НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ

Morphological disambiguation

Francis M. Tyers

ftyers@hse.ru
https://www.hse.ru/org/persons/209454856

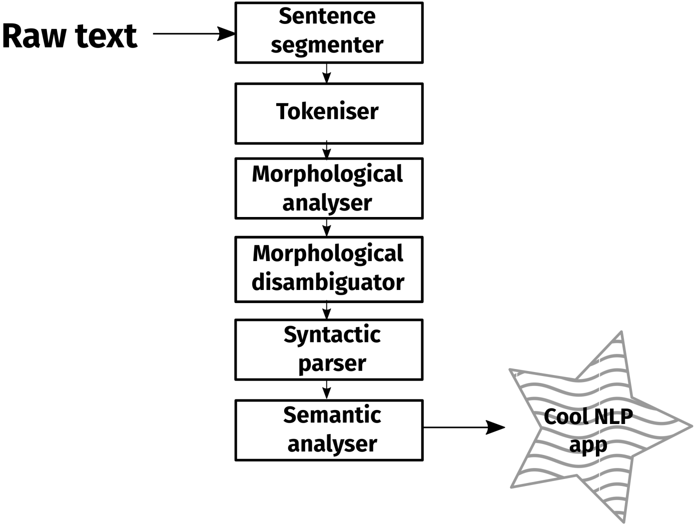
Национальный исследовательский университет
«Высшая школа экономики» (Москва)

6 марта 2018 г.

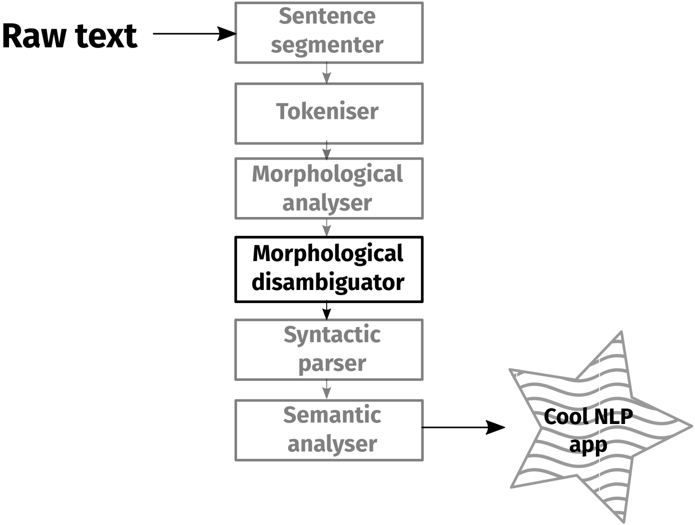
Introduction

- Introduction, tagsets
- Approaches
  - Rule-based
  - HMM-based
  - Averaged perceptron
- Discussion

Pipeline


Raw text
Sentence segmenter
Tokeniser
Morphological analyser
Morphological disambiguator
Syntactic parser
Semantic analyser
Cool NLP app

Pipeline



Raw text

Sentence segmenter

Tokeniser

Morphological analyser

Morphological disambiguator

Syntactic parser

Semantic analyser

Cool NLP app

Motivating example

при

Motivating example

при:

- *при* pr
- *пря* n f nn sg gen
- *пря* n f nn pl nom
- *пря* n f nn pl acc
- *переть* vblex impf tv imp p2 sg
- *переть* vblex impf iv imp p2 sg

Motivating example

Это я знал еще с 46-го года, когда начал писать, а может быть и раньше, – и факт этот не раз поражал меня и ставил меня в недоумение о полезности искусства **при** таком видимом его бессилии.

при:

- *при* pr
- *пря* n f nn sg gen
- *пря* n f nn pl nom
- *пря* n f nn pl acc
- *переть* vblex impf tv imp p2 sg
- *переть* vblex impf iv imp p2 sg

Applications

Aside from being a stage in the pipeline, what can use POS tagging directly ?

- **Speech synthesis**: How to pronounce a word in context, e.g. *conduct*
  - NOUN: /'kondukt/, VERB /kon'dukt/
- **Disambiguation of meaning**:
  - lie NOUN vs. lie VERB
- **Corpus linguistics**:
  - Find sequences of lexical categories
  - Limit searches for a wordform to a particular category

Terminology

Part-of-speech tagging:

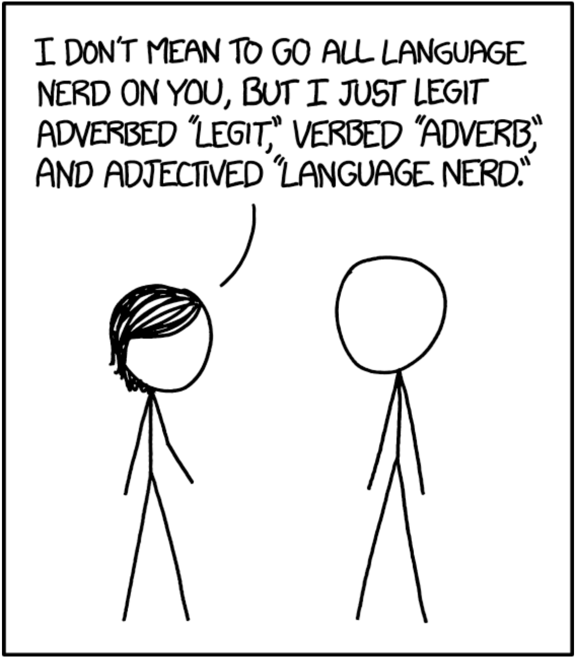
- Traditional term, based on approach(es) for English, finite-set of tags for all combinations of lexical category and morphology.

In:	This	is	a	test
	This/PRON	is/VERB	a/DET	test/NOUN

Morphological disambiguation:

- More cross-linguistically applicable, conception is of disambiguating after morphological analysis.

In:	This/DET/PRON	is/VERB	a/DET	test/VERB/NOUN
	This/PRON	is/VERB	a/DET	test/NOUN



I DON'T MEAN TO GO ALL LANGUAGE NERD ON YOU, BUT I JUST LEGIT ADVERBED "LEGIT," VERBED "ADVERB," AND ADJECTIVED "LANGUAGE NERD."

What is a tagset?

- Lemmas generalise over sets of inflectional forms
- Part-of-speech tags generalise over sets of lexemes/lemmas that have similar syntactic distribution

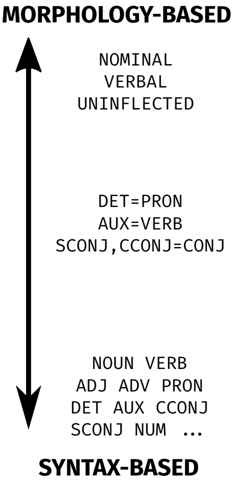
Tagset design

Examples:

- Splitting: Participles from adjectives
- Merging: One class for all nominals

Questions:

- Can the ambiguity be resolved?
- Does the distinction help downstream applications?



MORPHOLOGY-BASED

```
NOMINAL
VERBAL
UNINFLECTED
```

```
DET=PRON
AUX=VERB
SCONJ,CCONJ=CONJ
```

```
NOUN VERB
ADJ ADV PRON
DET AUX CCONJ
SCONJ NUM ...
```

SYNTAX-BASED

Example tagsets

Penn Treebank

```
This/DT
tagset/NNS
contains/VBZ
48/CD
unique/JJ
tags/NNP
./.
```

- 48 tags
- Tags are atomic
- Principles have been applied to other languages (Chinese, Bengali, …)
- Extensible ?

Example tagsets

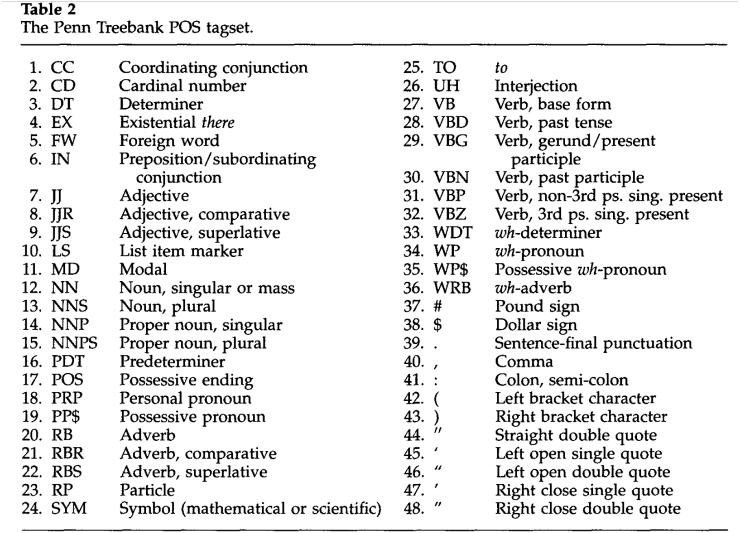
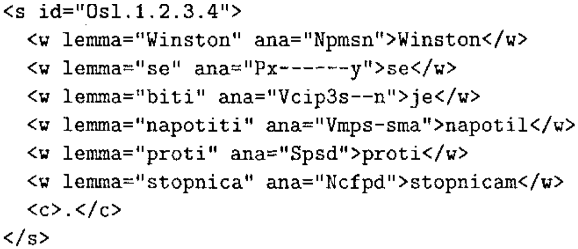


Table 2
The Penn Treebank POS tagset.

1.	CC	Coordinating conjunction	25.	TO	*to*
2.	CD	Cardinal number	26.	UH	Interjection
3.	DT	Determiner	27.	VB	Verb, base form
4.	EX	Existential *there*	28.	VBD	Verb, past tense
5.	FW	Foreign word	29.	VBG	Verb, gerund/present participle
6.	IN	Preposition/subordinating conjunction	30.	VBN	Verb, past participle
7.	JJ	Adjective	31.	VBP	Verb, non-3rd ps. sing. present
8.	JJR	Adjective, comparative	32.	VBZ	Verb, 3rd ps. sing. present
9.	JJS	Adjective, superlative	33.	WDT	*wh*-determiner
10.	LS	List item marker	34.	WP	*wh*-pronoun
11.	MD	Modal	35.	WP$	Possessive *wh*-pronoun
12.	NN	Noun, singular or mass	36.	WRB	*wh*-adverb
13.	NNS	Noun, plural	37.	#	Pound sign
14.	NNP	Proper noun, singular	38.	$	Dollar sign
15.	NNPS	Proper noun, plural	39.	.	Sentence-final punctuation
16.	PDT	Predeterminer	40.	,	Comma
17.	POS	Possessive ending	41.	:	Colon, semi-colon
18.	PRP	Personal pronoun	42.	(	Left bracket character
19.	PP$	Possessive pronoun	43.	)	Right bracket character
20.	RB	Adverb	44.	"	Straight double quote
21.	RBR	Adverb, comparative	45.	‘	Left open single quote
22.	RBS	Adverb, superlative	46.	“	Left open double quote
23.	RP	Particle	47.	’	Right close single quote
24.	SYM	Symbol (mathematical or scientific)	48.	”	Right close double quote

Example tagsets

Positional tags



```
<s id="Osl.1.2.3.4">
  <w lemma="Winston" ana="Npmsn">Winston</w>
  <w lemma="se" ana="Px------y">se</w>
  <w lemma="biti" ana="Vcip3s--n">je</w>
  <w lemma="napotiti" ana="Vmps-sma">napotil</w>
  <w lemma="proti" ana="Spsd">proti</w>
  <w lemma="stopnica" ana="Ncfpd">stopnicam</w>
  <c>.</c>
</s>
```

+ Compact
- Hard to read
- No support for derivational morphology

Example tagsets

Mnemonic tags

```
Sápmelaččas [sápmelaš] N Sg Loc
leai [leat] V IV Ind Prt Sg3
dakkár [dakkár] Pron Dem Attr
luondu [luondu] N Sg Nom
, [,] CLB
ahte [ahte] CS
son [son] Pron Pers Sg3 Nom
háliidišgođii [háliidit] V TV Der/goahti Ind Prt Sg3
gottiid [goddi] N Pl Acc
. [.] CLB
```

+ Easily handle derivations
+ Implicit morphological structure

- Number of tags can explode
- Modelling derivation is less language-independent

Example tagsets

Feature/value pairs

1	Польша	_	PROPN	_	Animacy=Inan|Case=Nom|Gender=Fem|Number=Sing	_
2	является	_	VERB	_	Aspect=Imp|Number=Sing|Person=3|Tense=Pres	_
3	безъядерной	_	ADJ	_	Animacy=Inan|Case=Ins|Gender=Fem|Number=Sing	_
4	страной	_	NOUN	_	Animacy=Inan|Case=Ins|Gender=Fem|Number=Sing	_
5	.	_	PUNCT	_	_	_

+ Easy to read
- No support for derivational morphology
- No implicit morphological structure
- Takes up a lot of space

Limitations

ӄорагыркиплыткогъат

- ӄора-гырки-плыткогъат
- reindeer-catch-finish-3PL
- *They finished catching reindeer.*

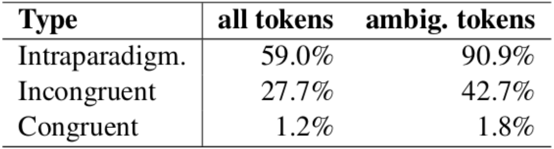
- Does VERB really capture what this is ?
- We can represent morphology easily, maybe even derivation
- But what about the incorporation ?

Scale of the problem

- UD corpora
- Percentage of tokens and types that receive more than one analysis
- Underestimation, e.g. Turkish *için*:
  + for.POST
  + inside.GEN
  - inside.2SG.NOM
  - drink.IMP.2PL

Language	Tokens	/type	/token
Turkish	58k	4.29	17.44
Finnish	201k	3.46	18.09
Kurmanji	10k	9.35	36.72
Basque	121k	11.47	38.47
Russian	1.1M	13.50	40.94
Erzya	2k	9.73	41.37
Norwegian	301k	8.28	43.78
Czech	1.5M	18.09	47.17
English	254k	14.20	52.34
German	292k	20.17	56.52
Portuguese	227k	13.19	64.51
Catalan	531k	8.31	66.49
Hebrew	161k	15.56	71.62
Hindi	351k	36.28	86.84

Types of ambiguity



Type	all tokens	ambig. tokens
Intraparadigm.	59.0%	90.9%
Incongruent	27.7%	42.7%
Congruent	1.2%	1.8%

- Intraparadigmatic:
  - ’тела (SG.GEN) vs. тел’а (PL.NOM)
- Morphosyntactically incongruent:
  - до’рога (NOUN) vs. дорог’а (ADJ)
- Morphosyntactically congruent:
  - ’замок (SG.NOM) vs. зам’ок (SG.NOM)

Approaches

- **Rule-based**
- **HMM-based**
- **Averaged perceptron**

Rule-based

Constraint Grammar

- Developed by Fred Karlsson[1] in the late 1980s
- Does not aim at producing a full “parse tree”
- Describes what is *ungrammatical*, not what is grammatical
- Linguists formalise “constraints” which describe language impossibilities
    - e.g. “No noun can be in prepositional case without a preposition which governs the prepositional case.”
- No “encapsulation”, all parts of the analysis (surface form → semantics) are always available
- Input is all possible analyses, output is only possible analyses

[1] The same Fred Karlsson that wrote “Finsk grammatik”.

Formalism

Input:

```
«Польша>”
    ”Польша”np top f sg nom
«является>”
    ”являться”v impf iv pres p3 sg
«безъядерной>”
    ”безъядерный”adj f an sg gen
    ”безъядерный”adj f an sg dat
    ”безъядерный”adj f an sg prp
    ”безъядерный”adj f an sg ins
«страной>”
    ”страна”n f nn sg ins
«.>”
    ”.”sent
```

Operators:

- select: Discard all readings except the reading matching a condition
- remove: Discard a single reading matching a condition

Context conditions:

- (-1 pres) → previous token has the tag PRES
- (1C ins) → following token *only* has the tag INS
- (NOT -1* pr) → no token to the left has the tag PR

Basic example

Input:

```
«Польша>”
    ”Польша”np top f sg nom
«является>”
    ”являться”v impf iv pres p3 sg
«безъядерной>”
    ”безъядерный”adj f an sg gen
    ”безъядерный”adj f an sg dat
    ”безъядерный”adj f an sg prp
    ”безъядерный”adj f an sg ins
«страной>”
    ”страна”n f nn sg ins
«.>”
    ”.”sent
```

Basic example

Input:

```
«Польша>”
    ”Польша”np top f sg nom
«является>”
    ”являться”v impf iv pres p3 sg
«безъядерной>”
    ”безъядерный”adj f an sg gen
    ”безъядерный”adj f an sg dat
    ”безъядерный”adj f an sg prp
    ”безъядерный”adj f an sg ins
«страной>”
    ”страна”n f nn sg ins
«.>”
    ”.”sent
```

```
REMOVE prp IF (not -1* pr)
```

Basic example

Input:

```
«Польша>”
    ”Польша”np top f sg nom
«является>”
    ”являться”v impf iv pres p3 sg
«безъядерной>”
    ”безъядерный”adj f an sg gen
    ”безъядерный”adj f an sg dat
    ”безъядерный”adj f an sg prp
    ”безъядерный”adj f an sg ins
«страной>”
    ”страна”n f nn sg ins
«.>”
    ”.”sent
```

1. REMOVE prp IF (not -1* pr)
2. REMOVE gen IF (-1 pres) (0C adj) (not 1 gen)

Basic example

Input:

```
«Польша>”
    ”Польша”np top f sg nom
«является>”
    ”являться”v impf iv pres p3 sg
«безъядерной>”
    ”безъядерный”adj f an sg gen
    ”безъядерный”adj f an sg dat
    ”безъядерный”adj f an sg prp
    ”безъядерный”adj f an sg ins
«страной>”
    ”страна”n f nn sg ins
«.>”
    ”.”sent
```

1. REMOVE prp IF (not -1* pr)
2. REMOVE gen IF (-1 pres) (0C adj) (not 1 gen)

Exercise: Can we safely remove the dative reading?

Less basic example

« Для соседних с Руандой государств руандийские события апреля – июля 1994 года вылились в огромное число прибывших беженцев . »

Standard trigram taggers

Для	PR
соседних	A=pl,gen,plen
с	PR
Руандой	S,f,inan=sg,ins
государств	S,n,inan=pl,gen
руандийские	A=pl,acc,inan,plen
события	S,n,inan=pl,acc
апреля	S,m,inan=sg,gen
–	–
июля	S,m,inan=sg,gen
1994	NUM=ciph
года	S,m,inan=sg,gen
вылились	V,pf,intr,med=pl,praet,indic
в	PR
огромное	A=n,sg,acc,inan,plen
число	S,n,inan=sg,acc
прибывших	V,pf,intr,act=partcp,pl,gen,praet,plen
беженцев	S,m,anim=pl,gen
.	.

Standard trigram taggers

Для	PR
соседних	A=pl,gen,plen
с	PR
Руандой	S,f,inan=sg,ins
государств	S,n,inan=pl,gen
руандийские	**A=pl,acc,inan,plen**
события	**S,n,inan=pl,acc**
апреля	S,m,inan=sg,gen
–	–
июля	S,m,inan=sg,gen
1994	NUM=ciph
года	S,m,inan=sg,gen
вылились	V,pf,intr,med=pl,praet,indic
в	PR
огромное	A=n,sg,acc,inan,plen
число	S,n,inan=sg,acc
прибывших	V,pf,intr,act=partcp,pl,gen,praet,plen
беженцев	S,m,anim=pl,gen
.	.

2 / 19 = 89.5% accuracy

Input: Morphological analysis

```
«Для>”
   ”для”pr
«соседних>”
   ”соседний”adj mfn an pl gen
   ”соседний”adj mfn an pl prp
   ”соседний”adj mfn aa pl acc
«с>”
   ”с”pr
«Руандой>”
   ”Руанда”np top f sg ins
«государств>”
   ”государство”n nt nn pl gen
«руандийские>”
   ”руандийский”adj mfn an pl nom
   ”руандийский”adj mfn nn pl acc
«события>”
   ”событие”n nt nn sg gen
   ”событие”n nt nn pl nom
   ”событие”n nt nn pl acc
«апреля>”
   ”апрель”n m nn sg gen
«->”
   ”-”guio
«июля>”
   ”июль”n m nn sg gen
```

```
«1994>”
   ”1994”num
«года>”
   ”год”n m nn sg gen
«вылились>”
   ”вылиться”v perf iv past mfn pl
«в>”
   ”в”pr
«огромное>”
   ”огромный”adj nt an sg nom
   ”огромный”adj nt an sg acc
«число>”
   ”число”n nt nn sg acc
   ”число”n nt nn sg nom
«прибывших>”
   ”прибыть”v perf iv pp actv mfn an pl acc
   ”прибыть”v perf iv pp actv mfn an pl prp
   ”прибыть”v perf iv pp actv mfn aa pl gen
«беженцев>”
   ”беженец”n m aa pl gen
   ”беженец”n m aa pl acc
«.>”
   ”.”sent
```

rule: –

Input: Morphological analysis

```
«Для>”
   ”для”pr
«соседних>”
   ”соседний”adj mfn an pl gen
   ”соседний”adj mfn an pl prp
   ”соседний”adj mfn aa pl acc
«с>”
   ”с”pr
«Руандой>”
   ”Руанда”np top f sg ins
«государств>”
   ”государство”n nt nn pl gen
«руандийские>”
   ”руандийский”adj mfn an pl nom
   ”руандийский”adj mfn nn pl acc
«события>”
   ”событие”n nt nn sg gen
   ”событие”n nt nn pl nom
   ”событие”n nt nn pl acc
«апреля>”
   ”апрель”n m nn sg gen
«->”
   ”-”guio
«июля>”
   ”июль”n m nn sg gen
```

```
«1994>”
   ”1994”num
«года>”
   ”год”n m nn sg gen
«вылились>”
   ”вылиться”v perf iv past mfn pl
«в>”
   ”в”pr
«огромное>”
   ”огромный”adj nt an sg nom
   ”огромный”adj nt an sg acc
«число>”
   ”число”n nt nn sg acc
   ”число”n nt nn sg nom
«прибывших>”
   ”прибыть”v perf iv pp actv mfn an pl acc
   ”прибыть”v perf iv pp actv mfn an pl prp
   ”прибыть”v perf iv pp actv mfn aa pl gen
«беженцев>”
   ”беженец”n m aa pl gen
   ”беженец”n m aa pl acc
«.>”
   ”.”sent
```

rule: 1

Input: Morphological analysis

```
«Для>”
   ”для”pr
«соседних>”
   ”соседний”adj mfn an pl gen
   ”соседний”adj mfn an pl prp
   ”соседний”adj mfn aa pl acc
«с>”
   ”с”pr
«Руандой>”
   ”Руанда”np top f sg ins
«государств>”
   ”государство”n nt nn pl gen
«руандийские>”
   ”руандийский”adj mfn an pl nom
   ”руандийский”adj mfn nn pl acc
«события>”
   ”событие”n nt nn sg gen
   ”событие”n nt nn pl nom
   ”событие”n nt nn pl acc
«апреля>”
   ”апрель”n m nn sg gen
«->”
   ”-”guio
«июля>”
   ”июль”n m nn sg gen
```

```
«1994>”
   ”1994”num
«года>”
   ”год”n m nn sg gen
«вылились>”
   ”вылиться”v perf iv past mfn pl
«в>”
   ”в”pr
«огромное>”
   ”огромный”adj nt an sg nom
   ”огромный”adj nt an sg acc
«число>”
   ”число”n nt nn sg acc
   ”число”n nt nn sg nom
«прибывших>”
   ”прибыть”v perf iv pp actv mfn an pl acc
   ”прибыть”v perf iv pp actv mfn an pl prp
   ”прибыть”v perf iv pp actv mfn aa pl gen
«беженцев>”
   ”беженец”n m aa pl gen
   ”беженец”n m aa pl acc
«.>”
   ”.”sent
```

rule: 2

Input: Morphological analysis

```
«Для>”
   ”для”pr
«соседних>”
   ”соседний”adj mfn an pl gen
   ”соседний”adj mfn an pl prp
   ”соседний”adj mfn aa pl acc
«с>”
   ”с”pr
«Руандой>”
   ”Руанда”np top f sg ins
«государств>”
   ”государство”n nt nn pl gen
«руандийские>”
   ”руандийский”adj mfn an pl nom
   ”руандийский”adj mfn nn pl acc
«события>”
   ”событие”n nt nn sg gen
   ”событие”n nt nn pl nom
   ”событие”n nt nn pl acc
«апреля>”
   ”апрель”n m nn sg gen
«->”
   ”-”guio
«июля>”
   ”июль”n m nn sg gen
```

```
«1994>”
   ”1994”num
«года>”
   ”год”n m nn sg gen
«вылились>”
   ”вылиться”v perf iv past mfn pl
«в>”
   ”в”pr
«огромное>”
   ”огромный”adj nt an sg nom
   ”огромный”adj nt an sg acc
«число>”
   ”число”n nt nn sg acc
   ”число”n nt nn sg nom
«прибывших>”
   ”прибыть”v perf iv pp actv mfn an pl acc
   ”прибыть”v perf iv pp actv mfn an pl prp
   ”прибыть”v perf iv pp actv mfn aa pl gen
«беженцев>”
   ”беженец”n m aa pl gen
   ”беженец”n m aa pl acc
«.>”
   ”.”sent
```

rule: 3

Input: Morphological analysis

```
«Для>”
   ”для”pr
«соседних>”
   ”соседний”adj mfn an pl gen
   ”соседний”adj mfn an pl prp
   ”соседний”adj mfn aa pl acc
«с>”
   ”с”pr
«Руандой>”
   ”Руанда”np top f sg ins
«государств>”
   ”государство”n nt nn pl gen
«руандийские>”
   ”руандийский”adj mfn an pl nom
   ”руандийский”adj mfn nn pl acc
«события>”
   ”событие”n nt nn sg gen
   ”событие”n nt nn pl nom
   ”событие”n nt nn pl acc
«апреля>”
   ”апрель”n m nn sg gen
«->”
   ”-”guio
«июля>”
   ”июль”n m nn sg gen
```

```
«1994>”
   ”1994”num
«года>”
   ”год”n m nn sg gen
«вылились>”
   ”вылиться”v perf iv past mfn pl
«в>”
   ”в”pr
«огромное>”
   ”огромный”adj nt an sg nom
   ”огромный”adj nt an sg acc
«число>”
   ”число”n nt nn sg acc
   ”число”n nt nn sg nom
«прибывших>”
   ”прибыть”v perf iv pp actv mfn an pl acc
   ”прибыть”v perf iv pp actv mfn an pl prp
   ”прибыть”v perf iv pp actv mfn aa pl gen
«беженцев>”
   ”беженец”n m aa pl gen
   ”беженец”n m aa pl acc
«.>”
   ”.”sent
```

rule: 4

Input: Morphological analysis

```
«Для>”
   ”для”pr
«соседних>”
   ”соседний”adj mfn an pl gen
   ”соседний”adj mfn an pl prp
   ”соседний”adj mfn aa pl acc
«с>”
   ”с”pr
«Руандой>”
   ”Руанда”np top f sg ins
«государств>”
   ”государство”n nt nn pl gen
«руандийские>”
   ”руандийский”adj mfn an pl nom
   ”руандийский”adj mfn nn pl acc
«события>”
   ”событие”n nt nn sg gen
   ”событие”n nt nn pl nom
   ”событие”n nt nn pl acc
«апреля>”
   ”апрель”n m nn sg gen
«->”
   ”-”guio
«июля>”
   ”июль”n m nn sg gen
```

```
«1994>”
   ”1994”num
«года>”
   ”год”n m nn sg gen
«вылились>”
   ”вылиться”v perf iv past mfn pl
«в>”
   ”в”pr
«огромное>”
   ”огромный”adj nt an sg nom
   ”огромный”adj nt an sg acc
«число>”
   ”число”n nt nn sg acc
   ”число”n nt nn sg nom
«прибывших>”
   ”прибыть”v perf iv pp actv mfn an pl acc
   ”прибыть”v perf iv pp actv mfn an pl prp
   ”прибыть”v perf iv pp actv mfn aa pl gen
«беженцев>”
   ”беженец”n m aa pl gen
   ”беженец”n m aa pl acc
«.>”
   ”.”sent
```

rule: 5

Input: Morphological analysis

```
«Для>”
   ”для”pr
«соседних>”
   ”соседний”adj mfn an pl gen
   ”соседний”adj mfn an pl prp
   ”соседний”adj mfn aa pl acc
«с>”
   ”с”pr
«Руандой>”
   ”Руанда”np top f sg ins
«государств>”
   ”государство”n nt nn pl gen
«руандийские>”
   ”руандийский”adj mfn an pl nom
   ”руандийский”adj mfn nn pl acc
«события>”
   ”событие”n nt nn sg gen
   ”событие”n nt nn pl nom
   ”событие”n nt nn pl acc
«апреля>”
   ”апрель”n m nn sg gen
«->”
   ”-”guio
«июля>”
   ”июль”n m nn sg gen
```

rule: 5

```
«1994>”
   ”1994”num
«года>”
   ”год”n m nn sg gen
«вылились>”
   ”вылиться”v perf iv past mfn pl
«в>”
   ”в”pr
«огромное>”
   ”огромный”adj nt an sg nom
   ”огромный”adj nt an sg acc
«число>”
   ”число”n nt nn sg acc
   ”число”n nt nn sg nom
«прибывших>”
   ”прибыть”v perf iv pp actv mfn an pl acc
   ”прибыть”v perf iv pp actv mfn an pl prp
   ”прибыть”v perf iv pp actv mfn aa pl gen
«беженцев>”
   ”беженец”n m aa pl gen
   ”беженец”n m aa pl acc
«.>”
   ”.”sent
```

Proposed rule (I)

1 Immediately after “для” remove any reading which is in a case other than genitive.

Exceptions:

- None ?

Formalised:

```
LIST Gen = gen ;
SET NGDAIP = nom OR gen OR dat OR acc OR ins OR prp ;
REMOVE NGDAIP - Gen IF (-1C ("для")) ;
```

Proposed rule (II)

2 After “в” remove any reading which is in nominative

Exceptions:

- Joining an organisation ?

Formalised:

```
LIST Nom = nom ;
REMOVE Nom IF (-1C ("в")) ;
```

Proposed rule (III)

3. In a sentence with a single intransitive finite verb, remove any reading in accusative which is not immediately governed by a preposition

Exceptions:

- There is a trans. part. form having an acc. arg.
- Some adverbial forms... *Мы проехали километр.*

Formalised:

```
LIST IV = iv ;
LIST TV = tv ;
LIST Acc = acc ;
LIST Pr = pr ;
REMOVE Acc IF (0 Acc LINK NOT -1* Pr) ((-1* IV) OR (1*
IV)) (0 Acc LINK NOT 1* TV) (0 Acc LINK NOT -1* TV) ;
```

Proposed rule (IV)

4 Select nominative if there is an intransitive verb which agrees with a nominative noun in the sentence for number (and/or gender)

- and is preceeded by an adj. that can only be nom.
- and there is no other nom. head in the sentence.

Exceptions:

- Appositions, titles, parentheticals ? Non-canonical agreement ?

Formalised:

```
LIST Head = np n prn ;
SET NUM = (sg) OR (pl) ;
SELECT Nom + $$NUM IF (-1C A + Nom) (NOT -1* Head + Nom)
(NOT 1* Head + Nom) ((-1* V + $$NUM) OR (1* V + $$NUM));
```

Proposed rule (V)

5 If there is a prepositional case reading, remove it if you see a noun which is only in a case other than prepositional without any preceeding transitive participle form

Exceptions:

- ...

Formalised:

```
LIST Prp = prp ;
LIST N = n ;
REMOVE Prp IF (-1* N + NGDAIP - Prp) ;
```

Output: Morphologically-disambiguated text

```
«Для>"
   "для"pr
«соседних>"
   "соседний"adj mfn an pl gen
   "соседний"adj mfn an pl prp
   "соседний"adj mfn aa pl acc
«с>"
   "с"pr
«Руандой>"
   "Руанда"np top f sg ins
«государств>"
   "государство"n nt nn pl gen
«руандийские>"
   "руандийский"adj mfn an pl nom
   "руандийский"adj mfn nn pl acc

«события>"
   "событие"n nt nn sg gen
   "событие"n nt nn pl nom
   "событие"n nt nn pl acc

«апреля>"
   "апрель"n m nn sg gen
«->"
   "-"guio
«июля>"
   "июль"n m nn sg gen
```

```
«1994>"
   "1994"num
«года>"
   "год"n m nn sg gen
«вылились>"
   "вылиться"v perf iv past mfn pl
«в>"
   "в"pr
«огромное>"
   "огромный"adj nt an sg nom
   "огромный"adj nt an sg acc
«число>"
   "число"n nt nn sg acc
   "число"n nt nn sg nom

«прибывших>"
   "прибыть"v perf iv pp actv mfn an pl acc
   "прибыть"v perf iv pp actv mfn an pl prp
   "прибыть"v perf iv pp actv mfn aa pl gen
«беженцев>"
   "беженец"n m aa pl gen
   "беженец"n m aa pl acc

«.>"
   "."sent
```

Examples

Languages with constraint grammars:

- Finnish
- North Sámi, Lule Sámi, South Sámi
- Norwegian (Nynorsk, Bokmål)
- Faroese
- Udmurt
- Breton

HMM-based

Predict hidden states from observed events

- hidden states = sequence of part of speech tags
- observed events = ambiguity classes or surface forms

$$M = (A, B, \pi)$$

- A = transition probabilities
- B = emission probabilities
- π = initial probabilities

Visible events

The visible events can be either:

- **Surface forms:** In many traditional HMM-based taggers, the visible events are surface forms
- **Ambiguity classes:** Generalisation over types of ambiguity e.g. NOUN/VERB, DET/PRON

Example:

Surface forms:	This	is	a	test	.
Ambig. classes:	PRON/DET	VERB/AUX	DET	VERB/NOUN	PUNCT

A tagged corpus

Analysed:

Vino/NOUN/VERB a/ADP la/DET/PRON playa/NOUN ./PUNCT
Voy/VERB a/ADP la/DET/PRON casa/NOUN/VERB ./PUNCT
Bebe/VERB vino/NOUN/VERB en/ADP casa/NOUN/VERB ./PUNCT
La/DET/PRON casa/NOUN/VERB es/VERB grande/ADJ ./PUNCT
Es/VERB una/DET/PRON/VERB ciudad/NOUN grande/ADJ ./PUNCT

Tagged:

Vino/VERB a/ADP la/DET playa/NOUN ./PUNCT
Voy/VERB a/ADP la/DET casa/NOUN ./PUNCT
Bebe/VERB vino/NOUN en/ADP casa/NOUN ./PUNCT
La/DET casa/NOUN es/VERB grande/ADJ ./PUNCT
Es/VERB una/DET ciudad/NOUN grande/ADJ ./PUNCT

Transition probabilities

We calculate the transition probabilities, *A* from a matrix of transition counts:

	Second tag						
	VERB	NOUN	DET	PRON	ADP	ADJ	PUNCT
VERB	0	1	1	0	2	1	0
NOUN	1	0	0	0	1	1	3
DET	0	4	0	0	0	0	0
PRON	0	0	0	0	0	0	0
ADP	0	1	2	0	0	0	0
ADJ	0	0	0	0	0	0	2
PUNCT	3	0	1	0	0	0	0

Transition probabilities

We calculate the transition probabilities, *A* from a matrix of transition counts:

	Second tag						
	VERB	NOUN	DET	PRON	ADP	ADJ	PUNCT
VERB	0	1	1	0	2	1	0
NOUN	1	0	0	0	1	1	3
DET	0	4	0	0	0	0	0
PRON	0	0	0	0	0	0	0
ADP	0	1	2	0	0	0	0
ADJ	0	0	0	0	0	0	2
PUNCT	3	0	1	0	0	0	0

Transition probabilities

We calculate the transition probabilities, *A* from a matrix of transition counts:

	Second tag						
	VERB	NOUN	DET	PRON	ADP	ADJ	PUNCT
VERB	**0**	**1**	**1**	**0**	**2**	**1**	**0**
NOUN	1	0	0	0	1	1	3
DET	0	4	0	0	0	0	0
PRON	0	0	0	0	0	0	0
ADP	0	1	2	0	0	0	0
ADJ	0	0	0	0	0	0	2
PUNCT	3	0	1	0	0	0	0

Transition probabilities

We calculate the transition probabilities, *A* from a matrix of transition counts:

	Second tag						
	VERB	NOUN	DET	PRON	ADP	ADJ	PUNCT
VERB	0	0.2	0.2	0	0.4	0.2	0
NOUN	0.16	0	0	0	0.16	0.16	0.5
DET	0	1	0	0	0	0	0
PRON	0	0	0	0	0	0	0
ADP	0	0.3	0.6	0	0	0	0
ADJ	0	0	0	0	0	0	1
PUNCT†	0.75	0	0.25	0	0	0	0

† This row represents the initial probabilities, π of the model.

Transition probabilities

We calculate the transition probabilities, *A* from a matrix of transition counts:

	Second tag						
	VERB	NOUN	DET	PRON	ADP	ADJ	PUNCT
VERB	0	0.2	0.2	0	0.4	0.2	0
NOUN	0.16	0	0	0	0.16	0.16	0.5
DET	0	1	0	0	0	0	0
PRON	0	0	0	0	0	0	0
ADP	0	0.3	0.6	0	0	0	0
ADJ	0	0	0	0	0	0	1
PUNCT†	0.75	0	0.25	0	0	0	0

† This row represents the initial probabilities, π of the model.

A tagged corpus

Analysed:

Vino/NOUN/VERB a/ADP la/DET/PRON playa/NOUN ./PUNCT
Voy/VERB a/ADP la/DET/PRON casa/NOUN/VERB ./PUNCT
Bebe/VERB vino/NOUN/VERB en/ADP casa/NOUN/VERB ./PUNCT
La/DET/PRON casa/NOUN/VERB es/VERB grande/ADJ ./PUNCT
Es/VERB una/DET/PRON/VERB ciudad/NOUN grande/ADJ ./PUNCT

Tagged:

Vino/VERB a/ADP la/DET playa/NOUN ./PUNCT
Voy/VERB a/ADP la/DET casa/NOUN ./PUNCT
Bebe/VERB vino/NOUN en/ADP casa/NOUN ./PUNCT
La/DET casa/NOUN es/VERB grande/ADJ ./PUNCT
Es/VERB una/DET ciudad/NOUN grande/ADJ ./PUNCT

Emission probabilities

The probability of seeing an ambiguity class given a tag, *B*.

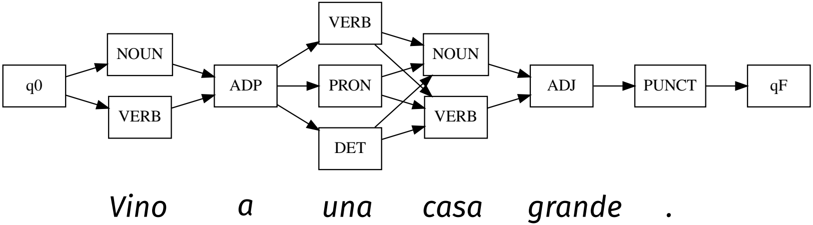
	VERB	NOUN	DET	PRON	ADP	ADJ	PUNCT
ADJ	0	0	0	0	0	0	2
DET/PRON	0	0	3	0	0	0	0
DET/PRON/VERB	0	0	1	0	0	0	0
NOUN	0	2	0	0	0	0	0
NOUN/VERB	1	4	0	0	0	0	0
ADP	0	0	0	0	3	0	0
PUNCT	0	0	0	0	0	0	5
VERB	4	0	0	0	0	0	0
Total:	5	6	4	0	3	2	5

Emission probabilities

The probability of seeing an ambiguity class given a tag, *B*.

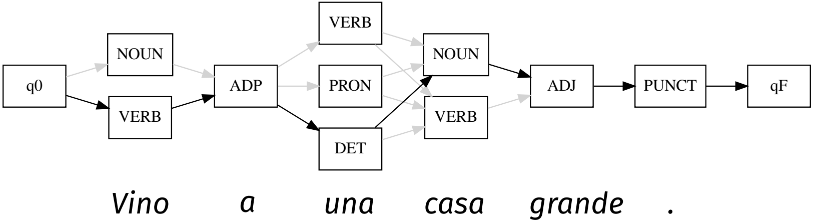
	VERB	NOUN	DET	PRON	ADP	ADJ	PUNCT
ADJ	0	0	0	0	0	1.0	0
DET/PRON	0	0	0.75	0	0	0	0
DET/PRON/VERB	0	0	0.25	0	0	0	0
NOUN	0	0.33	0	0	0	0	0
NOUN/VERB	0.2	0.67	0	0	0	0	0
ADP	0	0	0	0	1.0	0	0
PUNCT	0	0	0	0	0	0	1.0
VERB	0.8	0	0	0	0	0	0

Viterbi



- Dynamic programming algorithm
- Find the most likely sequence of hidden states given observed sequence
- e.g. Find POS tag sequence given words or ambiguity classes

Viterbi



- Dynamic programming algorithm
- Find the most likely sequence of hidden states given observed sequence
- e.g. Find POS tag sequence given words or ambiguity classes

Decoding

Vino a una casa grande.

→

q_F							
VERB							
NOUN							
DET							
PRON							
ADP							
ADJ							
PUNCT							
	VERB/NOUN	ADP	DET/PRON/VERB	NOUN/VERB	ADJ	PUNCT	
	Vino	*a*	*una*	*casa*	*grande*	.	

Decoding

→

q_F							
VERB	0.15, q_0						
NOUN	0.0, q_0						
DET							
PRON							
ADP							
ADJ							
PUNCT							
	VERB/NOUN	ADP	DET/PRON/VERB	NOUN/VERB	ADJ	PUNCT	
	Vino	*a*	*una*	*casa*	*grande*	.	

- = P(VERB,PUNCT) * P(VERB, VERB/NOUN) = 0.75 * 0.2 = 0.15
- = P(NOUN,PUNCT) * P(NOUN, VERB/NOUN) = 0.0 * 0.67 = 0.0

Decoding

→

q_F							
VERB	0.15, q_0						
NOUN	0.0, q_0						
DET							
PRON							
ADP		0.06, VERB					
ADJ							
PUNCT							
	VERB/NOUN	ADP	DET/PRON/VERB	NOUN/VERB	ADJ	PUNCT	
	Vino	*a*	*una*	*casa*	*grande*	.	

- = P(ADP,VERB) * P(ADP, ADP) * P(PATH) = 0.4 * 1.0 * 0.15 = 0.06
- = P(ADP,NOUN) * P(ADP, ADP) * P(PATH) = 0.16 * 1.0 * 0.0 = 0

Decoding

→

q_F							
VERB	0.15, q_0		0.0, ADP				
NOUN	0.0, q_0						
DET			0.009, ADP				
PRON			0.0, ADP				
ADP		0.06, VERB					
ADJ							
PUNCT							
	VERB/NOUN	ADP	DET/PRON/VERB	NOUN/VERB	ADJ	PUNCT	
	Vino	*a*	*una*	*casa*	*grande*	.	

- = P(DET,ADP) * P(DET, DET/PRON/VERB) * P(PATH) = 0.6 * 0.25 * 0.06 = 0.009
- = P(PRON,ADP) * P(PRON, DET/PRON/VERB) * P(PATH) = 0.0 * 0.0 * 0.06 = 0.0
- = P(VERB,ADP) * P(VERB, DET/PRON/VERB) * P(PATH) = 0.0 * 0.0 * 0.06 = 0.0

Decoding

→

q_F							
VERB	0.15, q_0		0.0, ADP	0.0, DET*			
NOUN	0.0, q_0			0.006, DET			
DET			0.009, ADP				
PRON			0.0, ADP				
ADP		0.06, VERB					
ADJ							
PUNCT							
	VERB/NOUN	ADP	DET/PRON/VERB	NOUN/VERB	ADJ	PUNCT	
	Vino	*a*	*una*	*casa*	*grande*	.	

- = P(NOUN,VERB) * P(NOUN, NOUN/VERB) * P(PATH) = 0.2 * 0.67 * 0.009 = 0.001
- = P(VERB,VERB) * P(VERB, NOUN/VERB) * P(PATH) = 0.0 * 0.2 * 0.009 = 0.0
- = P(NOUN,DET) * P(NOUN, NOUN/VERB) * P(PATH) = 1.0 * 0.67 * 0.009 = 0.006
- = P(VERB,DET) * P(VERB, NOUN/VERB) * P(PATH) = 0.0 * 0.2 * 0.009 = 0.0
- = P(NOUN,PRON) * P(NOUN, NOUN/VERB) * P(PATH) = 0.0 * 0.67 * 0.009 = 0.0
- = P(VERB,PRON) * P(VERB, NOUN/VERB) * P(PATH) = 0.0 * 0.67 * 0.009 = 0.0

Decoding

→

q_F							
VERB	0.15, q_0		0.0, ADP	0.0, DET*			
NOUN	0.0, q_0			0.006, DET			
DET			0.009, ADP				
PRON			0.0, ADP				
ADP		0.06, VERB					
ADJ					0.001, NOUN		
PUNCT							
	VERB/NOUN	ADP	DET/PRON/VERB	NOUN/VERB	ADJ	PUNCT	
	Vino	*a*	*una*	*casa*	*grande*	.	

- = P(ADJ,NOUN) * P(ADJ, ADJ) * P(PATH) = 0.16 * 1.0 * 0.006 = 0.00096
- = P(ADJ,VERB) * P(ADJ, ADJ) * P(PATH) = 0.2 * 1.0 * 0.0 = 0.0

Decoding

→

q_F							
VERB	0.15, q_0		0.0, ADP	0.0, DET*			
NOUN	0.0, q_0			0.006, DET			
DET			0.009, ADP				
PRON			0.0, ADP				
ADP		0.06, VERB					
ADJ					0.001, NOUN		
PUNCT						0.001, ADJ	
	VERB/NOUN	ADP	DET/PRON/VERB	NOUN/VERB	ADJ	PUNCT	
	Vino	*a*	*una*	*casa*	*grande*	.	

- = P(PUNCT,ADJ) * P(PUNCT, PUNCT) * P(PATH) = 1.0 * 1.0 * 0.001 = 0.001

Decoding

→

q_F							
VERB	0.15, q_0		0.0, ADP	0.0, DET*			
NOUN	0.0, q_0			0.006, DET			
DET			0.009, ADP				
PRON			0.0, ADP				
ADP		0.06, VERB					
ADJ					0.001, NOUN		
PUNCT						0.001, ADJ	
	VERB/NOUN	ADP	DET/PRON/VERB	NOUN/VERB	ADJ	PUNCT	
	Vino	*a*	*una*	*casa*	*grande*	.	

PUNCT

Decoding

→

q_F							
VERB	0.15, q_0		0.0, ADP	0.0, DET*			
NOUN	0.0, q_0			0.006, DET			
DET			0.009, ADP				
PRON			0.0, ADP				
ADP		0.06, VERB					
ADJ					0.001, NOUN		
PUNCT						0.001, **ADJ**	
	VERB/NOUN	ADP	DET/PRON/VERB	NOUN/VERB	ADJ	PUNCT	
	Vino	*a*	*una*	*casa*	*grande*	.	

ADJ PUNCT

Decoding

→

q_F							
VERB	0.15, q_0		0.0, ADP	0.0, DET*			
NOUN	0.0, q_0			0.006, DET			
DET			0.009, ADP				
PRON			0.0, ADP				
ADP		0.06, VERB					
ADJ					0.001, **NOUN**		
PUNCT						0.001, ADJ	
	VERB/NOUN	ADP	DET/PRON/VERB	NOUN/VERB	ADJ	PUNCT	
	Vino	*a*	*una*	*casa*	*grande*	*.*	

NOUN ADJ PUNCT

Decoding

→

q_F							
VERB	0.15, q_0		0.0, ADP	0.0, DET*			
NOUN	0.0, q_0			0.006, **DET**			
DET			0.009, ADP				
PRON			0.0, ADP				
ADP		0.06, VERB					
ADJ					0.001, NOUN		
PUNCT						0.001, ADJ	
	VERB/NOUN	ADP	DET/PRON/VERB	NOUN/VERB	ADJ	PUNCT	
	Vino	*a*	*una*	*casa*	*grande*	.	

DET NOUN ADJ PUNCT

Decoding

→

q_F							
VERB	0.15, q_0		0.0, ADP	0.0, DET*			
NOUN	0.0, q_0			0.006, DET			
DET			0.009, **ADP**				
PRON			0.0, ADP				
ADP		0.06, VERB					
ADJ					0.001, NOUN		
PUNCT						0.001, ADJ	
	VERB/NOUN	ADP	DET/PRON/VERB	NOUN/VERB	ADJ	PUNCT	
	Vino	*a*	*una*	*casa*	*grande*	.	

ADP DET NOUN ADJ PUNCT

Decoding

→

q_F							
VERB	0.15, q_0		0.0, ADP	0.0, DET*			
NOUN	0.0, q_0			0.006, DET			
DET			0.009, ADP				
PRON			0.0, ADP				
ADP		0.06, **VERB**					
ADJ					0.001, NOUN		
PUNCT						0.001, ADJ	
	VERB/NOUN	ADP	DET/PRON/VERB	NOUN/VERB	ADJ	PUNCT	
	Vino	*a*	*una*	*casa*	*grande*	.	

VERB ADP DET NOUN ADJ PUNCT

Decoding

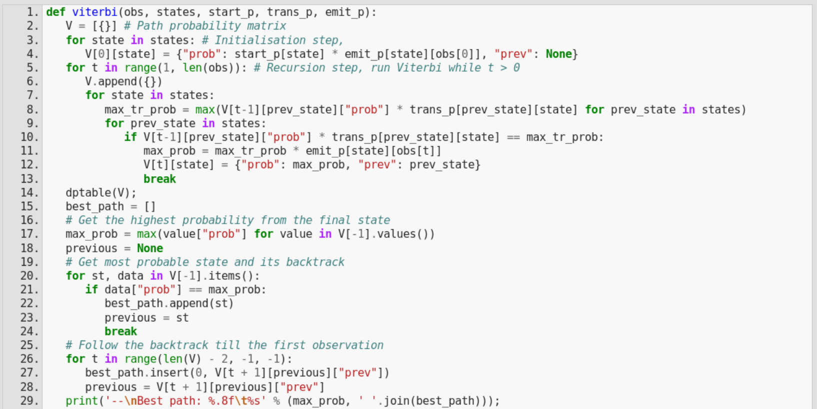
→

q_F							
VERB	0.15, q_0		0.0, ADP	0.0, DET*			
NOUN	0.0, q_0			0.006, DET			
DET			0.009, ADP				
PRON			0.0, ADP				
ADP		0.06, VERB					
ADJ					0.001, NOUN		
PUNCT						0.001, ADJ	
	VERB/NOUN	ADP	DET/PRON/VERB	NOUN/VERB	ADJ	PUNCT	
	Vino	*a*	*una*	*casa*	*grande*	.	

VERB ADP DET NOUN ADJ PUNCT

Implementation

https://paste2.org/HMgn7amd



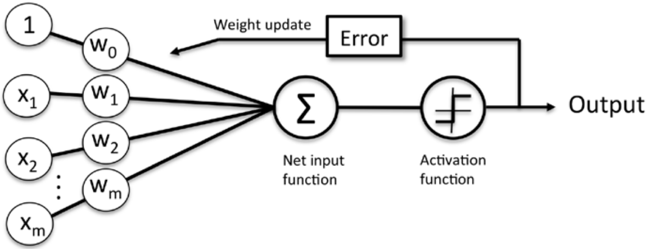
```python
def viterbi(obs, states, start_p, trans_p, emit_p):
   V = [{}] # Path probability matrix
   for state in states: # Initialisation step,
      V[0][state] = {"prob": start_p[state] * emit_p[state][obs[0]], "prev": None}
   for t in range(1, len(obs)): # Recursion step, run Viterbi while t > 0
      V.append({})
      for state in states:
         max_tr_prob = max(V[t-1][prev_state]["prob"] * trans_p[prev_state][state] for prev_state in states)
         for prev_state in states:
            if V[t-1][prev_state]["prob"] * trans_p[prev_state][state] == max_tr_prob:
               max_prob = max_tr_prob * emit_p[state][obs[t]]
               V[t][state] = {"prob": max_prob, "prev": prev_state}
               break
   dptable(V);
   best_path = []
   # Get the highest probability from the final state
   max_prob = max(value["prob"] for value in V[-1].values())
   previous = None
   # Get most probable state and its backtrack
   for st, data in V[-1].items():
      if data["prob"] == max_prob:
         best_path.append(st)
         previous = st
         break
   # Follow the backtrack till the first observation
   for t in range(len(V) - 2, -1, -1):
      best_path.insert(0, V[t + 1][previous]["prev"])
      previous = V[t + 1][previous]["prev"]
   print('--\nBest path: %.8f\t%s' % (max_prob, ' '.join(best_path)));
```

Extensions

- Trigrams
  - Instead of conditioning on previous tag, condition on previous two
- Unknown words
  - Incorporate suffixes into the tags
- Backoff
  - If the bi-/tri-gram hasn't been seen, backoff to lower order model
- Capitalisation
  - Use capitalisation features

Averaged perceptron

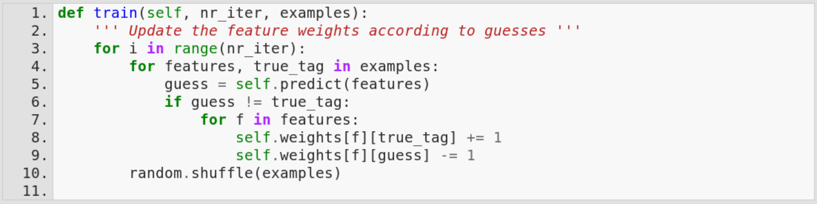
A binary perceptron:



- Discriminative model ... find the category, not the distribution
- Beautifully simple

`github.com/ftyers/conllu-perceptron-tagger`

Training



```python
def train(self, nr_iter, examples):
    ''' Update the feature weights according to guesses '''
    for i in range(nr_iter):
        for features, true_tag in examples:
            guess = self.predict(features)
            if guess != true_tag:
                for f in features:
                    self.weights[f][true_tag] += 1
                    self.weights[f][guess] -= 1
        random.shuffle(examples)

```

- We iterate through the whole training data *n* times
- For each tag we try and predict the value
    - If we get it wrong, we increase the weight of the features for the correct class

Features

Vino	a	una	casa	grande	.
$i-2$	$i-1$	i	$i+1$	$i+2$	$i+3$

- Specify whatever features you want,
- Easy to add new ones!

i	Trigram suffix	una
i	Unigram prefix	u
$i-1$	Tag	ADP
$i-2$	Tag	VERB
i	Word	una
$i-1, i$	Tag, Word	ADP + una
$i-1$	Word	a
$i-1$	Trigram suffix	a
$i-2$	Word	Vino
$i+1$	Word	casa
$i+1$	Trigram suffix	asa
$i+2$	Word	grande

Averaging

Problem:

- In later iterations it changes the weights a lot for the last few samples it is getting wrong
- …overfitting

Solution:

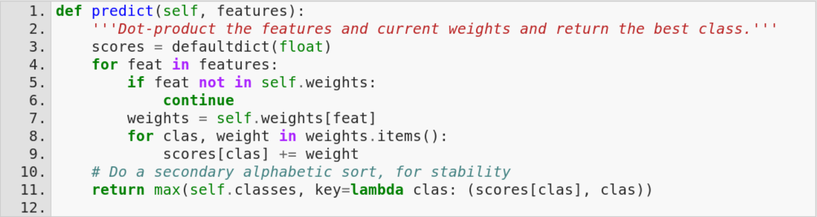
- Average the weights over the iterations

Dictionary

Given the amount of unambiguous words:

- Make a dictionary
- When you see the word, output it
- But have a frequency threshold, e.g. 20

Prediction



```python
def predict(self, features):
    '''Dot-product the features and current weights and return the best class.'''
    scores = defaultdict(float)
    for feat in features:
        if feat not in self.weights:
            continue
        weights = self.weights[feat]
        for clas, weight in weights.items():
            scores[clas] += weight
    # Do a secondary alphabetic sort, for stability
    return max(self.classes, key=lambda clas: (scores[clas], clas))

```

- For each class (POS), add the weights of the features we've seen
- Take the class with the maximum weight

Discussion

Comparison of approaches

	+	-
CG	Start from scratch	Tagset not learnt
HMM	Model distribution	Hard to incorporate feats
Perceptron	Easy to incorporate feats	No *n*-best

All techniques can reach 97% token accuracy.

- Great ... but 57% full-sentence accuracy.

Comparison of approaches

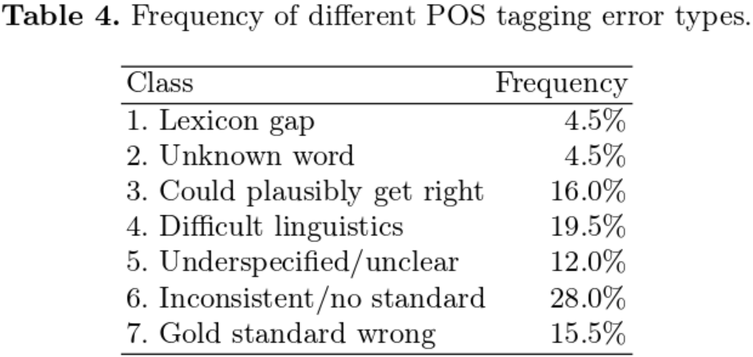


Table 4. Frequency of different POS tagging error types.

Class	Frequency
1. Lexicon gap	4.5%
2. Unknown word	4.5%
3. Could plausibly get right	16.0%
4. Difficult linguistics	19.5%
5. Underspecified/unclear	12.0%
6. Inconsistent/no standard	28.0%
7. Gold standard wrong	15.5%

Chris Manning (2011) “Part-of-Speech Tagging from 97% to 100%: Is It Time for Some Linguistics?”

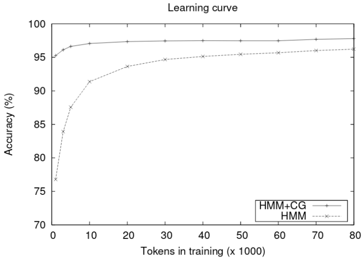
How much time and effort ?

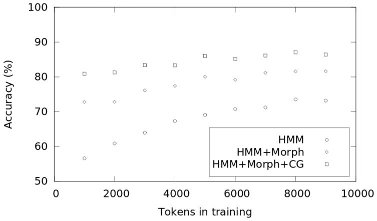
Annotation time vs. rule-writing time

- Hand annotation: 8000–10000 tokens/month
  - 50–100k tokens = 6–12 months
- Rule-based:
  - Morphological analyser: 3–6 months
  - Constraint grammar: 3–6 months

Approaching a new language, depends on what you like doing more.

Tagger combination





- Voting systems
- Combine systems that make complementary errors

Some taggers

Russian:

- `pymorphy2`
- `mystem3`

Trainable:

- HunPos (HMM, OCaml)
- UDPipe/MorphoDiTa (Av. Perceptron, C++)
- MarMot (CRF, Java)
- NLTK (various, Python)

Practicals

https://ftyers.github.io/2017-КЛ_МКЛ/practicals/disambiguation.html

- **Tagger comparison**:
  - Compare three taggers on a language/domain of your choice
- **Constraint grammar**:
  - Select a small text (one paragraph) in a language of your choice
  - Analyse it with a morphological analyser
  - Resolve as much of the ambiguity as you can
- **Perceptron tagger**:
  - Download https://github.com/ftyers/conllu-perceptron-tagger
  - Run it on a language from Universal Dependencies
  - Improve it so that you get better performance
    - Add support for morphological features
    - Try tweaking other features