A New Approach for HMM Based Chunking for Hindi

Term Project on
Speech and Natural Language Processing

BY

Ashish Tiwari
02CS3015
Department of Computer Science and Engineering,
Indian Institute of Technology,
Kharagpur
&
Arnab Sinha
02CS3022
Department of Computer Science and Engineering,
Indian Institute of Technology,
Kharagpur

Done under the guidance of

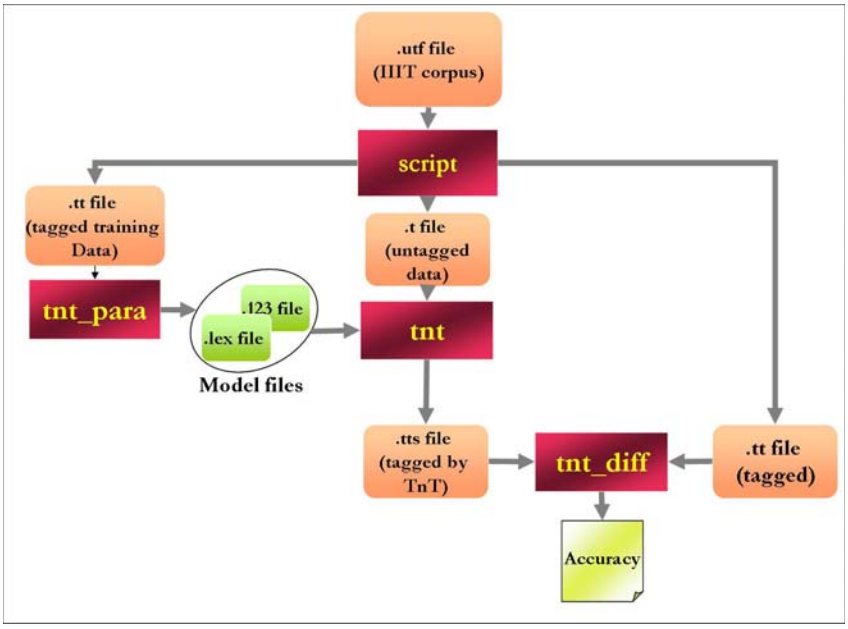
Dr. Sudeshna Sarkar
Department of Computer Science & Engineering,
Indian Institute of Technology,
Kharagpur

1. The Basic Idea

A robust chunker is an important component for various applications requiring natural language processing. Applications like *information extraction*, *machine translation*, and even *search* call for the application of chunkers.

A chunker or shallow parser identifies simple verb-phrases, noun-phrases, adjectival and adverbial phrases in running text. Chunkers may be built with hand-crafted linguistic rules. But that is time consuming and non-robust. In this work, our primary focus is to build a robust chunker using a simple technique using a popular Hidden Markov Model-based chunker.

The following flow-chart shows the working of the HMM based tagger – TnT.



We want to exploit the tool-support of the TnT for the chunking of Hindi. The input to '*tnt-para*' is (token, POS-tag) for generating the language model. By simply feeding (POS-tag, Chunk-tag) to this utility we can generate the model for chunking of Hindi.
Since TnT is implemented up to a tri-gram model, we use a second order HMM (k = 2) in our study.

We expect to get better results, if we can train the tool with the contextual knowledge. To do that, we can either *manipulate* the *tokens* or the *chunk-tags*. We plan to train the tool with the following tag schemes,
1. (word-token, Chunk-tag)
2. (POS-tag, Chunk-tag)
3. (word_POS-tag, Chunk-tag)
4. (POS-tag_word, Chunk-tag)

The order of word and the POS-tag is important since, TnT module use suffix information while carrying out smoothing of transition and emission probabilities for sparse data.
The other technique is to manipulate the chunk-tag. We can use a chunk tag-set where the chunk-tags are '*Chunk-tag:POS-tag*'. Thereby we augment the chunk tag-set, and make it more realistic one. If time permits, we might look at 3-tag (i.e. chunk tag-set = {STRT, CNT, END}) and 4-tag scheme (chunk tag-set = {STRT, CNT, END, STRT_END}), where, STRT = this token lies in start of the chunk, CNT = this token lies in middle of the chunk, END = this token lies in end of the chunk, and STRT_END = this token lies in a chunk of its own.

2. Experiments

((ashish))	((arnab ke pIche))	((bajar meM))	((gaya .))
ashish	arnab of behind	market in	went .
NN	NN PREP PREP	NN PREP	VB SYM
STRT	STRT CNT CNT	STRT CNT	STRT CNT

Ex:2-tag scheme

In this example, the chunk tags considered are STRT and CNT where STRT indicates that the new chunk starts at the token which is assigned this tag and CNT indicated that the token which is assigned this tag is inside the chunk. We refer to this as 2-tag scheme. Under second-order HMM, the prediction of chunk tag at i^{th} token is conditional on the only two previous chunk tags. Thus in the example, the fact that the chunk terminates at the word pIche (behind) with the POS tag PREP is not captured in tagging the token bajar (market). This example implies that the chunk-tag boundary is independent of the POS-tag. But, this assumption restricts the prediction of subsequent chunk tags. To overcome this limitation in using TnT, we experimented with additional chunk tags.

Therefore we experimented with 2-tag,3-tag,4tag scheme
.

((ashish))	((arnab ke pIche))	((bajar meM))	((gaya .))
ashish	arnab of behind	market in	went .
NN	NN PREP PREP	NN PREP	VB SYM
STRT	STRT CNT END	STRT END	STRT END

Ex:3-tag scheme

((ashish))	((arnab ke pIche))	((bajar meM))	((gaya .))
ashish	arnab of behind	market in	went .
NN	NN PREP PREP	NN PREP	VB SYM
STRT_END	STRT CNT END	STRT END	STRT END

Ex:4-tag scheme

Now we manipulated the tokens, using four schemes. Adding pos-tag information to input tokens, we get the following.

1. (word-token, Chunk-tag)

((ashish))	((arnab ke pIche))	((bajar meM))	((gaya .))
ashish	arnab of behind	market in	went .
NN	NN PREP PREP	NN PREP	VB SYM
STRT	STRT CNT CNT	STRT CNT	STRT CNT

Ex: Word as token and 2tag chunking

2. (POS-tag, Chunk-tag)

((ashish))	((arnab ke pIche))	((bajar meM))	((gaya .))
NN	NN PREP PREP	NN PREP	VB SYM
STRT	STRT CNT CNT	STRT CNT	STRT CNT

Ex:POS-tag as token and 2tag chunking

3. (word_POS-tag, Chunk-tag)

((ashish))	((arnab ke pIche))	((bajar meM))	((gaya .))
ashish	arnab of behind	market in	went .
ashish _NN	arnab _NN of _PREP behind _PREP	market _NN in _PREP	went _VB SYM
STRT	STRT CNT CNT	STRT CNT	STRT CNT

Ex: **Word_POS-tag as token and 2tag chunking**

4. (POS-tag_word, Chunk-tag)

((ashish))	((arnab ke pIche))	((bajar meM))	((gaya .))
ashish	arnab of behind	market in	went .
NN_ ashish	NN_ arnab PREP_ of PREP_ behind	NN_ market PREP_ in	VB_ went SYM_.
STRT	STRT CNT CNT	STRT CNT	STRT CNT

Ex: **POS_Word-tag as token and 2tag chunking**

Next, we attempted modification of chunk tags using contextual information. The new output tags considered were a combination of POS tags and chunk tags using any one of the chunk tag schemes discussed in the earlier section. The new format of chunk tags considered was POS:ChunkTag, which is illustrated for 2-tag scheme in the example below.

((ashish))	((arnab ke pIche))	((bajar meM))	((gaya .))
ashish	arnab of behind	market in	went .
NN	NN PREP PREP	NN PREP	VB SYM
NN :STRT	NN :STRT PREP :CNT PREP :CNT	NN:STRT PREP:CNT	VB :STRT SYM :CNT

Ex: **Word as token and POS:2tag chunking**

We think that adding the pos information to chunking and pos information to input tokens should improve the results.

3. Work Done

The following tables show the work done.

	Set 1		Set 2		Set 3		Set 4		Set 5		Avg	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk)	80.77	94.74	76.28	93.74	84.10	94.01	88.25	86.60	79.11	93.41	81.7	92.5
(POS, Chunk)	90.86	94.42	81.84	94.25	93.76	94.82	94.18	83.82	83.3	94.29	88.79	92.32
(word_POS, Chunk)	86.86	96.94	80.79	95.45	91.32	96.16	92.29	89.08	84.85	95.49	87.22	94.62
(POS_word, Chunk)	82.01	96.94	76.24	95.45	85.28	96.16	88.62	89.08	79.73	95.49	82.38	94.62

Table 1: Result for '*phrase-tags*' as chunk-tags

	Set 1		Set 2		Set 3		Set 4		Set 5		Avg	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk:POS)	67.66	95.47	68.85	98.03	74.72	95.61	79.78	92.65	70.94	97.74	72.39	95.9
(POS, Chunk:POS)	91.28	98.78	83.11	98.91	93.23	98.46	94.09	95.33	85.43	99.09	89.43	98.11
(word_POS, Chunk:POS)	84.52	98.90	81.97	99.60	91.85	98.75	91.59	97.35	85.64	99.67	87.11	98.85
(POS_word, Chunk:POS)	71.85	98.90	72.22	99.60	78.03	98.75	84.23	97.35	74.41	99.67	76.15	98.85

Table 2: Result for '*phrase-tags:POS-tags*' as chunk-tags

Here, the phrase-tags (e.g. NP, VP etc) have been used as the chunk-tags in Table 1. While in Table 2, the chunk-tags are **`'<phrase_tag>:<POS-tag>'.`** We have tested on 5 different samples both the *Precision* and *Recall*. From the experimental-data, we can see that the recall has significantly increased in the Table 2. The precision has not changed

significantly primarily because the paucity of training data. Also, it is interesting to observer that recall for both **word_POS**, and **POS_word** are same in both the tables. Because when the same data is used for training and testing, the order of the word and POS-tag does not affect the emission-probabilities.
As it can be inferred from Table 1,that using simple chunk phrase tags if we ignore the word information and use pos-tags the recall and precision increase .In case of pos-tag token and phrase chunk scheme the accuracy is highest 89.43% which has improved from 72.39% in case of only words as token. It must be also noted that there is not much difference between word_pos token and pos chunk scheme results precision 87.11% and 89.43% respectively, also recall is higher in case of word_pos. We believe that due to paucity of data results might be slightly skewed, on larger size of corpus word_pos tag should do marginally better than pos-tag only scheme.

Also it is to be noted that adding the pos-tag information to chunking tags in Table-2,the precision increases marginally in case of POS-tags as tokens from 87.11% to 89.43%, reaffirming our belief that adding pos information to chunking tags might increase overall knowledge base of the underlying system resulting in better results.

Our next plan was to implement the cases of 3-tag ({STRT, CNT, STP}) and 4-tag ({STRT, CNT, STP, STRT_STP}) scheme already mentioned earlier in the initial discussion, and report the precision and recall of identifying the chunk-boundaries.

	Set 1		Set 2		Set 3		Set 4		Set 5		Avg	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk)	60.52	87.4	73.34	89.81	78.35	81.64	74.4	81.86	75.54	89.74	72.43	86.09
(POS, Chunk)	64.72	82.06	63.13	82.82	78.27	73.86	61.07	73.45	72.38	82.82	67.71	83.2
(word_POS, Chunk)	65.68	87.97	76.68	92.39	80.31	85.31	79.58	85.43	78.94	92.43	76.24	88.71
(POS_word, Chunk)	63.56	87.97	74.73	92.39	79.33	85.31	77.86	85.43	77.19	92.43	74.53	88.71

Table 3: For 2-tag chunks ({STRT, CNT})

	Set 1		Set 2		Set 3		Set 4		Set 5		Avg	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rc
(word, Chunk:POS)	58.97	94.74	67.27	97.71	69.26	95.52	76.67	89.35	69.53	97.27	68.34	94.91
(POS, Chunk:POS)	76.21	97.64	84.6	97.67	91.4	98.02	92.52	93.58	87.05	97.85	86.35	96.95
(word_POS, Chunk:POS)	73.42	98.74	81.81	98.84	88.01	98.52	88.77	95.09	83.99	98.69	83.20	97.97
(POS_word, Chunk:POS)	63.43	98.74	71.82	98.84	72.81	98.52	81.64	95.09	73.34	98.69	72.61	97.98

Table 4: For 2-tag chunks ({STRT:POS, CNT:POS})

	Set 1		Set 2		Set 3		Set 4		Set 5		Avg	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk)	45.81	85.12	57.56	88.21	64.53	78	62.58	76.64	62.86	88.14	58.70	83.22
(POS, Chunk)	55.38	73.13	55.58	72.85	68.12	60.49	57.32	64.86	64.72	73.18	60.20	70.0
(word_POS, Chunk)	56.08	87.97	63.56	91.41	69.79	83.18	69.33	82.65	67.33	92.03	65.22	87.50
(POS_word, Chunk)	51.64	87.97	60.81	91.41	64.74	83.18	66.89	82.65	64.58	92.03	62.00	87.44

Table 5: For 3-tag chunks ({STRT, CNT, STP})

	Set 1		Set 2		Set 3		Set 4		Set 5		Avg	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk:POS)	50.74	95.11	55.99	97.74	59.89	94.94	67.36	87.98	59.12	96.87	58.58	94.52
(POS, Chunk:POS)	72.49	97.27	76.8	96.36	86.79	97.44	87.43	91.34	79.84	95.82	80.67	95.64
(word_POS, Chunk:POS)	65.88	98.61	70.92	99.16	78.56	98.02	80.01	94.06	72.59	98.65	73.61	97.71
(POS_word, Chunk:POS)	55.2	98.61	60.56	99.16	63.11	98.02	71.78	94.06	62.42	98.65	62.62	97.70

Table 6: For 3-tag chunks ({STRT:POS, CNT:POS, STP:POS})

	Set 1		Set 2		Set 3		Set 4		Set 5		Avg	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk)	46.25	86.47	54.93	90.36	63.68	81.47	62.41	77.91	60.36	89.92	57.53	85.22
(POS, Chunk)	59.07	76.03	63.35	76.82	69.43	67.24	60.66	66.71	67.23	76.53	63.95	72.68
(word_POS, Chunk)	55.17	90.38	61.86	93.27	71.46	86.06	68.49	83.2	66.13	93.09	64.63	89.20
(POS_word, Chunk)	50.99	90.38	57.81	93.27	63.72	86.06	65.81	83.2	62.52	93.09	60.15	89.19

Table 7: For 4-tag chunks ({STRT, CNT, STP, STRT_STP})

	Set 1		Set 2		Set 3		Set 4		Set 5		Avg	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk:POS)	48.93	94.99	53.85	97.93	59.52	94.82	64.97	88.87	56.27	97.45	56.71	94.81
(POS, Chunk:POS)	71.2	96.66	74.3	96.07	85.77	96.97	86.06	91.72	77.67	95.71	79.00	95.42
(word_POS, Chunk:POS)	63.33	98.37	67.68	99.2	77.17	97.56	77.16	95.12	70.29	98.84	71.12	97.82
(POS_word, Chunk:POS)	52.65	98.37	57.75	99.2	62.17	97.56	69.1	95.12	59.53	98.84	60.24	97.82

Table 8: For 4-tag chunks ({STRT:POS, CNT:POS, STP:POS, STRT_STP:POS})

From the table pairs Table-3 and Table 4, Table-5 and Table-6, Table-7 and Table-8 it is very clear that adding the pos-context information to chunk boundaries results in higher precision and greater recall, in general. It may be noted that precision and recall seems to decrease in value as we increase the number of tags but this comparison is not justified. To compare across the different schemes, the output were converted to the reduced chunk

tag sets which are denoted by 4→3, 4→2 and 3→2 in the table. This ensures that the measurement metric is the same no matter which tagging scheme is used, thus allowing us to compare across the tagging schemes. It should be noted that converting from the 4 tag set to 3 or 2 tags results in no loss in information. This is because it is trivial to convert the 2-tag set to the corresponding 4-tag set and vice-versa.

For 3-tag → 2-tag

	Set 1		Set 2		Set 3		Set 4		Set 5		Avg	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk)	58.89	90.62	72.6	92.65	78.39	85.54	76.26	84.85	76.43	92.61	72.61	89.26
(POS, Chunk)	64.34	82.59	66.93	83.08	78.11	69.77	67.68	76.23	75.51	83.11	70.51	78.95
(word_POS, Chunk)	64.95	92.21	75.66	94.58	81.13	88.42	79.87	88.53	79.08	94.98	76.12	91.75
(POS_word, Chunk)	62.63	92.21	74.67	94.58	78.48	88.42	78.67	88.53	77.81	94.98	74.45	91.75

Table 9: From 3-tag scheme to 2-tag (without the POS-tag suffix)

	Set 1		Set 2		Set 3		Set 4		Set 5		Avg	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk:POS)	67.12	98.37	76.43	99.31	81.66	98.14	84.2	94.5	80.52	98.91	78.0	97.8
(POS, Chunk:POS)	75.82	98.53	85.12	98.14	91.81	98.75	92.06	94.64	87.46	97.85	86.45	97.58
(word_POS, Chunk:POS)	71.48	99.31	80.7	99.67	86.75	98.98	87.43	96.26	82.79	99.34	81.83	98.71
(POS_word, Chunk:POS)	68.21	99.31	77.67	99.67	82.8	98.98	84.84	96.26	80.49	99.34	78.8	98.71

Table10: From 3-tag scheme to 2-tag (with the POS-tag suffix)

For 4-tag → 2-tag

	Set 1		Set 2		Set 3		Set 4		Set 5		Avg	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk)	60.36	92.54	72.81	94.83	78.76	90.08	78.24	87.63	77.26	94.58	73.45	91.93
(POS, Chunk)	68.98	86.79	77.64	86.72	81.08	78.96	74.08	79.32	79.59	86.35	76.28	83.62
(word_POS, Chunk)	66.74	95.03	77.33	96.62	83.77	92.64	82.16	90.55	80.80	96.51	78.15	94.27
(POS_word, Chunk)	64.15	95.03	74.88	96.62	78.96	92.64	80.48	90.55	78.74	96.51	75.44	94.32

Table 11: From 4-tag scheme to 2-tag (without the POS-tag suffix)

	Set 1		Set 2		Set 3		Set 4		Set 5		Avg	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk:POS)	66.25	98.17	76.43	99.38	81.66	98.2	83.39	95.36	79.29	99.24	77.4	98.07
(POS, Chunk:POS)	75.46	98.29	84.53	98.03	91.72	98.63	92.38	95.23	87.05	97.82	86.23	97.6
(word_POS, Chunk:POS)	70.5	99.18	79.99	99.71	86.38	98.92	86.41	97.29	82.48	99.49	82.1	98.92
(POS_word, Chunk:POS)	67.15	99.18	77.33	99.71	82.33	98.82	83.82	97.29	79.22	99.49	77.97	98.89

Table 12: From 4-tag scheme to 2-tag (with the POS-tag suffix)

For 4-tag → 3-tag

	Set 1		Set 2		Set 3		Set 4		Set 5		Avg	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk)	46.25	87.04	54.93	90.54	63.68	82.92	62.41	79.49	60.36	90.14	57.55	86.02
(POS, Chunk)	59.07	76.97	63.35	77.04	69.43	68.66	60.66	68.09	67.23	76.75	63.95	73.40
(word_POS, Chunk)	55.17	90.83	61.86	93.41	71.46	87.29	68.49	84.58	66.13	93.23	64.68	89.86
(POS_word, Chunk)	50.99	90.83	57.81	93.41	63.72	87.29	65.81	84.58	62.52	93.23	60.20	89.87

Table 13: From 4-tag scheme to 3-tag (without the POS-tag suffix)

	Set 1		Set 2		Set 3		Set 4		Set 5		Avg	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk:POS)	58.4	96.74	63.72	98.69	70.97	96.65	74.72	92.24	67.5	98.51	67.1	96.6
(POS, Chunk:POS)	71.97	96.78	76.18	96.11	86.42	97.27	87.72	92.17	79.05	95.74	80.2	95.6
(word_POS, Chunk:POS)	64.59	98.45	70.06	99.2	78.11	97.82	79.17	95.36	72.28	98.84	72.9	97.8
(POS_word, Chunk:POS)	59.64	98.45	65.67	99.2	71.79	97.82	75.33	95.36	67.85	98.84	68.1	97.9

Table 14: From 4-tag scheme to 3-tag (with the POS-tag suffix)

From table pairs
Table 5(3tag) & Table 9(3tag->2tag),
Table 6(3colontag) & Table 10(3colontag->2tag) ,
Table 7(4tag) & Table 13(4tag->3tag),
Table 7(4tag) & Table 11(4tag->2tag),
Table 8(4colontag) & Table 12(4colontag->3tag),
Table 8(4colontag) & Table 14(4colontag->2tag),it is clear that even though the information content in the 3 different chunk tag representations is the same, using higher tag scheme for training and then later converting back to 2-tags results in a significant improvement in the precision of the tagger

4. Summary

	Phr-Tag		Phr-Tag with pos information	
	Prec	Prec	Prec	Rec
(word, Chunk)	81.7	72.39	72.39	86.09
(POS, Chunk)	88.79	89.43	89.43	83.2
(word_POS, Chunk)	87.22	87.33	87.33	88.71
(POS_word, Chunk)	82.38	76.15	76.15	88.71

Table 14.1: Summary of phrase tags (Colon vs Non-Colon)

Table 14.1 reaffirms the fact that adding pos tag info to chunking tags improves the precision and accuracy of tagger. Also note that word_pos token might perform better than simple pos-tags as tokens on a larger data set as the difference between % precision is not much in both cases, and it seems to be the logical way of thinking as adding pos-tag information to word tags should increase knowledge base and hence improve results.

4.1 Chunk Boundary identification

	2-Tag		3-Tag		4-Tag	
	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk)	72.43	86.09	58.70	83.22	57.53	85.22
(POS, Chunk)	67.71	83.2	60.20	70.0	63.95	72.68
(word_POS, Chunk)	76.24	88.71	65.22	87.50	64.63	89.20
(POS_word, Chunk)	74.53	88.71	62.00	87.44	60.15	89.19

Table 15: Summary of normal chunk-tags

Comparing Columns 1,2 of Table 16 with Colum1 of Table-15 we see improvement in 2 tag scheme .It also is evident that word_pos token scheme and conversion of 4tag→2tag has the highest accuracy and precision.

	3→2 Tag		4→2-Tag		4→3-Tag	
	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk)	72.61	89.26	73.45	91.93	57.55	86.02
(POS, Chunk)	70.51	78.95	76.28	83.62	63.95	73.40
(word_POS, Chunk)	76.12	91.75	78.15	94.27	64.68	89.86
(POS_word, Chunk)	74.45	91.75	75.44	94.32	60.20	89.87

Table 16: Summary of Tag-scheme shifts(without the POS suffixes)

	2-Tag		3-Tag		4-Tag	
	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk:POS)	68.34	94.91	58.58	94.52	56.71	94.81
(POS, Chunk:POS)	86.01	96.95	80.67	95.64	79.00	95.42
(word_POS, Chunk:POS)	83.20	97.97	73.61	97.71	71.12	97.82
(POS_word, Chunk:POS)	72.61	97.98	62.62	97.70	60.24	97.82

Table 17: Summary of '*chunk-tag:POS-tag*'

	3→2 Tag		4→2-Tag		4→3-Tag	
	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk:POS)	78.0	97.8	77.4	98.07	67.1	96.6
(POS, Chunk:POS)	86.1	97.58	86.23	97.6	80.2	95.6
(word_POS, Chunk:POS)	81.83	98.71	82.1	98.92	72.9	97.8
(POS_word, Chunk:POS)	78.8	98.71	77.97	98.89	68.1	97.9

Table 18: Summary of Tag-scheme shifts(with the POS suffixes)

Comparing Columns 1,2 of Table 18 with Column1 of Table-17 we see improvement in 2 tag scheme .It also is evident that word_pos token scheme and conversion of 4tag→2tag has the highest accuracy and precision. Also comparing 4tag→2tag column in Table 16 and Table-18 we see huge improvement in precision and recall.

4.2 Labeling the Chunks

Three schemes are adopted. In the *first* scheme,
token: <word>_<POS-tag>
label: <2-tag chunk boundary>:POS-tag:<chunk label> (if this is the first token of the chunk.)
<2-tag chunk boundary>:POS-tag (otherwise)

In the *second* scheme,
token: <word>_<POS-tag>

label: <2-tag chunk boundary>:POS-tag:<chunk label> (for all tokens)

In the *third* scheme,
token: <word>_<POS-tag>
label: <2-tag chunk boundary>:POS-tag:<chunk label> (if this is the last token of the chunk.)
<2-tag chunk boundary>:POS-tag (otherwise)

	Set 1		Set 2		Set 3		Set 4		Set 5		Avg	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk:POS)	71.09	95.43	64.8	97.6	68.57	95.98	76.11	91.86	68.53	97.45	69.82	95.66
(POS, Chunk:POS)	93.78	97.88	82.28	98.03	90.87	98.05	92.32	94.16	85.88	97.89	89.02	97.20
(word_POS, Chunk:POS)	89.16	98.9	79.71	98.91	87.08	98.43	88.48	97.08	83.48	99.09	85.58	98.48
(POS_word, Chunk:POS)	76.6	98.9	69.59	98.91	71.99	98.43	81.18	97.08	72.35	99.09	74.34	98.48

Table19:Scheme 1

	Set 1		Set 2		Set 3		Set 4		Set 5		Avg	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk:POS)	63.85	96.21	59.14	98.11	64.86	95.75	71.81	93.78	62.93	97.63	64.53	96.28
(POS, Chunk:POS)	92.51	98.53	77.27	98.58	87.89	98.25	89.73	95.16	80.49	98.87	85.57	97.1
(word_POS, Chunk:POS)	78.64	99.1	72.72	99.6	81.08	98.6	84.55	97.56	76.78	99.71	78.75	98.91
(POS_word, Chunk:POS)	66.08	99.1	63.04	99.6	67.43	98.6	76.35	97.56	65.58	99.71	67.61	98.91

Table20:Scheme 2

	Set 1		Set 2		Set 3		Set 4		Set 5		Avg	
	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec	Prec	Rec
(word, Chunk:POS)	56.42	95.07	54.59	97.53	59.6	95.05	65.38	91.48	55.72	97.23	58.34	95.27
(POS, Chunk:POS)	89.03	97.11	73.74	97.31	84.39	97.35	85.98	92.58	77.46	97.27	82.12	96.34
(word_POS, Chunk:POS)	73.06	98.61	68.11	98.94	76.97	97.76	76.93	96.87	70.73	98.73	73.16	98.18
(POS_word, Chunk:POS)	60.38	98.61	57.9	98.94	62.17	97.76	69.33	96.87	59.98	98.73	61.95	98.18

Table 21:Scheme 3

As can be inferred from Table 19, Table 20 and Table 21 the first scheme
token: <word>_<POS-tag>
label: <2-tag chunk boundary>:POS-tag:<chunk label> (if this is the first token of the chunk.)

<2-tag chunk boundary>:POS-tag (otherwise)

is giving the highest precision 89.02% but again to be noted that word_pos tag approach is not far behind with 85.58% precision and highest recall 98.48%.

Another interesting fact is that recall value of word_pos and pos_word approach is same in all schemes, this is because ordering seems to add no new knowledge to existing model.

5. Conclusion

In conclusion, from our experiments we claim that taking POS or word_POS as the tokens and following a 4-colontag (chunking tag with pos information) scheme and subsequently converting the results to 2-tag set is the better way to identify chunk boundaries. For chunk labeling the chunks scheme 1 is best .Adding pos-tag information to tokens improves the precision and recall of chunk labeling.

Since the approach is pretty much generalized, so this approach can be used for other Indian languages as well.

References:

An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition
By Daniel Jurafsky and James H. Martin

Miles Osborne 2000. Shallow Parsing as Partof-Speech Tagging. *Proceedings of CoNLL-2000*. (2000)

Lance A. Ramshaw, and Mitchell P. Marcus. 1995.Text Chunking Using Transformation-Based Learning.*Proceedings of the 3rd Workshop on Very Large Corpora* (1995) 88.94

W. Skut and T. Brants 1998. Chunk Tagger, Statistical Recognition of Noun Phrases. *ESSLLI-1998* (1998)

Thorsten Brants. 2000. TnT - A Statistical Part-of-Speech Tagger *Proceedings of the sixth conference on Applied Natural Language Processing* (2000) 224.231