

Research Paper Implementation: SVMTool

Varada KOLHATKAR
kolha002@d.umn.edu

Machine Learning Class Presentation, Apr 22, 2009

- 1 Introduction
 - Outline
 - Background
 - Problem to solve
- 2 SVMTool
 - Approaches for Part-of-Speech Tagging
 - SVMTool Approach
 - SVMTool
- 3 Experiments and Results
 - Experiments on WSJ
 - Linear SVM in primal formulation
 - My Implementation
- 4 Conclusion

Part-of-Speech₁

- A part-of-speech(POS) is a linguistic category of words.
Examples: noun, verbs, pronoun, article, adjective, adverb etc
- More extensive tagsets
 - Penn Treebank 45 word classes

Tag	Description	Example
CC	coordin conjunction	<i>and,but,or</i>
DT	determiner	<i>a,the</i>
VB	verb, base form	<i>eat</i>
VBD	verb, past tense	<i>ate</i>
...

- Brown 87 word classes
- C7 146 word classes

Part-of-Speech₂

Book/VB that/DT Flight/NN ./.

- POS tagging is the process of assigning a POS tag to each word in a corpus
- Tags are also assigned to punctuation marks
- not simple because a word can have multiple parts-of-speech
Book/NN that/DT Flight/NN ./ . ??

Significance of Part-of-Speech

- Give a lot of information about a word and its neighbors
- Can tell how the word is pronounced
Example: the noun CONtent and the adjective conTENT
- Stemming for information retrieval

Problem Statement

To Find an accurate, robust, portable, flexible and efficient solution for part-of-speech tagging

Accurate → competitive accuracy with other POS taggers

Robust → can handle overfitting and unknown words

Portable → language independent

Flexible → highly customizable

Efficient → in terms of learning and tagging time

Approaches for POS tagging

- Hidden Markov Models [Weischedel et al. 93, Brants 00 (TnT)]
Probability of a given word having a given tag in a given context
- Rule Based Tagging [Voutilainenm, 1995, 1999]
A set of handwritten disambiguation rules.
- Transformation Based Tagging [Brill 95]
Share features of rule based and HHM based taggers
- Maximum Entropy [Ratnaparkhi 96, Toutanova & Manning 00]
- Decision Trees [Màrquez & Rodríguez 97]
- Support Vector Machines [Nakagawa et al. 01]

The state-of-the-art accuracy lie between 96.4% - 96.7%

SVMTool by Jesús Giménez and Lluís Màrquez

The DT investor NN was VBD instrumental JJ in IN tapping VBG
Mr. NNP Wolf NNP to TO run VB the DT air NN cargo NN
unit NN of IN Tiger NNP International NNP Inc NNP . .

input

SVMT
tagger

output

DICTIONARY

MODELS

Dictionary

- Extracted from the training corpus
- Contains all possible tags for each word
- Format $\langle word \rangle \langle N_occurrences \rangle \langle N_possible_tags \rangle$
 $1\{\langle tag(i) \rangle \langle N_occurrences(i) \rangle\}N$
- Examples:
walk 12 3 NN 2 VB 5 VBP 5
talk 51 3 NN 26 VB 20 VBP 5
0.60 5 1 CD 5
- Can be repaired based on frequency heuristics or with the corrections provided by the user
- Makes the tagger robust to corpus errors

SVM Models

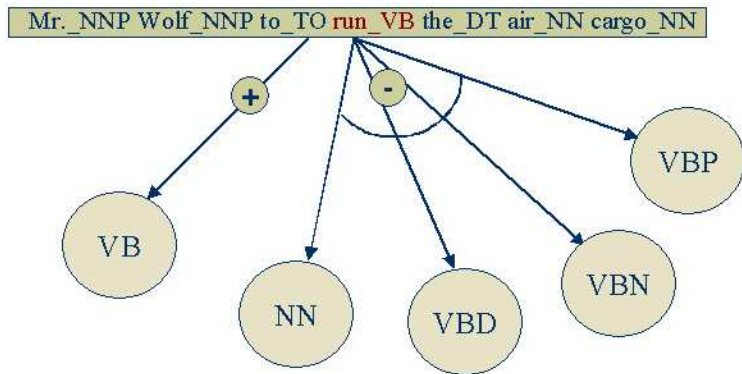
- Uses Joachims's SVM^{light} (<http://svmlight.joachims.org/>)
 - Input: Feature vectors
 - Output: α weights and bias
- Linear discriminant (w, b) hyperplane
 $\text{sgn}(f(x, w, b))$, where $f(x, w, b) = \langle w \cdot x \rangle + b$
- Dual Formulation α
 $f(x, \alpha, b) = \sum_{i=1}^N y_i \alpha_i \langle x_i \cdot x \rangle + b$
- Linear SVM in a primal formulation
 - Dual representation based output of SVM^{light} is converted into primal form using

$$w^* = \sum_{i=1}^N y_i \alpha_i^* x_i$$

Binarizing the classification problem

- Part-of-speech tagging is a multi-class classification problem
- One vs All binarization
 - SVM is trained for every part of speech
 - When tagging, the word is evaluated against all SVM models and the most confident tag is considered
- Dictionary based reduction of word ambiguity
- Wise selection of negative examples

Selection of negative examples



<run 211 5 NN 37 VB 98 VBD 2 VBN 50 VBP 24>

Feature Codification

- Sparse binary features (yes/no type)

Example: `previous_word_is_the`, `following_word_may_be_IN`

word features	$w_{-3}, w_{-2}, w_{-1}, w_0, w_{+1}, w_{+2}, w_{+3}$
POS features	$p_{-3}, p_{-2}, p_{-1}, p_0, p_{+1}, p_{+2}, p_{+3}$
ambiguity classes	a_0, a_1, a_2, a_3
maybe's	m_0, m_1, m_2, m_3
word bigrams	$(w_{-2}, w_{-1}), (w_{-1}, w_{+1}), (w_{-1}, w_0)$ $(w_0, w_{+1}), (w_{+1}, w_{+2})$
PoS bigrams	$(p_{-2}, p_{-1}), (p_{-1}, a_{+1}), (a_{+1}, a_{+2})$
word trigrams	$(w_{-3}, w_{-2}, w_{-1}), (w_{-2}, w_{-1}, w_0)$ $(w_{-2}, w_{-1}, w_{+1}), (w_{-1}, w_0, w_{+1})$ $(w_{-1}, w_{+1}, w_{+2}), (w_0, w_{+1}, w_{+2})$
PoS trigrams	$(p_{-3}, p_{-2}, p_{-1}), (p_{-2}, w_{-1}, a_{+1})$ (p_{-1}, a_{+1}, a_{+2})

Feature Codification Example

Example: Do/VBP n't/RB talk/VB the/DT **talk/NN** unless/IN
you/PRP can/MD walk/VB the/DT walk/NN

Dictionary entry: talk 51 3 NN 26 VB 20 VBP 5

- Word Features

w₋₃is_n't, w₋₂is_talk, w₋₁is_the, w₀is_talk, w₁is_unless ...

- POS Features

p₋₃is_RB, p₋₂is_VB, p₋₁is_DT, p₀is_NN , p₁is_IN ...

- ambiguity classes

a₀is_NN-VB-VBP

- maybe's

m₀may_be_NN, m₀may_be_VB, m₀may_be_VBP ...

- Feature vector Example:

-1 39:1 40:1 47:1 354:1 375:1 420:1 1232:1 1233:1 1234:1
1235:1

Learning Algorithm

- 1: INPUT \rightarrow formatted train-file
- 2: OUTPUT \rightarrow SVM models learned for **all** part of speech
- 3: $POS[] = \{\text{the set of possible part of speech tags}\}$
- 4: Create a different example file corresponding to each POS
- 5: **for** each example word w tagged as t_i in the train-file **do**
 - 6: Extract features, codify features and create a feature vector f_i
 - 7: $T = \{\text{the set of possible tags for } w\}$ (Note that $t_i \in T$)
 - 8: Use f_i as a positive example for t_i
 - 9: Use f_i as a negative example $\forall t_j \in \{T - t_i\}$
- 10: **end for**
- 11: **for** each p_i in POS **do**
 - 12: svm_learn on the examples corresponding to p_i
- 13: **end for**
- 14: return the SVM models learned for each part of speech

Classification Algorithm

- 1: INPUT \rightarrow formatted test file, learned SVM models
- 2: OUTPUT \rightarrow Part-of-speech tagged test file
- 3: $score = 0$
- 4: **for** each target word w in the test file **do**
 - 5: Extract features and create a feature vector for w
 - 6: $possible_pos[] =$ possible tags of w in the dictionary
 - 7: **for** each p_i in $possible_pos$ **do**
 - 8: $ans = svm_classify$ with p_i SVM model
 - 9: **if** $ans > score$ **then**
 - 10: $score = ans$
 - 11: $target_pos = p_i$
 - 12: **end if**
 - 13: **end for**
 - 14: Assign $target_pos$ to w
- 15: **end for**
- 16: return the Part-of-speech tagged test file

Handling Unknown Words

- Add features which will help the system to make decision
- Consider all available POS as ambiguity classes

All features for known words	fs2
prefixes	$s_1, s_1s_2, s_1s_2s_3, s_1s_2s_3s_4$
suffixes	$s_n, s_{n-1}s_n, s_{n-2}s_{n-1}s_n, s_{n-3}s_{n-2}s_{n-1}s_n$
begins with upper case	yes/no
all upper case	yes/no
all lower case	yes/no
contains Capital letter not at the beginning	yes/no
contains > 1 Capital letters not at the beginning	yes/no
contains a period	yes/no
contains a number	yes/no
contains a hyphen	yes/no
word length	integer

SVMTool Software Package

<http://www.lsi.upc.edu/~nlp/SVMTool/>

- SVMT-learner [Training of SVM classifiers]
- SVMT-tagger [POS-tagging of a given input]
- SVMT-evaluator [Study of tagging results]
- SVMT API [Embedded usage of SVMT-tagger]

SVMTlearner Options₁

- Sliding window length [def: 5] core position [def:2]
- Configurable feature set gives flexibility
- SVM model Compression
 - Weight vector components lower than a threshold can be filtered out
 - Increased efficiency while preserving accuracy
 - Can discard up to 70% of the weight vectors
- Feature Filtering
 - Features appearing less than n times can be discarded
 - Addresses overfitting problem and exhibits higher accuracy

SVMTlearner Options₂

- C parameter tuning
 - Overfitting problem is well addressed via tuning of C
 - Need to provide validation set and the interval to be explored
 - Improvement in results was observed
- Dictionary repairing
 - Heuristics
 - Using a list of corrections provided

< the 50975 6 CD 1 DT 50959 JJ 7 NN 1 NNP 6 VBP 1 >
< the 50975 1 DT 50959 >

SVMTagger Options

- tagging scheme
 - greedy [default]
 - sentence-level
 - The global sentence sum of SVM tagging scores is the function to maximize
- tagging direction
 - left-to-right [default]
 - right-to-left
 - both left-to-right and right-to-left
- number of tagging passes (1 or 2) [default: 1]
- backup lexicon

Experiments Setting

- Wall Street Journal(Penn Treebank III)
- 1,17 million words
- Randomly divided into Training (60%), Validation (20%), Test (20%)
- (2.81%) words from the test set are unknown to the training set
- Penn Tree bank tag set \rightarrow 48 tags (including punctuation)
- One pass, left-to-right and right-to-left combined greedy tagging scheme

English Results on WSJ

	known	amb.	unk.	all.	time _{tagging}
TnT	96.76%	92.16%	85.86%	96.46%	23 sec
SVMTool	97.39%	93.91%	89.01%	97.16%	780 sec

	TnT	SVMT	Nakagawa
Speed _{tagging} (words/sec)	50,000	1500	20
WSJ time _{tagging}	23sec	780sec	16hours

- Outperforming TnT but slow
- Faster than other SVM based approaches

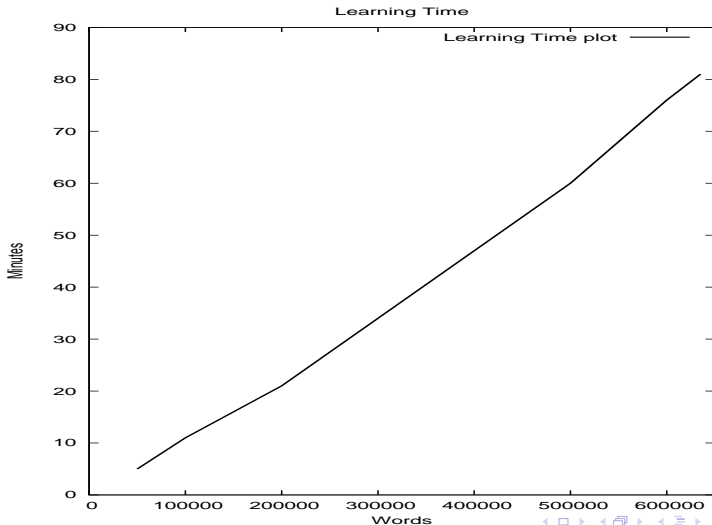
Make the tagger efficient!

- Transform the alpha weights to the weights in the original space using

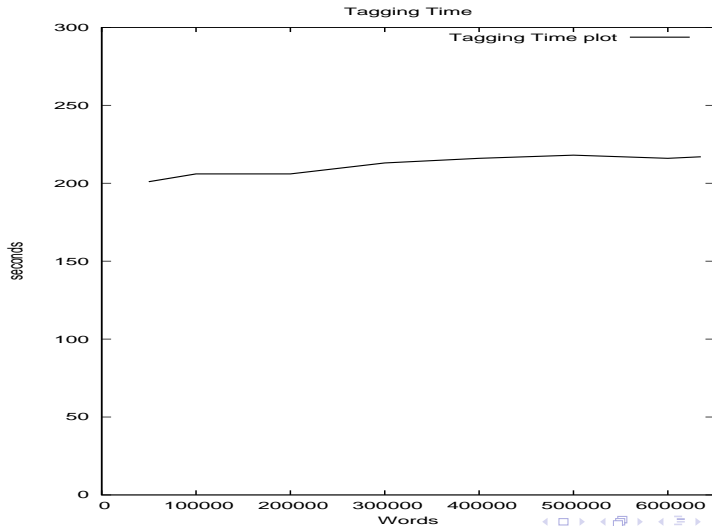
$$w^* = \sum_{i=1}^N y_i \alpha_i^* x_i$$

- Linear kernels: more efficient learning as well as tagging
- Learning time still remains linear with respect to the number of training examples
- Forces to define a richer feature space

Learning Time



Tagging Time



Preliminary Results

- Basic left to right tagging scheme
- Supports word features, POS features, ambiguity classes and maybe's
- Experiments done on a subset of Penn Treebank II

	500 words	100k
Varada-SVMTool	83.88%	80.83%
SVMTool	98.58%	93.88%

- Trying to figure out the reasons for low accuracy ...

Conclusion

Practical SVM based part-of-speech tagger

- Accurate: competitive with other part-of-speech taggers
- Efficient: linear SVMs, primal formulation, feature filtering and model compression
- Robust: soft margin SVMs, two-passes, LR + RL
- Flexible: rich feature set, tagging strategies
- Portable: applied to English, Spanish and Catalan
- Simple: ease to configure, tune and use

Acknowledgements

Presentation available at
<http://www.d.umn.edu/~kolha002/Project-SVMTool.pdf>

THANK YOU!