Pos tagging with HMM & SVM

1 Dataset

Brown Corpus

- 500 samples of English-language text from works published in the United States in 1961
  - 1’100’000 million words
  - 56’000 different words
- Universal Tagset
  - Twelve universal part-of speech categories

Tag	Meaning	English Examples
ADJ	adjective	*new, good, high, special, big, local*
ADP	adposition	*on, of, at, with, by, into, under*
ADV	adverb	*really, already, still, early, now*
CONJ	conjunction	*and, or, but, if, while, although*
DET	determiner, article	*the, a, some, most, every, no, which*
NOUN	noun	*year, home, costs, time, Africa*
NUM	numeral	*twenty-four, fourth, 1991, 14:24*
PRT	particle	*at, on, out, over per, that, up, with*
PRON	pronoun	*he, their, her, its, my, I, us*
VERB	verb	*is, say, told, given, playing, would*
.	punctuation marks	*. , ; !*
X	other	*ersatz, esprit, dunno, gr8, univeristy*

2 HMM Pos Tagger

HMM

Considering an Hidden Markov Model,

- Tags are the **hidden** events, since they are unobservable.
- Words are the **observable** events.

Markov Assumption: $P(q_i|\ q_1 \ldots\ q_{i-1}) = P(q_i|\ q_{i-1})$

Decoding: Viterbi Algorithm

- Determining which sequence of hidden state(tags) is the underlying source of some observations(words)

How does it work?

$$\hat{t}_1^n = argmax_{\,t_1^n} P(t_1^n | w_1^n)$$

$$\hat{t}_1^n = argmax_{t_1^n} \frac{P(w_1^n | t_1^n) P(t_1^n)}{P(w_1^n)}$$

$$\hat{t}_1^n = argmax_{t_1^n} P(w_1^n | t_1^n) P(t_1^n)$$

$$P(w_1^n | t_1^n) \approx \prod_{i=1}^{n} P(w_i | t_i)$$

$$P(t_1^n) \approx \prod_{i=1}^{n} P(t_i | t_{i-1})$$

$$\hat{t}_1^n = argmax_{\,t_1^n} P(t_1^n | w_1^n) \approx argmax_{\,t_1^n} \prod_{i=1}^{n} P(w_i | t_i) P(t_i | t_{i-1})$$

$\boldsymbol{P(w_i | t_i)}$ = ***Emission probability***

$\boldsymbol{P(t_i | t_{i-1})}$ = ***Transition probability***

Learning Markov Models

Count the number of occurences in the corpus

Sentence: ”He was about 50 years old.”

$\mathbf{C(<S>\ PRON)++}$ $\quad$ $\boldsymbol{C(PRON\ VERB)++}$

<S>	→	PRON	→	VERB	→	ADV	→	NUM	→	NOUN	→	ADJ	→	.
		↓		↓		↓		↓		↓		↓		↓
		He		was		about		50		years		old		.

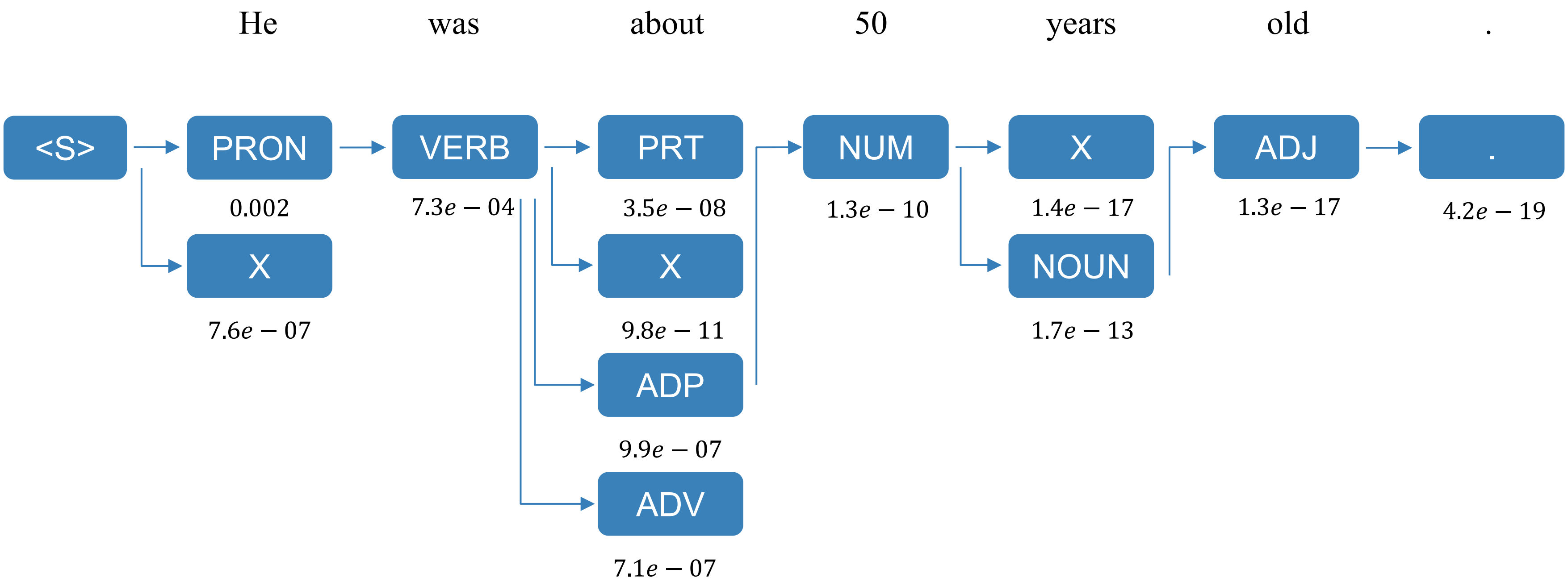
$\boldsymbol{C(PRON \rightarrow He)++}$ $\quad$ $\boldsymbol{C(VERB \rightarrow was)++}$

Divide by the context to get probability

$$P_T(VERB \mid PRON) = \frac{C(PRON\ VERB)}{C(PRON)}$$

$$P_E(was \mid VERB) = \frac{C(VERB \rightarrow was)}{C(VERB)}$$

Viterbi decoding



Unknown words

Words are considered as unknown if they do not appear in the train set.
Indeed, the emission probability could not be considered.

So, the tag associated to this words only depends on the transition probability.

In the Viterbi step associated to the unknown word, will be only considered $P(t_i|t_{i-1})$ instead of $P(w_i|t_i)P(t_i|t_{i-1})$.

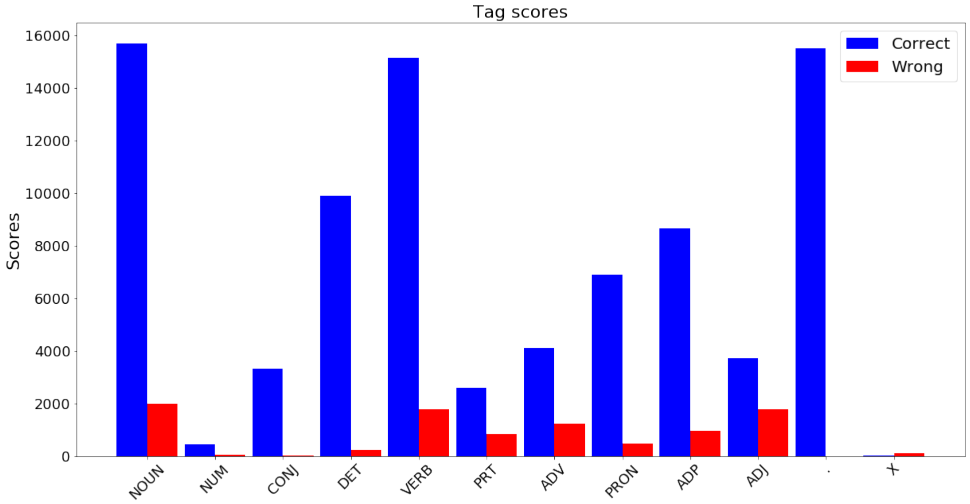
93.2%

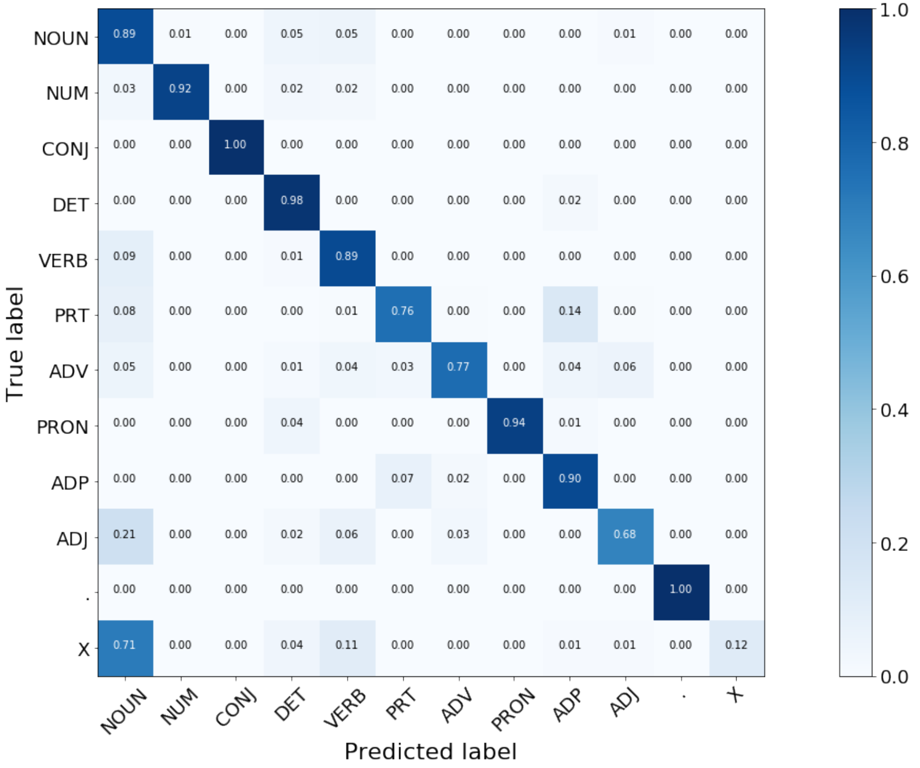
Accuracy

Train set length: 51606 sentences
Test set length: 5734

Precision: 93.3%
Recall: 93.2%
F-Measure: 93.2%

Results


Tag scores
Scores
16000
14000
12000
10000
8000
6000
4000
2000
0
Correct
Wrong
NOUN
NUM
CONJ
DET
VERB
PRT
ADV
PRON
ADP
ADJ
.
X


True label
Predicted label
NOUN 0.89 0.01 0.00 0.05 0.05 0.00 0.00 0.00 0.00 0.01 0.00 0.00
NUM 0.03 0.92 0.00 0.02 0.02 0.00 0.00 0.00 0.00 0.00 0.00 0.00
CONJ 0.00 0.00 1.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
DET 0.00 0.00 0.00 0.98 0.00 0.00 0.00 0.00 0.02 0.00 0.00 0.00
VERB 0.09 0.00 0.00 0.01 0.89 0.00 0.00 0.00 0.00 0.00 0.00 0.00
PRT 0.08 0.00 0.00 0.00 0.01 0.76 0.00 0.00 0.14 0.00 0.00 0.00
ADV 0.05 0.00 0.00 0.01 0.04 0.03 0.77 0.00 0.04 0.06 0.00 0.00
PRON 0.00 0.00 0.00 0.04 0.00 0.00 0.00 0.94 0.01 0.00 0.00 0.00
ADP 0.00 0.00 0.00 0.00 0.00 0.07 0.02 0.00 0.90 0.00 0.00 0.00
ADJ 0.21 0.00 0.00 0.02 0.06 0.00 0.03 0.00 0.00 0.68 0.00 0.00
. 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00 1.00 0.00
X 0.71 0.00 0.00 0.04 0.11 0.00 0.00 0.00 0.01 0.01 0.00 0.12
NOUN
NUM
CONJ
DET
VERB
PRT
ADV
PRON
ADP
ADJ
.
X
1.0
0.8
0.6
0.4
0.2
0.0

90.1%

Accuracy

Train set length: 5000 sentences
Test set length: 5734

Precision: 90.3%
F-Measure: 90%

3

SVM Pos Tagger

How does it work?

SVM training:

- Each training sample is the so-called “word context”.

- The “word context” is a set of features defined on/around the interested word (previous/successive words, tags, suffixes)

- Kernel: RBF
    - Also different kernels has been tried: Linear, sigmoidal and polynomials kernels provided really bad result in term of Accuracy

Problem SVM features

The greatest problem has been the mapping between words and vector (also known as Word Embedding) for which different solution has been tried:

- One hot
  - Each word is encoded as a 1-hot vector (where all the elements are 0 except one, which is 1).
  - Too high-dimensional space, for which the training time was not feasible
- Incremental
  - An incremental number assigned to each word.
  - Meaningless space representation
- Word2Vec
  - Produce a vector space, with each unique word in the corpus being assigned a corresponding vector in the space (10 dimensions).

Features Vector

$$[Word_{-1}\ Word\ \ Word_{+1}\ Tag_{-2}\ Tag_{-1}\ isCapital\ suffixMatch\ ]$$

$Word, Word_{-1}, Word_{+1}$: current word and the two words nearby.

Tag_{-2} , Tag_{-1}: The two previous tags

$isCapital$: Binary feature, 1 if the word starts with a capital letter, 0 otherwise

$suffixMatch$: Binary feature, 1 if the word suffix match a pattern, 0 otherwise

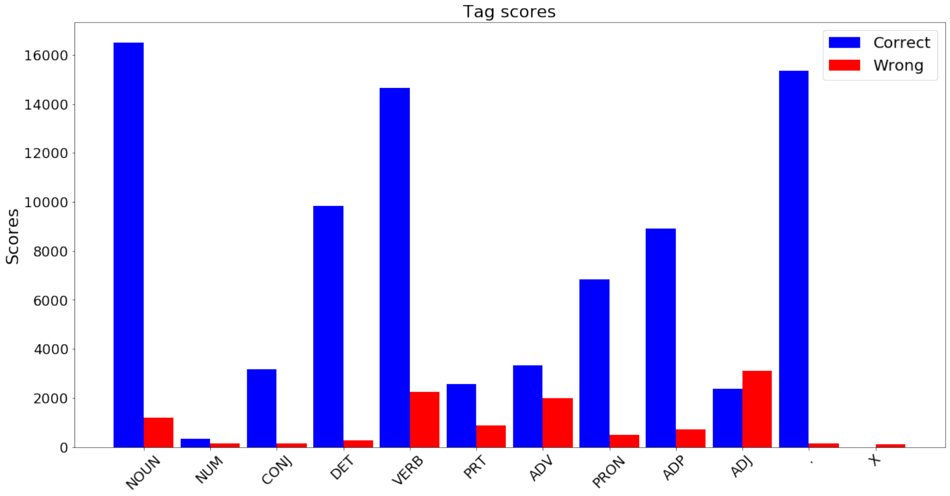
88.1%

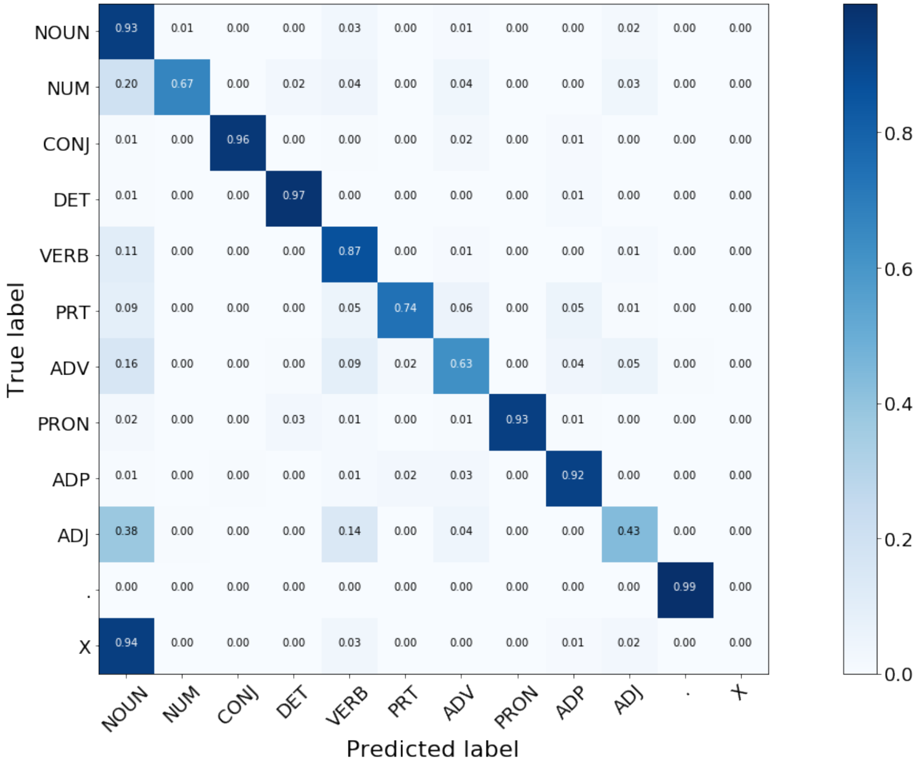
Accuracy

Train set length: 5000 sentences
Test set length: 5734

Precision: 88.2%
F-Measure: 87.7%

Results





	NOUN	NUM	CONJ	DET	VERB	PRT	ADV	PRON	ADP	ADJ	.	X
NOUN	0.93	0.01	0.00	0.00	0.03	0.00	0.01	0.00	0.00	0.02	0.00	0.00
NUM	0.20	0.67	0.00	0.02	0.04	0.00	0.04	0.00	0.00	0.03	0.00	0.00
CONJ	0.01	0.00	0.96	0.00	0.00	0.00	0.02	0.00	0.01	0.00	0.00	0.00
DET	0.01	0.00	0.00	0.97	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00
VERB	0.11	0.00	0.00	0.00	0.87	0.00	0.01	0.00	0.00	0.01	0.00	0.00
PRT	0.09	0.00	0.00	0.00	0.05	0.74	0.06	0.00	0.05	0.01	0.00	0.00
ADV	0.16	0.00	0.00	0.00	0.09	0.02	0.63	0.00	0.04	0.05	0.00	0.00
PRON	0.02	0.00	0.00	0.03	0.01	0.00	0.01	0.93	0.01	0.00	0.00	0.00
ADP	0.01	0.00	0.00	0.00	0.01	0.02	0.03	0.00	0.92	0.00	0.00	0.00
ADJ	0.38	0.00	0.00	0.00	0.14	0.00	0.04	0.00	0.00	0.43	0.00	0.00
.	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.99	0.00
X	0.94	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.01	0.02	0.00	0.00

4 Comparison

Comparison

PROs		CONs	
HMM	SVM	HMM	SVM
• Interpretability • Computationally low cost	• Addition of right features lead to better results • Space representation provide a better unknowns handling	• Limitation due to its structure • Unknown words handling • Out of the ordinary sentences bad predicted	• Interpretability • Computationally expensive

Thanks!

D'Amicis - Romeo

Cognitive Robotic Project – A.A. 2016-2017