ù (e ga)
10. (dwaaraa Ú se) Ù (e jaa)

Table 4: Goal State - List L2

The pattern strings numbered 1 to 8 in List L2 look for a single pattern match in the sentence. Example:

If the pattern “aa ga” is found in the sentence

then conclude it is a passive formation

However, the pattern strings 9 and 10 in Table 4 are looking for two patterns in a sentence. If both are available then it qualifies for a passive sentence, otherwise it is discarded. Example:

If “dwaaraa” and “e ga” are found in a sentence

then conclude it is a passive formation.

This was done in order to exclude sentences of the form given below from being classified as passive formations:

raama miThaii khaa*e ga*a.

It was found that the expanded list L2 shown in Table 4 decreases the false positives and thereby improves the specificity while not effecting the sensitivity.

Using the list L2, a contingency matrix shown in Table 5 was created

Performance of Filter		**True labels**	
		Passive	¬ Passive
PredictedLabels	Passive	337	13
	¬ Passive	0	1276

Table 5: Contingency matrix for the filter for classifying sentences as passive

$$Sensitivity ? \frac{337}{337 ? 0} ? 1$$

$$Specificity ? \frac{1276}{13 ? 1276} ? 0.99$$

As can be observed from Table 5, the number of false negatives are zero. This means that the list covered all patterns necessary for identifying passive formations, and hence sensitivity is one. Thus the filter at Tier 1 is perfect. Further, comparing the performance of list L2 (table 4) against list L1 (table 2) the following observations were made:

the number of false positives are reduced to 13 from 61, which increases the specificity from 0.95 to 0.99.

The number of sentences filtered (as likely passive sentences) are reduced to 350 (= 337 + 13) from 398. At the end of tier 1 according to List L2, the size of the corpus reduced from 1626 to 350, which is roughly one-fifth (21.5%) of its original size.

6. Conclusion

In-depth natural language processing (NLP) for high precision text classification is an expensive endeavor that can strain computational resources [Riloff 1994]. As an alternative to full-blown NLP, we have presented a 2-Tier Performance-Based Model for a Classifier. This model represents a compromise between pattern-matching techniques, at Tier 1, and in-depth natural language processing at Tier 2, so that the performance of the classifier is perfect. We evaluate the model for a low-level NLP task of classifying sentences as passive. We built Tier 1 using a Hill Climbing algorithm. At the end of Tier 1 the sensitivity was 1 (perfect) and the specificity was 0.99 (very nearly perfect) and the corpus size was reduced to 21.5%. Tier 2 can use a system like Anusaraka , which does full blown grammatical analysis of the sentences. The load on such a system will be minimal. It has to process just 21.5% of the corpus, and has to increase the specificity from 0.99 to 1. The results suggest that pattern-matching techniques can support high-precision classification for low level NLP tasks without straining computational resources.

7. References

1. P. Baldi, S. Brunak, Y. Chauvin, CAF Anderson, H Nielsen. Assessing the Accuracy of Prediction Algorithms for Classification: An Overview. Bioinformatics, 16 (5), 2000, pp 412 – 424.
2. Herald Borko. Automated Language Processing. John Wiley and Sons. 1967.
3. Claire Cardie. Empirical Methods in Information Extraction. AI Magazine. Winter 1997.
4. Sameen Fatima, R Krishnan. Stylistic Variation as a Basis for Genre-Based Text Classification. IETE Journal of Research. Vol 47, Nos 1&2, Jan – Apr 2001, pp 59 – 63.
5. Mohanlal D, Ashok B. Navyug hindi vyaakraN tathaa racnaa. Lakshmi Publications Pvt Ltd. New Delhi.
6. Proceedings of the Sixth Message understanding Conference (MUC-6). San Francisco, California. Morgan Kaufman, 1999.

7. Riloff E, Lehnert W. Information Extraction as A Basis for High-Precision Text Classification. ACM Transactions on Information Systems, Vol 12, No. 3, July 1994, pp 296 – 333.

8. Riloff E. Automatically Generating Extraction Patterns from Untagged Text. Proceedings of the 13th National Conference on AI. Menlo Park, Calif. AAAI. 1996, pp 1044 – 1049.

9. Bharati A, Chaitanya V, Sangal R. Natural Language Processing – A Paninian Perspective. Prentice Hall of India. 1995.

10. Srinivas B. Annotated and Unannotated Corpora in Natural Language Applications. Proceedings of the Workshop on Lexical Resources for Natural Language Processing, Language Technologies Research Centre, International Institute of Information Technology, Hyderabad, Jan 2001.

11. PC Wren, H Martin, NDV Prasada Rao. English Grammar and Composition. 1936.

About Authors

Dr. S Sameen Fatima is working in Department of Computer Science & Engineering, College of Engineering, Osmania University, Hyderabad, India.
E-mail : sameenf@rediffmail.com

Dr. R Krishnan is working in Advanced Data Processing Research Institute, Department of Space, Govt. of India, Secunderabad, India
E-mail : drrk@hotmail.com