ASSIGNMENT FINAL DEMO NLP

Kallol Dey
Rahul Mitra
Shubham Gautam

11.11.2012

Works Done

- Generative Bigram POS Tagging
- Trigram POS Tagging
- Next Word Prediction
- Descriminative Bigram POS Tagging
- A* Implementation for POS Tag
- Finding PATH between two entities using YAGO
- Study of Parser Projection

Bigram POS Tagging : Example 1

- Target word : **RISING**

- ***Oil prices are rising again .***
- **Oil_NN1 prices_NN2 are_VBB rising_VVG again_AV0 ._PUN**

- ***The rising sun .***
- **The_AT0 rising_AJ0 sun_NN1 ._PUN**

- ***The attempted rising was put down .***
- **The_AT0 attempted_AJ0 rising_NN1 was_VBD put_VVN down_AVP ._PUN**

Bigram POS Tagging : Example 2

- Target word : **Sort**

- ***We will sort it out .***
- **We_PNP will_VM0 sort_VVI it_PNP out_AVP ._PUN**

- ***Deb is the worst sort of person I ever met .***
- **Deb_NP0 is_VBZ the_AT0 worst_AJS sort_NN1 of_PRF person_NN1 I_PNP ever_AV0 met_VVN ._PUN**

- ***I am just sort of looking for solution .***
- **I_PNP am_VBB just_AV0 sort_NN1 of_PRF looking_VVG for_PRP solution_NN1 ._PUN [Fail should be 'AV0' adverb]**

Bigram POS Tagging : Example 3

- Target word : **That**

- ***That is my car .***
- **That_DT0 is_VBZ my_DPS car_NN1 ._PUN**

- ***Many experts claim that it is good for your growing baby .***
- **Many_DT0 experts_NN2 claim_CJT that_CJT it_PNP is_VBZ good_AV0 for_PRP your_DPS growing_AJ0 baby_NN1 ._PUN**

- ***It wasn't all that bad .***
- **It_PNP wasn't_NN1 all_AV0 that_DT0 bad_AJ0 ._PUN [Fail should be 'AV0' adverb , 'wasn't' also failed !!]**

A * POS Tagging : Example 1

- Target word : **RISING**

- ***Oil prices are rising again .***
- **Oil_NN1 prices_NN2 are_VBB rising_VVG again_AV0 ._PUN**

- ***The rising sun .***
- **The_AT0 rising_NN1 sun_NN1 ._PUN [Fail rising should be AJ0]**

- ***The attempted rising was put down .***
- **The_AT0 attempted_AJ0 rising_NN1 was_VBD put_VVN down_AVP ._PUN**

A* POS Tagging : Example 2

- Target word : **Sort**

- ***We will sort it out .***
- **We_PNP will_VM0 sort_VVI it_PNP out_AVP ._PUN**

- ***Deb is the worst sort of person I ever met .***
- **Deb_VBZ is_VBZ the_AT0 worst_AJS sort_NN1 of_PRF person_NN1 I_PNP ever_AV0 met_VVN ._PUN**

- ***I am just sort of looking for solution .***
- **I_PNP am_VBB just_AV0 sort_NN1 of_PRF looking_VVG for_PRP solution_NN1 ._PUN [Fail should be 'AV0' adverb]**

A* POS Tagging : Example 3

- Target word : **That**

- ***That is my car .***
- **That_DT0 is_VBZ my_DPS car_NN1 ._PUN**

- ***Many experts claim that it is good for your growing baby .***
- **Many_DT0 experts_NN2 claim_CJT that_CJT it_PNP is_VBZ good_AV0 for_PRP your_DPS growing_AJ0 baby_NN1 ._PUN**

- ***It wasn't all that bad .***
- **It_PNP wasn't_VVD all_DT0 that_DT0 bad_AJ0 ._PUN [Fail should be 'AV0' adverb]**

Discriminative Bigram POS Tagging : Example 1

- Target word : **RISING**

- ***Oil prices are rising again .***
- **Oil_NN1 prices_NN2 are_VBB rising_NN1 again_NN1 ._PUN**

- ***The rising sun .***
- **The_AT0 rising_NN1 sun_NN1 ._PUN [Fail rising should be AJ0]**

- ***The attempted rising was put down .***
- **The_AT0 attempted_NN1 rising_NN1 was_NN1 put_NN1 down_AVP ._PUN**

Discriminative POS Tagging : Example 2

- Target word : **Sort**

- ***We will sort it out .***
- **We_PNP will_VM0 sort_VVI it_PNP out_NN1 ._PUN**

- ***Deb is the worst sort of person I ever met .***
- **Deb_NP0 is_VBZ the_AT0 worst_NN1 sort_NN1 of_PRF person_NN1 I_PNP ever_AV0 met_VVD ._PUN**

- ***I am just sort of looking for solution .***
- **I_PNP am_VBB just_AV0 sort_NN1 of_PRF looking_VVG for_PRP solution_NN1 ._PUN [Fail should be 'AV0' adverb]**

Discriminative Bigram POS Tagging : Example 3

- Target word : **That**

- ***That is my car .***
- **That_DT0 is_VBZ my_NN1 car_NN1 ._PUN**

- ***Many experts claim that it is good for your growing baby .***
- **Many_DT0 experts_NN1 claim_NN1 that_CJT it_PNP is_VBZ good_AJ0 for_PRP your_DPS growing_NN1 baby_NN1 ._PUN**

- ***It wasn't all that bad .***
- **It_PNP wasn't_NN1 all_AV0 that_CJT bad_NN1 ._PUN [Fail should be 'AV0' adverb] * Bigram gen shows it as DT0.**

Same sentence : How all algorithms behave ? Case 1

Sentence : **To be or not to be that is the question .**

- Bigram POS :

  To_TO0 be_VBI or_CJC not_XX0 to_TO0 be_VBI that_CJT is_VBZ the_AT0 question_NN1 ._PUN

- A * :

  To_TO0 be_VBI or_CJC not_XX0 to_TO0 be_VBI that_CJT is_VBZ the_AT0 question_NN1 ._PUN

- Bigram Dis :

  To_TO0 be_VBI or_CJC not_XX0 to_TO0 be_VBI that_CJT is_VBZ the_AT0 question_NN1 ._PUN

Same sentence : How all algorithms behave : Case2

Sentence : **The rising sun .**

- Bigram POS :

The_AT0 rising_AJ0 sun_NN1 ._PUN

- A * :
- **The_AT0 rising_NN1 sun_NN1 ._PUN [Fail rising should be AJ0]**

- Bigram Dis :

The_AT0 rising_NN1 sun_NN1 ._PUN [Fail rising should be AJ0]

An Experiment Containing All Algorithms.

- Same setup applied for all the algorithms.
- Two Test Cases.
- Final Checking :
  - Efiiciency
  - Per Tag Accuracy
  - Confusion Matrix
  - Find Some Sentence Whihc Will Have Different POS Tag in different algorithm.
  -

Setup 1 : A00

- Confusion Matrix ,Pertag precison ,recall ,F-Score is available in .csv files.
- Bigram Generative Accuracy : 95 %
- Trigram Generative Accuracy :95.4%
- A* Accuracy : 85%
- Bigram Discriminative Accuracy : 80%

Setup 1 : A00 : NN1 Confusion

- Bigram Generative :

  AJ0:2,AV0:1,VVG 1:AVP:2 VVB:5,NN1:1002,NP0:36,NN2:39,VVI:1

- Trigram Generative :

  AJ0:3,AV0:1,VVG:1,AVP:1,VVB:6,NN1:997,NP0:35,NN2:43,VVI:2

- Bigram Discriminative :

  AJ0:10,AV0:6,VVG:1,VVB:1,NN1:890,NN2:14,NP0:167

- A* :

  NP0 : 50 , AV0 23

- **NN1 Highly Confused tag.**

Setup 1 : A00 : DT0 Confusion

- **Trigram Gen :**
- AJ0:3, AV0:5, PNX: 1 ,DT0:120, CJT:2,NP0:2

- **Generative Bigram :**
- AJ0:2 ,AV0:5,PNX:1,DT0:121,CJT:2,NP0:2

- Conclusion : DT0 is less confused !!

Setup 2 : A69

- Confusion Matrix ,Pertag precison ,recall ,F-Score is available in .csv files.
- Bigram Generative Accuracy : 94.8 %
- Trigram Generative Accuracy :95.2%
- A* Accuracy : 86%

Conclusions on POS Tagging

- A* POS Tagger on an average expands 6 times the length of the given sentence.
- We conclude that the Viterbi Algorithm is better than A* Star in this case because to minimize the no. of expansions in A* Star (to find a good heuristic for h) we need O(mn^2) time which is same as that of Viterbi with an extra overhead of searching.

NWP : Efficiency

- Made triplet of sentences and checked accuracy.
- We taken till available 5 suggestions in both bigram and trigram
- With Tag it was 31 %
- Perplexity With POS : 7612
- Without Tag it was about 25 %
- Perplexity WithOut POS : 6678

NWP Some Cases

Input : **The rising**

- ---------WithOut POST----------------
- Trigram :
- Word: **IMPORTANCE** Val :0.3333333333333333
- Word: **TIMES** Val :0.3333333333333333
- Word: ARMENIAN Val :0.3333333333333333
- Bigram :
- Word: . Val :0.09523809523809523
- Word: TO Val :0.09523809523809523
- ---------With POST----------------
- Trigram :
- Word: **IMPORTANCE** Val :1.53514768600473E-4
- Bigram :
- Word: **TIME** Val :2.2924415221811705E-4

NWP Some Cases

Input : **The attempted rising**

- ---------WithOut POST----------------
- Bigram :
- Word: . Val :0.09523809523809523
- Word: **TO** Val :0.09523809523809523
- Word: **AND** Val :0.07142857142857142
- ---------With POST----------------
- Bigram :
- Word: **TO** Val :0.010214862036588118
- Word: **IN** Val :0.005269375581831756
- Word: **WITH** Val :0.0018962539718156356
- Word: **FROM** Val :0.0012618319703277749

YAGO

We implemented it in both BFS way and DFS way. The efficiency of this two approach very from case to case.

Yago found out interesting relationship between two words.

We are producing some cases here :

YAGO : Cases

Relation between BigB with Guzaarish (IMDB rating 7.2 !!!)

- Amitabh_Bachchan : <hasChild> : Abhishek_Bachchan
- Abhishek_Bachchan : <isMarriedTo> : Aishwarya_Rai
- Aishwarya_Rai : <actedIn> : Guzaarish

YAGO : Cases

Relation between Jawaharlal_Nehru with India

- Jawaharlal_Nehru : <hasChild> : Indira_Gandhi
- Indira_Gandhi : <hasChild> : Sanjay_Gandhi
- Sanjay_Gandhi : <livesIn> : Uttar_Pradesh
- Uttar_Pradesh : <isLocatedIn> : India

So long path because directly no other short path between JN and India

Parser Projection

- Input : Source Language Sentence
- Output: Bracketed Tree structure of Target Language Sentence
- Conversion Function : a mapping function which depends on both the languages.

- From “A general approach to natural language conversion” by Md. Abu Nuser Musud, Md. Muntusir Mamun Joarder, Md. Turiq-Ul-Azam, IEEE JNMIC 2003

Parsing

- NLTK can be used for parsing of input sentence.
- After successful parsing, the exact structure of input sentence i.e. natural language-1 (NL1) is known.
- Now our Goal is : The corresponding structure of that NL2 must be known

Conversion / Mapping

- Goal : To convert this structure of NL1 to a structure suitable to generate NL2
- Problem : a large variation in both the structures (allignment).
- Solution : Rule-based conversion

Example

- The book is on the table.
- Grammar:

  S -> NP VP

  NP -> Det NP0

  ....
- But for Hindi, the sentence is :

  पुस्तक मेज पर है.

Example(cont..)

So here, “Det” does not play any role.

- The corresponding rule :
- NP -> NP0

and so on for furthur rules.

Problems with the approach

But there will be false positives and false negetives.

This is due to the fact of lexical ambiguities when a word has multiple meaning or sense. If the sense is correctly captured the parser and in the ruleset the the corresponding mapping is there only then can the projection be correct.

Wrong projection can be also due the fact that original english sentence may have syntactical ambiguity. eg.. I saw the boy with the telescope. Here two possible parse tree can be obtained so depending on which parse tree the english parser gives corresponding hindi projection will be given.

One Possible Solution

- We can use Dependency Parsing instead of convential parsing.
- This is because in Dependency Parse Trees has only relation between the main verb and all other parts example Subject and Object.
- Since the relation between the parts does not change between languages. Also we can then convert the dependency graph into bracketed structure.

Why AV0 is confused with PRP?

- Words that are sometimes prepositions can act as adverbs.

- A preposition requires an object while an adverb does not.

Example

- The bird flew off.
- Here, “off” will be treated as AV0 as it has not an object.

But .....

Example (cont)..

- The bird flew off *the door.*
- Here, “off” will be treated as PRP as it has an object phrase “the door”.

Reason for confusion

- In the corpus, probabilty of current word to be tagged depends on the previous tag and current word. Since these conflicting words have both PRP and AVO occurances in the corpus therefore the tag depends on the previous tag.
- But in reality, it should depends upon the the presence of “*object phrase*” and this information is not availbale in POS TAGGING stage..

Explanation for lower accuracy in A Star than Viterbi

We had taken our h(n) to be

- $h(n) = -\log(P(W_i | T_i)) + \sum_{j=i+1}^{n} - \log(\max(P(W_j| T)))$ where T is element of the set of all tags.

However here if any of the word in the right of the current word is unknown that is not is the training corpus then we assign a low probability for $P(W_i|T)$

But now h(n) is no longer less than $h^*(n)$ as – log(very low value) is large. So if a unknown word appears in the sentence then our algorithm did not give the optimum path. So we changed our h(n) to

- $h(n) = -\log(P(W_i | T_i))$ + (length of the sentence – current word position) + 1

The 1 at the end is added since PUN should have the second last column before goal state.

This heuristic does not suffer from the above problem and hence we found its accuracy to be 91% which was almost same as our bigram Tagger when Rule based unknown word handling was not introduced.

Thank You

Any Questions ?

Extra Slide : Tag Descriptions 1.

AJ0	Adjective (general or positive) (e.g. good, old, beautiful)
AJC	Comparative adjective (e.g. better, older)
AJS	Superlative adjective (e.g. best, oldest)
AT0	Article (e.g. the, a, an, no)
AV0	General adverb: an adverb not subclassified as AVP or AVQ (see below) (e.g. often, well, longer (adv.), furthest.
AVP	Adverb particle (e.g. up, off, out)
AVQ	Wh-adverb (e.g. when, where, how, why, wherever)
CJC	Coordinating conjunction (e.g. and, or, but)
CJS	Subordinating conjunction (e.g. although, when)
CJT	The subordinating conjunction that
CRD	Cardinal number (e.g. one, 3, fifty-five, 3609)
DPS	Possessive determiner-pronoun (e.g. your, their, his)
DT0	General determiner-pronoun: i.e. a determiner-pronoun which is not a DTQ or an AT0.
DTQ	Wh-determiner-pronoun (e.g. which, what, whose, whichever)
EX0	Existential there, i.e. there occurring in the there is ... or there are ... construction

Extra Slide : Tag Descriptions 2.

ITJ	Interjection or other isolate (e.g. oh, yes, mhm, wow)
NN0	Common noun, neutral for number (e.g. aircraft, data, committee)
NN1	Singular common noun (e.g. pencil, goose, time, revelation)
NN2	Plural common noun (e.g. pencils, geese, times, revelations)
NP0	Proper noun (e.g. London, Michael, Mars, IBM)
ORD	Ordinal numeral (e.g. first, sixth, 77th, last) .
PNI	Indefinite pronoun (e.g. none, everything, one [as pronoun], nobody)
PNP	Personal pronoun (e.g. I, you, them, ours)
PNQ	Wh-pronoun (e.g. who, whoever, whom)
PNX	Reflexive pronoun (e.g. myself, yourself, itself, ourselves)
POS	The possessive or genitive marker 's or '
PRF	The preposition of
PRP	Preposition (except for of) (e.g. about, at, in, on, on behalf of, with)
PUL	Punctuation: left bracket - i.e. (or [
PUN	Punctuation: general separating mark - i.e. . , ! , : ; - or ?
PUQ	Punctuation: quotation mark - i.e. ' or "

Extra Slide : Tag Descriptions 3.

PUR	Punctuation: right bracket - i.e.) or]
TO0	Infinitive marker to
UNC	Unclassified items which are not appropriately considered as items of the English lexicon.
VBB	The present tense forms of the verb BE, except for is, 's: i.e. am, are, 'm, 're and be [subjunctive or imperative]
VBD	The past tense forms of the verb BE: was and were
VBG	The -ing form of the verb BE: being
VBI	The infinitive form of the verb BE: be
VBN	The past participle form of the verb BE: been
VBZ	The -s form of the verb BE: is, 's
VDB	The finite base form of the verb DO: do
VDD	The past tense form of the verb DO: did
VDG	The -ing form of the verb DO: doing
VDI	The infinitive form of the verb DO: do
VDN	The past participle form of the verb DO: done
VDZ	The -s form of the verb DO: does, 's

Extra Slide : Tag Descriptions 4.

VHB	The finite base form of the verb HAVE: have, 've
VHD	The past tense form of the verb HAVE: had, 'd
VHG	The -ing form of the verb HAVE: having
VHI	The infinitive form of the verb HAVE: have
VHN	The past participle form of the verb HAVE: had
VHZ	The -s form of the verb HAVE: has, 's
VM0	Modal auxiliary verb (e.g. will, would, can, could, 'll, 'd)
VVB	The finite base form of lexical verbs (e.g. forget, send, live, return) [Including the imperative and present subjunctive]
VVD	The past tense form of lexical verbs (e.g. forgot, sent, lived, returned)
VVG	The -ing form of lexical verbs (e.g. forgetting, sending, living, returning)
VVI	The infinitive form of lexical verbs (e.g. forget, send, live, return)
VVN	The past participle form of lexical verbs (e.g. forgotten, sent, lived, returned)
VVZ	The -s form of lexical verbs (e.g. forgets, sends, lives, returns)
XX0	The negative particle not or n't
ZZ0	Alphabetical symbols (e.g. A, a, B, b, c, d)

Thank You.

Any Question ?