Gépi tanulás

Számítógépes nyelvészet – 2018 tavasz
9. óra

Simon Eszter – Mittelholcz Iván

2018. április 25.

MTA Nyelvtudományi Intézet

Tartalom

Racionalista és empirikus megközelítés

Racionalista és empirikus megközelítés

Racionalista:

- szabályalapú
- a nyelvész írja a szabályokat $\rightarrow$ a nyelvi információt expliciten adja át a számítógépnek

Empirikus:

- statisztikai alapú
- a számítógépes nyelvész az erőforrásokat adja oda a számítógépnek, ami azokat felhasználva magát tanítja

Kérdés: amikor számítógépes nyelvmodellt építünk, akkor korpuszadatokra vagy introspekcióra támaszkodunk?

A szabályalapú rendszerek előnyei és hátrányai

Előnyei:

- a fejlesztő nagyobb kontrollja a rendszer felett
- könnyebben értelmezhető visszacsatolás a rendszertől
- magas pontosság

Hátrányai:

- sok kézi munkát és nagy szakértelmet kíván
- nem hibatűrő
- bonyolult a fejlesztése, törékeny
- nehezen vihető át más nyelvre/doménre

A statisztikai alapú rendszerek előnyei és hátrányai

Előnyei:

- minden elemzéshez egy valószínűséget kapunk → rangsorolhatjuk őket → kiválaszthatjuk a leginkább odaillőt
- még akkor is adhat jó eredményt, ha a mögöttes nyelvmodell nem adekvát
- sokkal flexibilisebb megközelítés

Hátrányai:

- nagy mennyiségű annotált adatot igényelnek → a kézi munka nem tűnt el, csak átalakult
- a rendszer átvitele más nyelvre/doménre elég nagy teljesítménybeli visszaesést okoz

Szabályalapú rendszerek

- internal and external evidence: Magyarország és Svájc
- patternök, amelyek a nyelvi elem belső szerkezetét írják le & környezetfüggő szabályok, amelyek a kontextusról szólnak
- listák
- Java Annotation Patterns Engine (JAPE): annotációk fölött értelmezett reguláris kifejezésekből épít FST-ket
- számít a szabályok sorrendje

Példaszabályok

személynév

family_name+ *given_name*+ → PERSON_NAME

given_name → PERSON_NAME

family_name → PERSON_NAME

generational_prefix PERSON_NAME → PERSON_NAME

[IVX]+\. *regnal_name* → PERSON_NAME

DE:

Abdalláh ibn Abdal-Muttalib

Visvanath Pratap Szingh

Szabályalapú rendszerek – folyt.

- bizonyos kifejezéseknek (pl. dátumok) olyan transzparens belső szerkezetük van, hogy reguláris kifejezésekkel könnyen és hatékonyan felismerhetők → a statisztikai alapú rendszerek is alkalmaznak ilyen bináris jegyeket (pl. `isDate=True`)
- magas pontosság, alacsony fedés
- a fedést a listák méretének növelésével és újabb szabályok felvételével lehet növelni
- lehetetlen olyan patternöket írni, amik mindent lefednek, ami kell, de semmit, ami nem kell
- lehet súlyozni a szabályokat → hibrid rendszer

A statisztikai módszerek kategorizálása

- felügyelt (supervised)
- félig felügyelt (semi-supervised)
- felügyelet nélküli (unsupervised)

Felügyelet nélküli és félig felügyelt tanulás

Felügyelet nélküli tanulás

előfeldolgozott szöveg címkék nélkül → kérdés: mit lehet megtanulni a nyers szövegből?

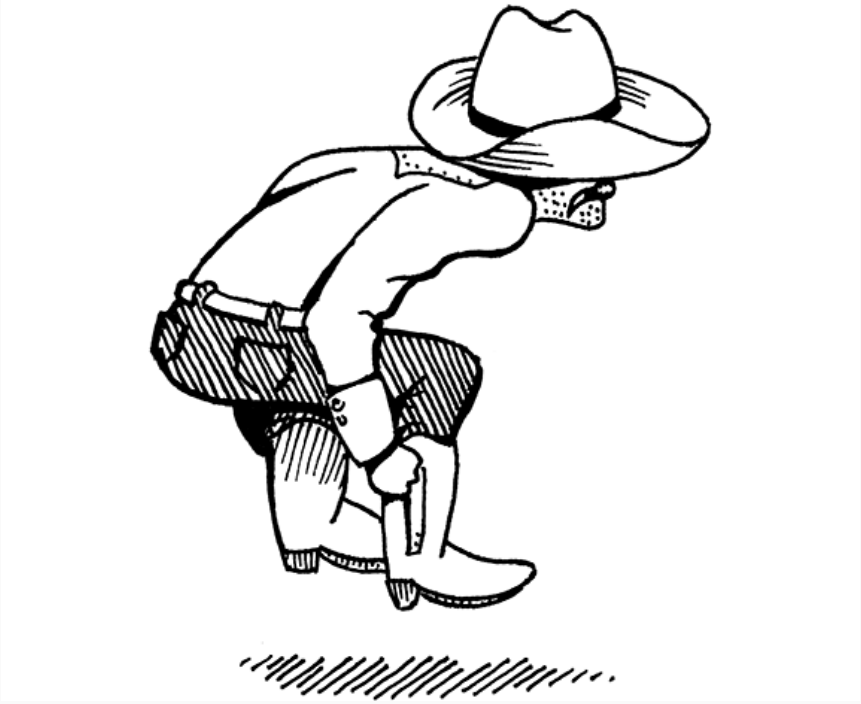
Klaszterezés:

a hasonló grammatikai viselkedésű elemek fognak osztályokba csoportosulni

1. nincsenek előre definiált osztályok
   - ha új típusokat akarunk találni
2. előre megszabjuk az osztályok számát
   - ha az adott feladatban megszokott osztályok szerint akarjuk kiértékelni

Félig felügyelt tanulás

- címkézetlen szövegből tanul
- kézzel összeállított kiinduló halmaz (’seed’)
- bootstrapping
- az adatban előforduló természetes redundanciára építenek



A felügyelt gépi tanulás menete

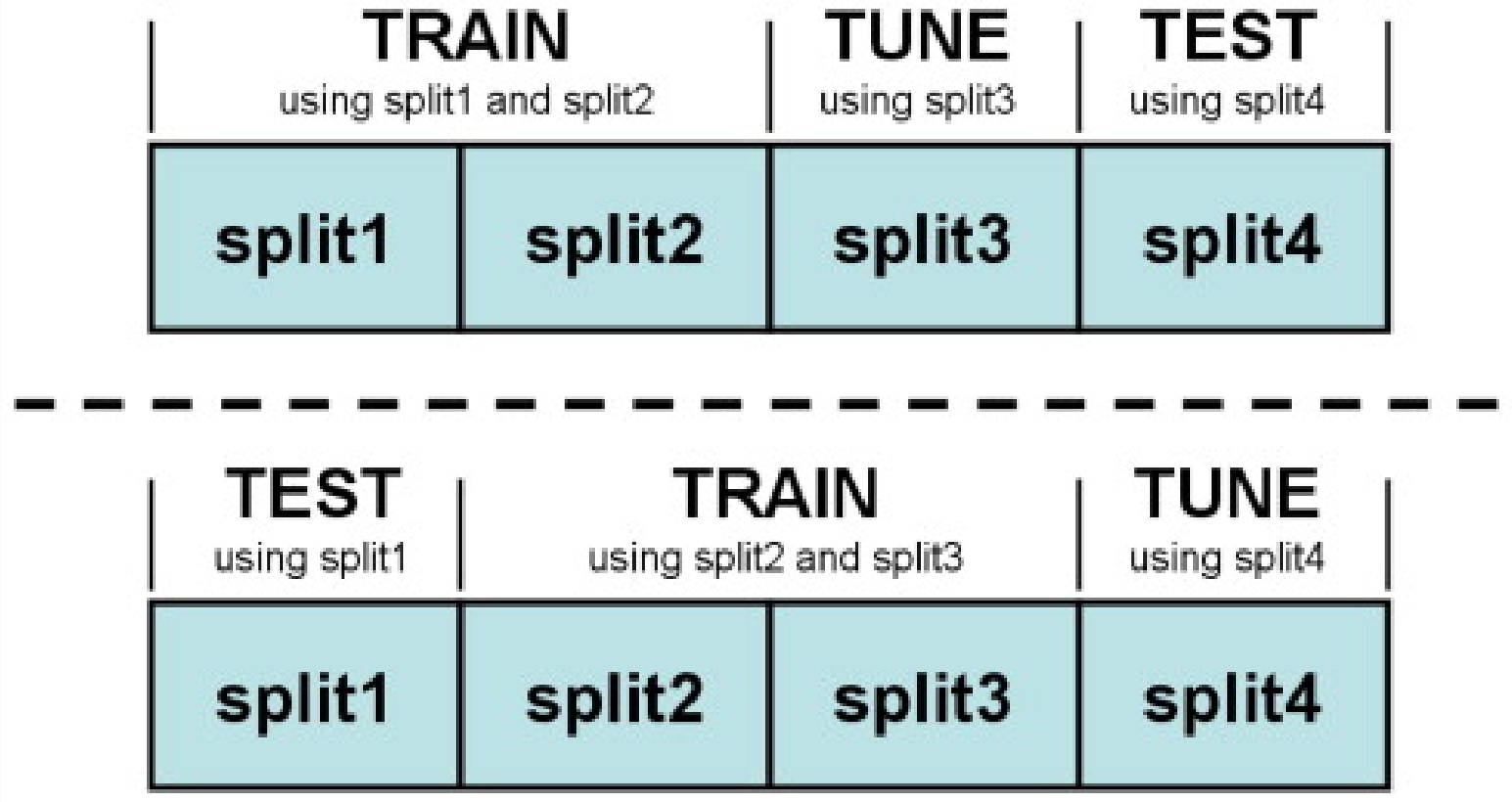
A felügyelt gépi tanulás alapvetése

- a felügyelt gépi tanulás azon a feltételezésen alapul, hogy az adatpontok egymástól független elemek, amelyeknek egyenletes az eloszlásuk
- feltesszük, hogy az eddig nem látott adatpontokra is igaz ez $\rightarrow$ így tudunk következtetni a már látott nyelvi elemekből a még nem látottakra
- nyelvi annotációval ellátott korpusz $\rightarrow$ ebből tanulja ki az adott adatpontokra jellemző jegyeket, és ez szolgál majd a kiértékelés alapjául is

Általános séma

1. gold standard korpusz
2. train–devel–test halmaz, keresztvalidáció
3. jegykinyerés
4. modellépítés
5. taggelés
6. kiértékelés

Train–devel–test & keresztvalidáció



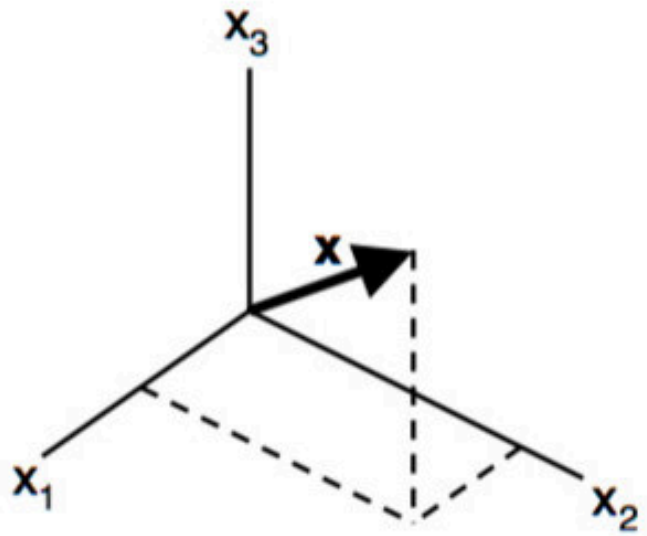
Túltanulás (overfitting)

- a teszthalmaz elemei nem szerepelhetnek a tanító halmazban
- fejlesztés közben mérni csak a develen szabad
- ha nem így teszünk, akkor az nem csak csalás, de a rendszerünk nem lesz képes általánosításokra $\rightarrow$ túlzottan rátanul az adott szövegre
- a rendszer a tanító halmazban levő random zajt prezentálja, nem általánosít

Jegykinyerés (feature extraction)

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \\ x_d \end{bmatrix}$$

Feature vector



Feature space (3D)

További lépések

modellépítés:

jegy–címke párok, mindegyikhez egy súly hozzárendelve, ami azt mutatja meg, hogy az adott jegy mennyire van hatással arra, hogy az adott jeggyel rendelkező token az adott címkét kapja

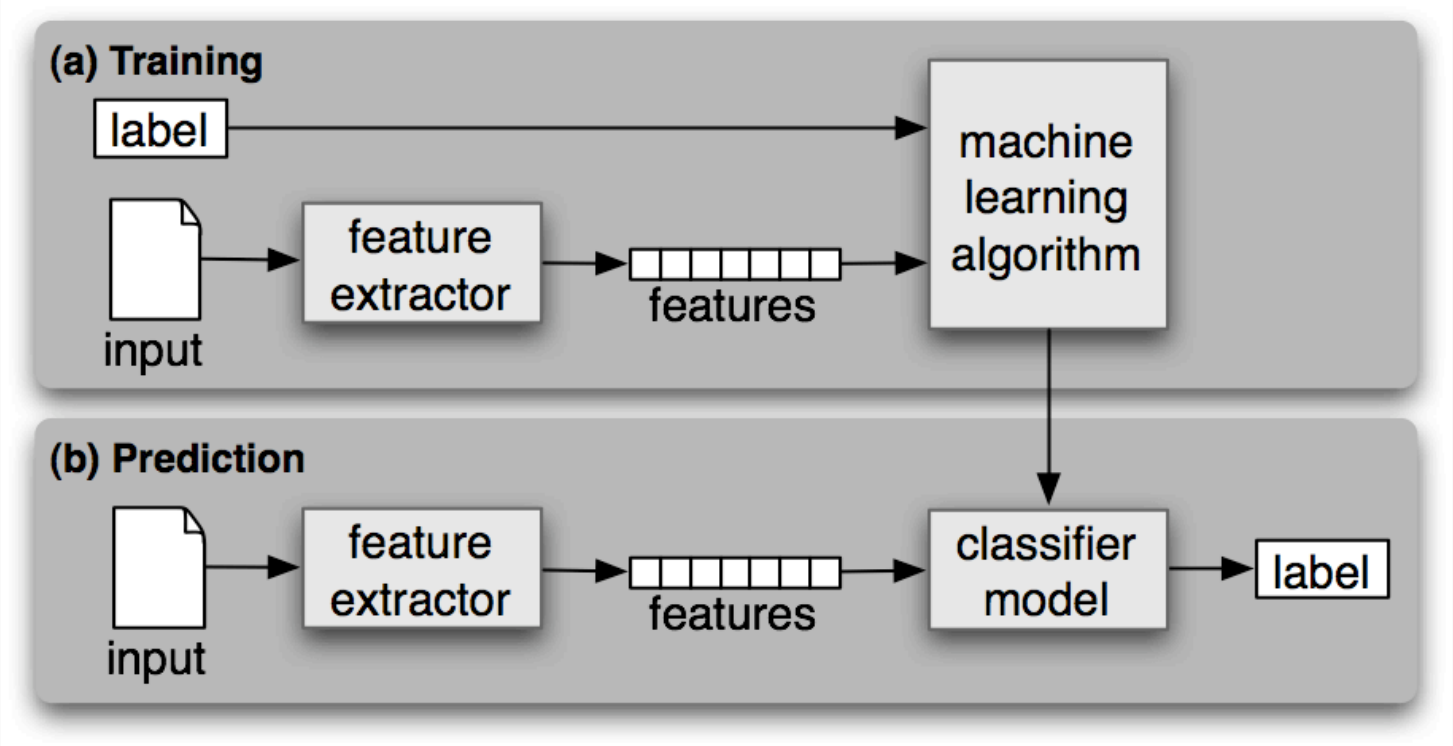
taggelés:

1. a kiértékelő halmaz fícsörizálása
2. a kapott fícsörvektorok és a nyelvmodell alapján címkék kibocsátása

baseline:

a rendszertől minimálisan elvárt teljesítmény $\rightarrow$ jellemzően a leggyakoribb címke kiosztása minden adatpontra

Áttekintés


(a) Training
label
input
feature extractor
features
machine learning algorithm
(b) Prediction
input
feature extractor
features
classifier model
label

A HunTag architektúrája

Szekvenciális vs. tokenalapú címkézési feladat

szekvenciális

lánc-struktúra $\rightarrow$ időbeli vagy térbeli egymásutániság (pl. szavak egy mondatban) $\rightarrow$ a szekvencia minden egyes eleméhez címkét kell rendelni

tokenalapú

egy adatponthoz (egy tokenhez) egy címkét kell rendelni

Klasszifikáció vs. regresszió

klasszifikáció

kategóriacímkét bocsát ki (két vagy több diszkrét kategória)

regresszió

valós számot bocsát ki (folytonos)

logisztikus regresszió

valós számot bocsát ki, ami valószínűségként értelmezhető (két kategória)

multinomiális logisztikus regresszió

valós számot bocsát ki, ami valószínűségként értelmezhető (több kategória)

Maximum entrópia

- multinomiális logisztikus regresszió
- tokenalapú
- címkék fölötti teljes valószínűség-eloszlást bocsát ki

token	gold	C1	P1	C2	P2
tájékoztatta	O	O	1		
a	O	O	1		
Magyar	B-ORG	B-ORG	0.92	1-ORG	0.08
Nemzeti	I-ORG	I-ORG	0.96	O	0.04
Bank	E-ORG	E-ORG	1		
csütörtökön	O	O	1		
az	O	O	1		
MTI-t	1-ORG	B-ORG	0.35	1-ORG	0.65

Rejtett Markov-modell (Hidden Markov Model, HMM)

Definíció (HMM)

$Q = q_1 q_2 ... q_N$: a HMM állapotainak véges halmaza

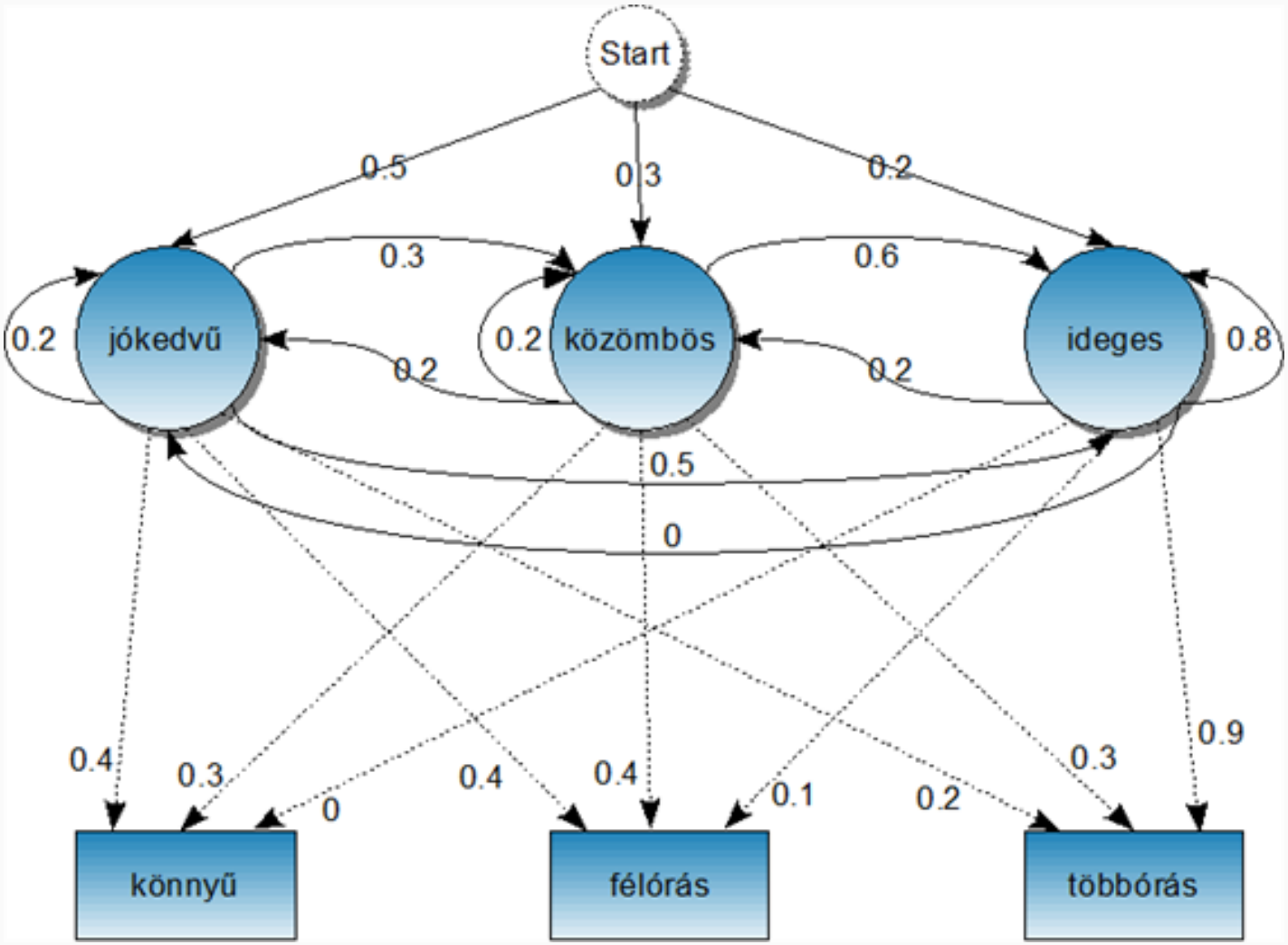
$A = a_{01} a_{02} ... a_{n1} ... a_{nn}$: átmenetvalószínűségek halmaza (mátrix)

$O = o_1 o_2 ... o_T$: a megfigyelt jelenségek halmaza

$B = b_i(o_t)$: megfigyelési/kibocsátási valószínűségek (egy o_t megfigyelés q_i állapotból kibocsátva)

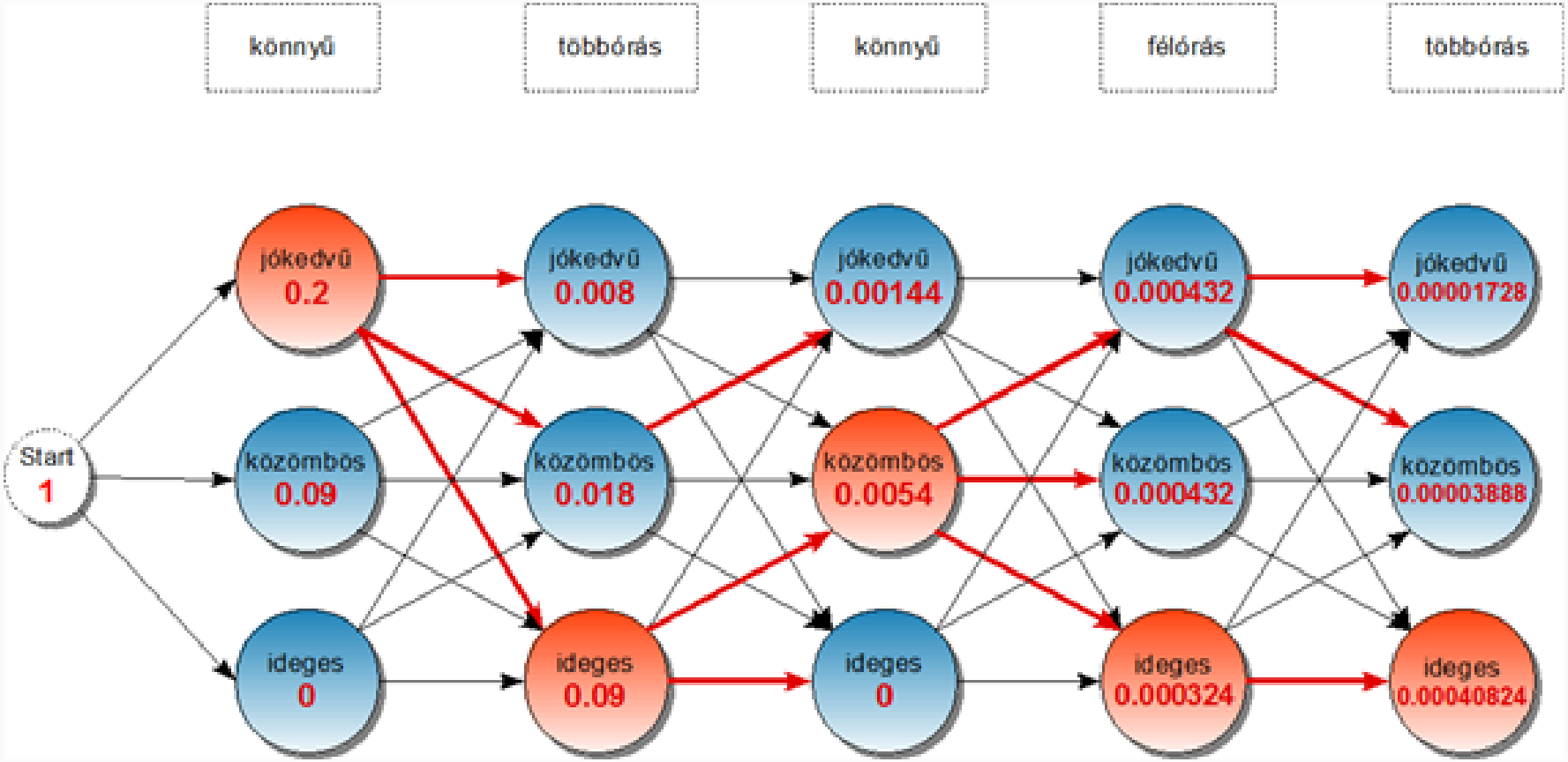
π: kezdeti valószínűség-eloszlás

Példa HMM


Start
0.5
0.3
0.2
0.3
0.6
0.2
jókedvű
közömbös
ideges
0.2
0.2
0.8
0.2
0.2
0.5
0
0.4
0.3
0
0.4
0.4
0.1
0.2
0.3
0.9
könnyű
félórás
többórás

Viterbi-algoritmus

maximalizálja az együttes átmenet–kibocsátási valószínűséget



Mit ad nekünk a HMM?

- tokenalapú $\rightarrow$ szekvenciális címkézés
- nyelvmodell (language model) & megfigyelési modell (observation model)
- nem feltétlenül a lokális maximum a jó
- implicit konzisztenciagarancia $\rightarrow$ jobb teljesítmény

n-gramok

George W. Bush

karakter *n*-gramok

unigramok: G, e, o, r, g, e, _, W, ., _, B, u, s, h

bigramok: Ge, eo, or, rg, ge, e_, _W, W., ._, _B, Bu, us, sh

trigramok: Geo, eor, org, rge, ge_, e_W, _W., W._, ._B, _Bu, Bus, ush

szó *n*-gramok

unigramok: George, W., Bush

bigramok: George W., W. Bush

trigramok: George W. Bush

címke *n*-gramok

unigramok: B-PER, I-PER, E-PER

bigramok: B-PER I-PER, I-PER E-PER

trigramok: B-PER I-PER E-PER

Rádiusz

- az aktuálisan vizsgált token nemcsak a saját magára vonatkozó fícsör–érték párokat kapja meg, hanem az előtte–utána levő *n* token fícsör–érték párjait is
- erre való a radius
- default NER: 3, NP: 5, de a config fájlban állítható, akár egyes fícsörökre külön-külön is
- ezzel a kontextus is figyelembe vehető

Cutoff

- eldobja azokat a fícsör–érték párokat, amik a megadott cutoff értéknél kevesebbszer fordulnak elő a teljes tanítóhalmaz felfícsörizálása után
- kiszűrjük a random zajt, de nagyobb tanítóanyagra lesz szükségünk

iscap

`cutoff = 2`

`iscap=True`: 15

`iscap=False`: 1

$\rightarrow$ `iscap=False` fícsör–érték párt eldobjuk

Jegymérnökség (feature engineering)

Mik a jegyek?

- a jegyek az adatpontok különféle tulajdonságait írják le
- a jegyeket a számítógépes nyelvész találja ki, definiálja és kódolja le (a “hagyományos” gépi tanulás során)
- a jegy hasznosságát az adat fogja meghatározni $\rightarrow$ a jegy megkülönböztető erejét ki kell mérni $\rightarrow$ utána lehet dönteni a jegy alkalmazásáról

A jegyek megkülönböztető ereje

- jegyek hozzáadása vagy paraméterek állítása egyesével $\rightarrow$ mérés $\rightarrow$ ha nem ront, akkor bennehagyjuk
- több jegyszelekciós módszer is létezik
  - a legegyszerűbb algoritmus: a fícsörtér minden lehetséges részhalmazát kimérni, és azt választani, amelyiknél a legnagyobb az F-mérték
  - a fícsörtér inkrementális bővítése (önkényesen vagy korábbi kísérletek eredményei alapján)

Jegytípusok

forrás

- felszíni tulajdonságok
- morfológiai információk
- szintaktikai információk
- listatagság

érték

- sztringértékű
- bináris

egység

- token
- mondat

Sztringértékű felszíni jegyek

- `word form` → 90,68%
- `ngram` (1,**2**,3...)
- `prefix` (...4,**5**,6...)
- `suffix` (...3,4,5,6...)
- `longpatt`
- `shortpatt`

Bináris felszíni jegyek

`hascap`: the token contains one or more upper case letters, e.g. *EasyJet*;

`allcaps`: contains only upper case characters, e.g. *XP*;

`capperiod`: comprises one upper case letter and a period, e.g. *A.*;

`camel`: is in camel case, e.g. *iPad*;

`3caps`: comprises three upper case characters, e.g. *IBM*;

`iscap`: its initial letter is upper case, e.g. *London*.

Morfológiai jegyek

`lemma`: the token's lemma;
`fulltag`: full morphological analysis;
`pos`: only the POS tag;
`tagend`: information about inflection;
`oov`: out-of-vocabulary word;
`tagpattern`: pattern of POS tags in a sentence;
`isbetweensamecases`: whether the inspected token's neighbours have the same case marking as the token itself;
`penntags`: an abstraction over Penn tags which groups together similar tags, e.g. tags starting with *VB* and *MD* get the value 'verb';
`plural`: whether the token is in the plural form.

Szintaktikai jegyek

`sentstart`: if the token is in sentence starting position, it gets the feature `sentstart=1`;

`sentend`: if it is in sentence ending position, it gets the feature `sentend=1`;

`chunktag`: the whole chunk tag, e.g. *B-NP*;

`chunktype`: the type of the chunk, e.g. *NP*;

`chunkpart`: the part of the tag which is indicating its position in a chunk, e.g. *B*;

`NpPart`: if it is a part of a NP, it gets the feature `nppart=1`;

`parsePatts`: pattern of chunk labels in the sentence in 5 and 7 words windows.

Listás jegyek

- ‘is a’ relációt fejez ki: ha a ‘Budapest’ név benne van a városok listájában, akkor Budapest egy város
- a listatagság erős indikátor szokott lenni
- a szabályalapú rendszerek nagyon függnek a listáktól, a statisztikaiaik sokkal kevésbé
- a fedés nagyon szoros összefüggésben van a listák méretével $\rightarrow$ ...

References

Jurafsky, D. and Martin, J. H. (2000). *Speech and Language Processing*. Prentice Hall, first edition.

Simon, E. (2013). *Approaches to Hungarian Named Entity Recognition*. PhD thesis, PhD School in Cognitive Psychology, Budapest University of Technology and Economics.

Javasolt olvasmányok:

- Jurafsky and Martin (2000)
- Simon (2013)

https://www.nng.com/hanagyleszek

MACHINE LEARNING SZAKÉRTŐ

Rendőr, postás, pék is lennék,
kertésznek is vígan mennék,
de leginkább azért főleg
machine learning szakértőnek.

Nem törődnék semmi mással,
mint a gépi tanulással.
Megtanítanám a gépem,
hogy kell viselkedni szépen.

A férfiaktól a nőket
hogy különböztesse ő meg,
s mi egymástól nem áll távol:
a muffint a csivavától.

Ha ráunt a kiskutyákra,
emberekkel diskurálna,
ámuldozna ám a jónép,
milyen okos számítógép!

Tanítgatnám, nevelgetném,
adatokkal etetgetném,
s ha már kapott elég ételt,
ronggyá verné Lékó Pétert.

Én lennék a soselátott,
bablevesbe belemártott,
sakkozókat kiborító
számítógép idomító!

machine learning *[mesín lörning] (ang.)* gépi tanulás