Dependency Parsing - Hindi

Ayan Biswas - 2019121009
Veeral Agarwal - 2019114009
Mentor - Saujas Vaduguru
Course instructor - Prof. Manish Shrivastava

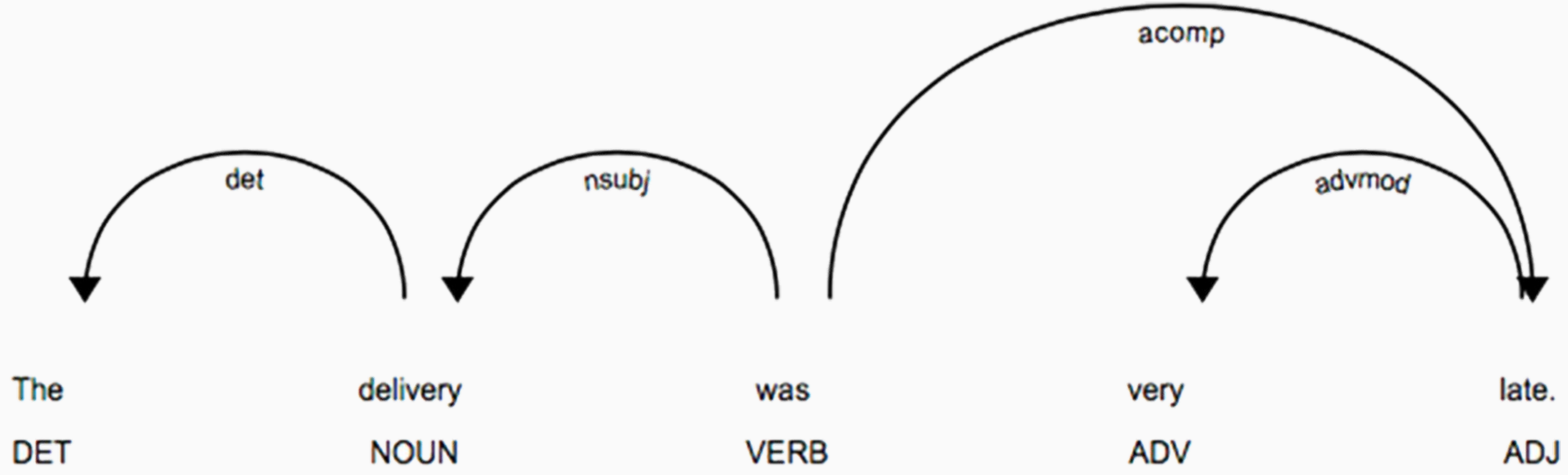
Problem statement

In this project, we are creating a dependency parser (Hindi) for which the training dataset will be provided. A dependency parser identifies the syntactic dependency between words in a sentence. While a lot of literature exists in the dependency parser community in NLP, there are few well-developed tools for Indian languages.

Introduction

A dependency parser explores a sentence's grammatical form, defining associations between "head" words and modifier words. A dependency parse of a short sentence is seen in the diagram below. The arrow from moving to faster shows that faster modify moving, and the mark ad mod applied to the arrow defines the dependency's exact existence.

Example


acomp
det
nsubj
advmod
The
DET
delivery
NOUN
was
VERB
very
ADV
late.
ADJ

Transition based parsing

The initial state is to have all of the words in order on the buffer, with a single dummy ROOT node on the stack. The following transitions can be applied:

- Left arc
- Right arc
- shift

Approaches

SVMs

- SVMs are comparatively simple.
- we can take a feature vector and use kernels.

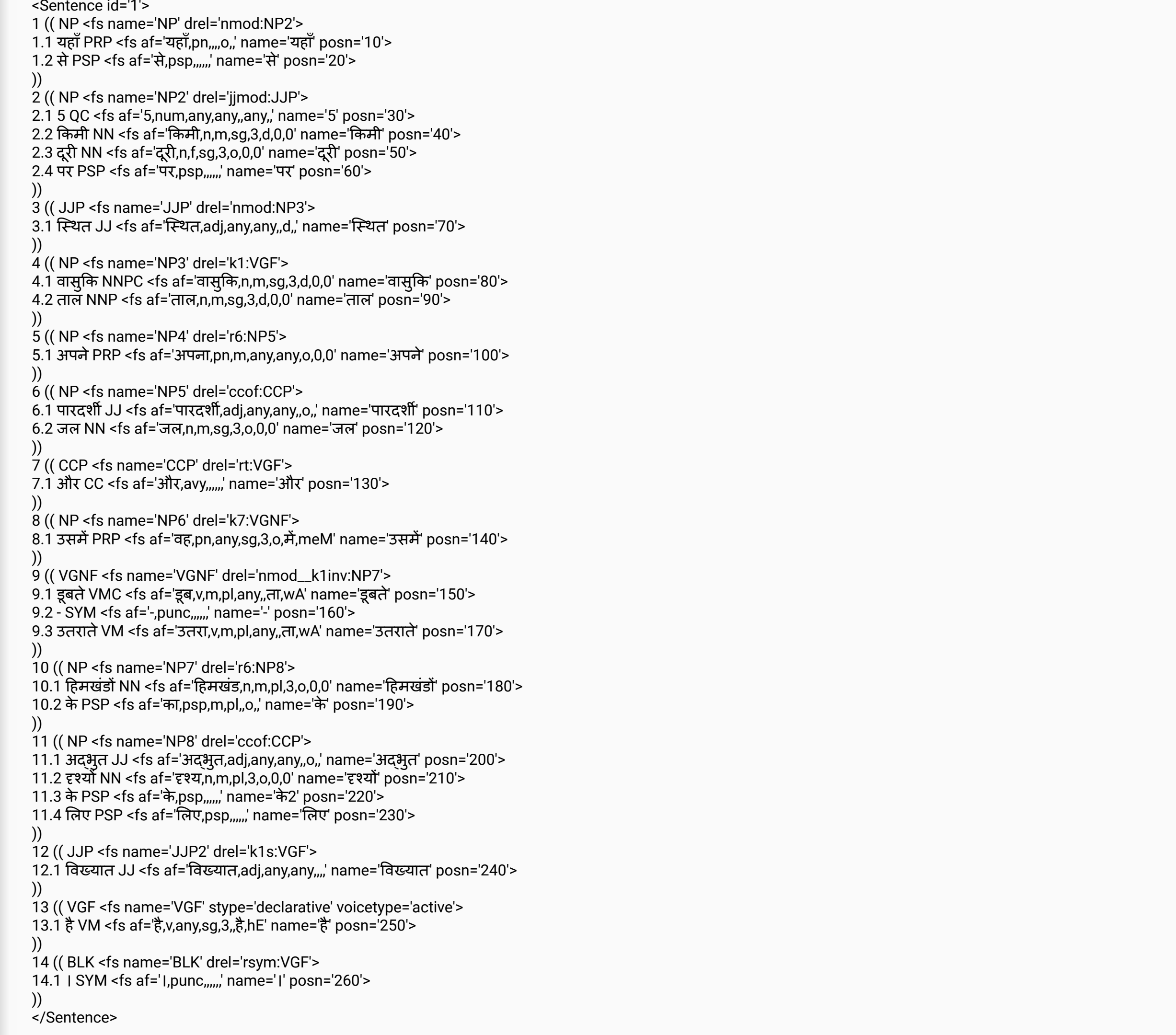
Neural networks

- Neural Nets would require us to take a look at the behavior of a few different architectures.

Dependency parsing using SVMs

Dataset

We have used the data corpus taken from LTRC, IIIT Hyderabad. The data that was provided was already a parsed data by hindi shallow parser in Shakti Standard Form(SSF). This parsed data was used for training , testing and development.



```
<Sentence id='1'>
1 (( NP <fs name='NP' drel='nmod:NP2'>
1.1 यहाँ PRP <fs af='यहाँ,pn,,,,o,,' name='यहाँ' posn='10'>
1.2 से PSP <fs af='से,psp,,,,,,' name='से' posn='20'>
))
2 (( NP <fs name='NP2' drel='jjmod:JJP'>
2.1 5 QC <fs af='5,num,any,any,,any,,' name='5' posn='30'>
2.2 किमी NN <fs af='किमी,n,m,sg,3,d,0,0' name='किमी' posn='40'>
2.3 दूरी NN <fs af='दूरी,n,f,sg,3,o,0,0' name='दूरी' posn='50'>
2.4 पर PSP <fs af='पर,psp,,,,,,' name='पर' posn='60'>
))
3 (( JJP <fs name='JJP' drel='nmod:NP3'>
3.1 स्थित JJ <fs af='स्थित,adj,any,any,,d,,' name='स्थित' posn='70'>
))
4 (( NP <fs name='NP3' drel='k1:VGF'>
4.1 वासुकि NNPC <fs af='वासुकि,n,m,sg,3,d,0,0' name='वासुकि' posn='80'>
4.2 ताल NNP <fs af='ताल,n,m,sg,3,d,0,0' name='ताल' posn='90'>
))
5 (( NP <fs name='NP4' drel='r6:NP5'>
5.1 अपने PRP <fs af='अपना,pn,m,any,any,o,0,0' name='अपने' posn='100'>
))
6 (( NP <fs name='NP5' drel='ccof:CCP'>
6.1 पारदर्शी JJ <fs af='पारदर्शी,adj,any,any,,o,,' name='पारदर्शी' posn='110'>
6.2 जल NN <fs af='जल,n,m,sg,3,o,0,0' name='जल' posn='120'>
))
7 (( CCP <fs name='CCP' drel='rt:VGF'>
7.1 और CC <fs af='और,avy,,,,,,' name='और' posn='130'>
))
8 (( NP <fs name='NP6' drel='k7:VGNF'>
8.1 उसमें PRP <fs af='वह,pn,any,sg,3,o,में,meM' name='उसमें' posn='140'>
))
9 (( VGNF <fs name='VGNF' drel='nmod__k1inv:NP7'>
9.1 डूबते VMC <fs af='डूब,v,m,pl,any,,ता,wA' name='डूबते' posn='150'>
9.2 - SYM <fs af='-,punc,,,,,,' name='-' posn='160'>
9.3 उतराते VM <fs af='उतरा,v,m,pl,any,,ता,wA' name='उतराते' posn='170'>
))
10 (( NP <fs name='NP7' drel='r6:NP8'>
10.1 हिमखंडों NN <fs af='हिमखंड,n,m,pl,3,o,0,0' name='हिमखंडों' posn='180'>
10.2 के PSP <fs af='का,psp,m,pl,,o,,' name='के' posn='190'>
))
11 (( NP <fs name='NP8' drel='ccof:CCP'>
11.1 अद्भुत JJ <fs af='अद्भुत,adj,any,any,,o,,' name='अद्भुत' posn='200'>
11.2 दृश्यों NN <fs af='दृश्य,n,m,pl,3,o,0,0' name='दृश्यों' posn='210'>
11.3 के PSP <fs af='के,psp,,,,,,' name='के2' posn='220'>
11.4 लिए PSP <fs af='लिए,psp,,,,,,' name='लिए' posn='230'>
))
12 (( JJP <fs name='JJP2' drel='k1s:VGF'>
12.1 विख्यात JJ <fs af='विख्यात,adj,any,any,,,,' name='विख्यात' posn='240'>
))
13 (( VGF <fs name='VGF' stype='declarative' voicetype='active'>
13.1 है VM <fs af='है,v,any,sg,3,,है,hE' name='है' posn='250'>
))
14 (( BLK <fs name='BLK' drel='rsym:VGF'>
14.1 । SYM <fs af='।,punc,,,,,,' name='।' posn='260'>
))
</Sentence>
```

Our approach

- Data extraction and simplification.
- Head and Dependency extraction.
- Non parsable sentences and Unknown dependencies.
- Model creation.
- Testing models.

Initial data

```
1 (( NP <fs name='NP' drel='nmod:NP2'>

1.1 यहाँ PRP <fs af='यहाँ,pn,,,,o,,' name='यहाँ' posn='10'>

1.2 से PSP <fs af='से,psp,,,,,,' name='से' posn='20'>

))

2 (( NP <fs name='NP2' drel='jjmod:JJP'>

2.1 5 QC <fs af='5,num,any,any,,any,,' name='5' posn='30'>

2.2 किमी NN <fs af='किमी,n,m,sg,3,d,0,0' name='किमी' posn='40'>

2.3 दूरी NN <fs af='दूरी,n,f,sg,3,o,0,0' name='दूरी' posn='50'>

2.4 पर PSP <fs af='पर,psp,,,,,,' name='पर' posn='60'>

))

3 (( JJP <fs name='JJP' drel='nmod:NP3'>

3.1 स्थित JJ <fs af='स्थित,adj,any,any,,d,,' name='स्थित' posn='70'>

))

4 (( NP <fs name='NP3' drel='k1:VGF'>

4.1 वासुकि NNPC <fs af='वासुकि,n,m,sg,3,d,0,0' name='वासुकि' posn='80'>

4.2 ताल NNP <fs af='ताल,n,m,sg,3,d,0,0' name='ताल' posn='90'>
```

Simplified data

H NP NP nmod NP2

T यहाँ PRP यहाँ

T से PSP से

H NP NP2 jjmod JJP

T 5 QC 5

T किमी NN किमी

T दूरी NN दूरी

T पर PSP पर

H JJP JJP nmod NP3

T स्थित JJ स्थित

H NP NP3 k1 VGF

T वासुकि NNPC वासुकि

Head extracted

H यहाँ यहाँ NP PRP NP nmod NP2

H दूरी दूरी NP NN NP2 jjmod JJP

H स्थित स्थित JJP JJ JJP nmod NP3

H ताल ताल NP NNP NP3 k1 VGF

H अपने अपना NP PRP NP4 r6 NP5

H जल जल NP NN NP5 ccof CCP

Dependency extracted

H यहाँ यहाँ NP PRP NP nmod NP2 ; H दूरी दूरी NP NN NP2 jjmod JJP ; R ; nmod

H दूरी दूरी NP NN NP2 jjmod JJP ; H स्थित स्थित JJP JJ JJP nmod NP3 ; R ; jjmod

H स्थित स्थित JJP JJ JJP nmod NP3 ; H ताल ताल NP NNP NP3 k1 VGF ; R ; nmod

H ताल ताल NP NNP NP3 k1 VGF ; H है है VGF VM VGF NULL ROOT ; R ; k1

Models

Features fit to the model are combinations of:

- Head words i, j
- Part of Speech tag of head words i, j
- Chunk Tags of chunks

i, j - Chunks passed to the classifier.

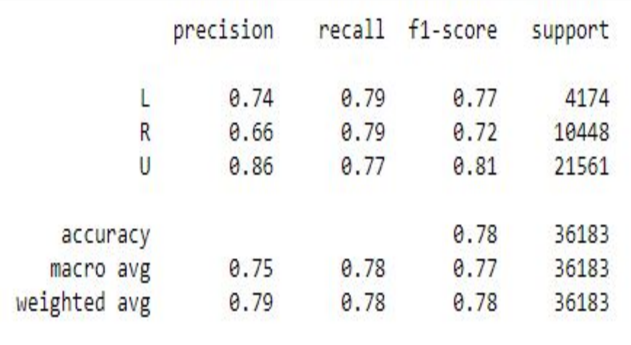
Labels to be predicted:

- Edge Dependency Direction - L, R, U
- Edge type: Ex: nmod, jjmod

Total no. of models = 14 (7 + 7 for each label set)

Results

Word + POS + Chunk - L/R/U

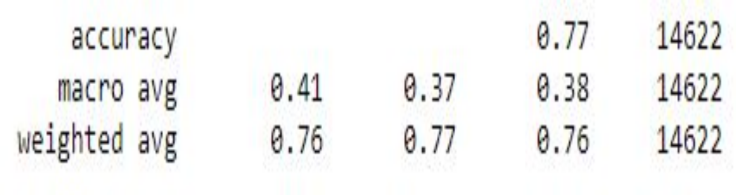


```
              precision    recall  f1-score   support

           L       0.74      0.79      0.77      4174
           R       0.66      0.79      0.72     10448
           U       0.86      0.77      0.81     21561

    accuracy                           0.78     36183
   macro avg       0.75      0.78      0.77     36183
weighted avg       0.79      0.78      0.78     36183
```

Word + POS + Chunk - Edge Relationship Labels

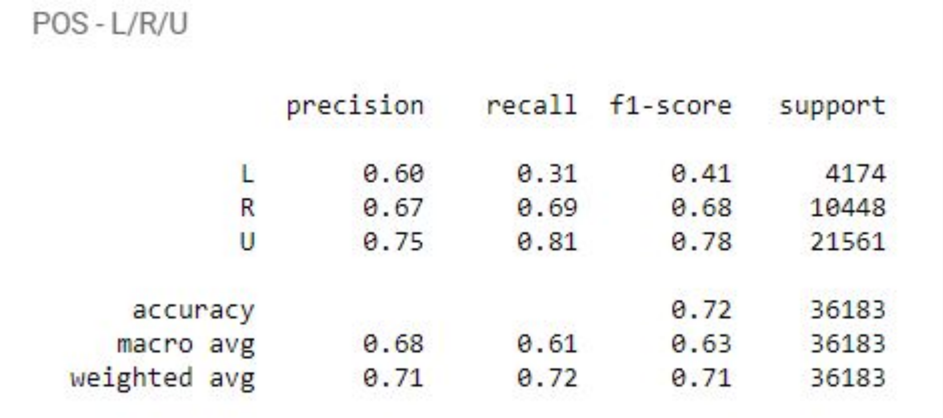


```
    accuracy                           0.77     14622
   macro avg       0.41      0.37      0.38     14622
weighted avg       0.76      0.77      0.76     14622
```

Conclusions and inferences

- Overall accuracy of models is around 77% for both L/R/U and Edge Relationship Labels.
- Data can be interpreted as an Ordered Set of Chunk Features.
- What does the overall 77% accuracy mean?
- POS tag -> L/R/U does terrible in Precision/Recall/F1 score. The accuracy seems to be high - 72% due to distribution of samples pulling it up. The data being high dimensional, and as previously argued decently linearly separable, it is fitting very strongly to the probability distribution of labels. I believe if the test set had different ratios of L/R/U, this would perform far worse.

Conclusions and inferences



POS - L/R/U

```
              precision    recall  f1-score   support

           L       0.60      0.31      0.41      4174
           R       0.67      0.69      0.68     10448
           U       0.75      0.81      0.78     21561

    accuracy                           0.72     36183
   macro avg       0.68      0.61      0.63     36183
weighted avg       0.71      0.72      0.71     36183
```

Conclusions (End)

- Accuracy is indeed very good, however, this is not a metric. This simple model is insufficient to achieve a robust dependency parse.
- Reference 14 provides a baseline that our process is correct, as accuracy scores match.

Thanks!