DIGITÁLIS BÖLCSÉSZETI KÖZPONT

Bevezetés a nyelvtechnológiába

Simon Eszter

2020. szeptember 23.

PIM DBK

Az előadás vázlata

1. Bevezetés
2. A nyelvtechnológia módszerei
3. A korpuszok
4. A nyelvfeldolgozás lépései
5. Szövegfeldolgozó eszközláncok magyarra

Bevezetés

Szinonimák?

- számítógépes nyelvészet
- természetesnyelv-feldolgozás (natural language processing, NLP)
- nyelvtechnológia (human language technology, HLT)

Besorolás


nyelvészet
számítógépes nyelvészet
számítástechnika
villamosmérnöki tudomány
beszédfelismerés
mesterséges intelligencia
természetes nyelv feldolgozás
számítógépes nyelvészet
számítógépes pszicholingvisztika
pszichológia
számítógépes nyelvészet
nyelvészet

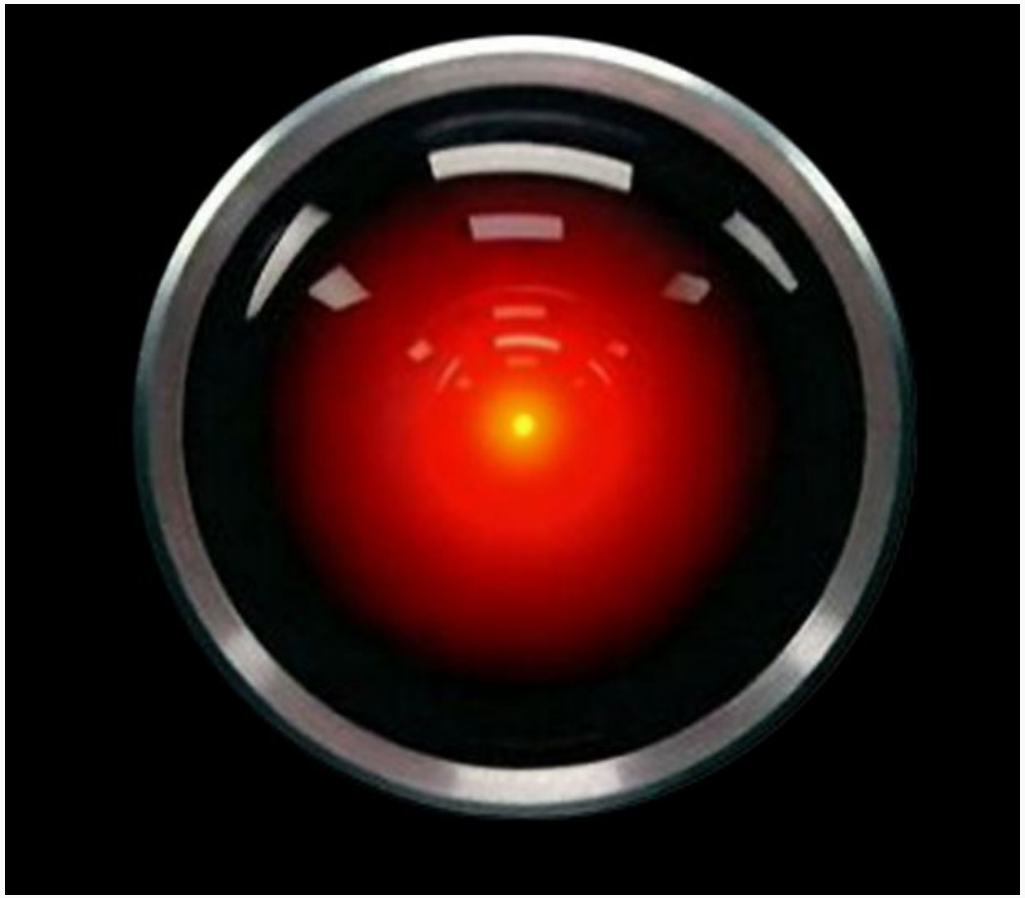
Definíció

- átfedésben van a mesterségesintelligencia-kutatással
- a természetes nyelvek számítógépes feldolgozásával foglalkozik
- a kutatások a nyelv szerkezetének gépi modellezésére irányulnak

Wikipédia:

A számítógépes nyelvészet olyan műszaki tudomány, amely a természetes nyelvű szövegek számítógépes feldolgozásával foglalkozik, de minden olyan elméleti és gyakorlati tevékenység ide tartozik, amely kapcsolatban van a természetes nyelvekkel. Egy interdiszciplína, vagyis olyan szakterület, amely több terület eredményeire és tudására épül, mint pl. az informatika, a matematika és a nyelvészet.

A Szent Grál



A nyelvtechnológia célja

olyan rendszer építése, amely fel tudja dolgozni és elő tudja állítani az emberi nyelvet – úgy, ahogy az ember teszi

elméleti motiváció: az emberi nyelvhasználatot leíró formalizált és konzisztens nyelvi modellek létrehozása

gyakorlati motiváció: a modellek gyakorlati, számítógépes megvalósítása $\rightarrow$ praktikus gépi alkalmazások

Alkalmazások

a nyelvtechnológiai fejlesztések tipikusan nagyobb alkalmazásokba beépítve jelennek meg

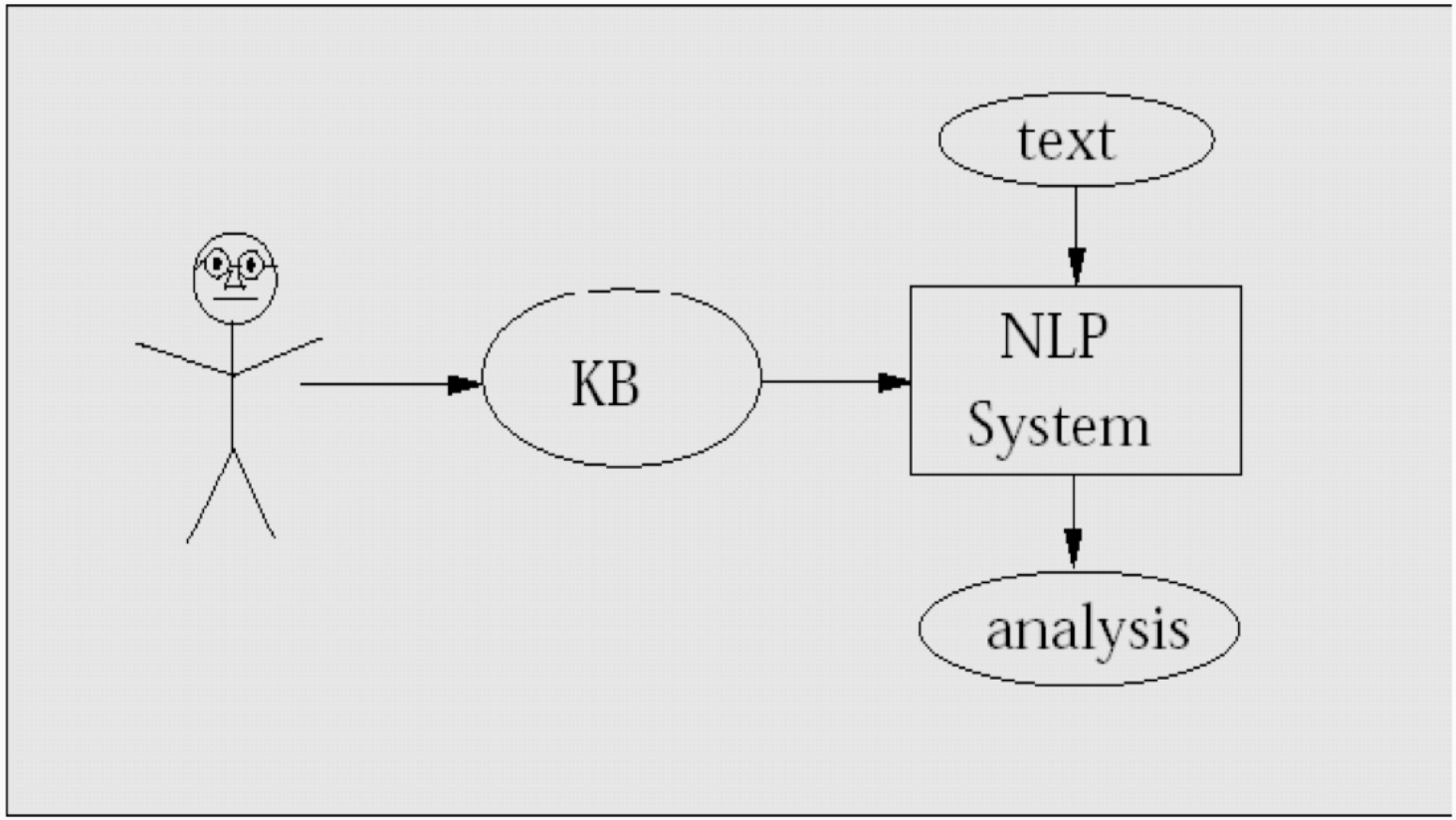
- helyesírás-ellenőrzés szövegszerkesztőben
- természetes nyelvű keresés a böngészőben
- gépi fordítás a böngészőben
- automatikus beszédgenerálás a GPS alkalmazásban
- diktálás írott szöveggé alakítása a mobilon

A nyelvtechnológia feladata

- bár a nyelvtechnológia folyamatosan fejlődik, még távolról sem tekinthető megoldottnak a nyelvfeldolgozás minden lépése ← az emberi nyelv komplexitása
- az egyik fő nehézség: az ember az értelmezés során számos nehezen formalizálható tényezőt is figyelembe vesz
  - a megnyilatkozás körülményei (hol, mikor, kikkel)
  - többletjelentés (ígéret, fenyegetés, irónia)
- a nyelvtechnológia feladata jelenleg: a szövegfolyamban detektálható releváns információ adott célnak megfelelő feldolgozása

A nyelvtechnológia módszerei

Szabályalapú metodológia



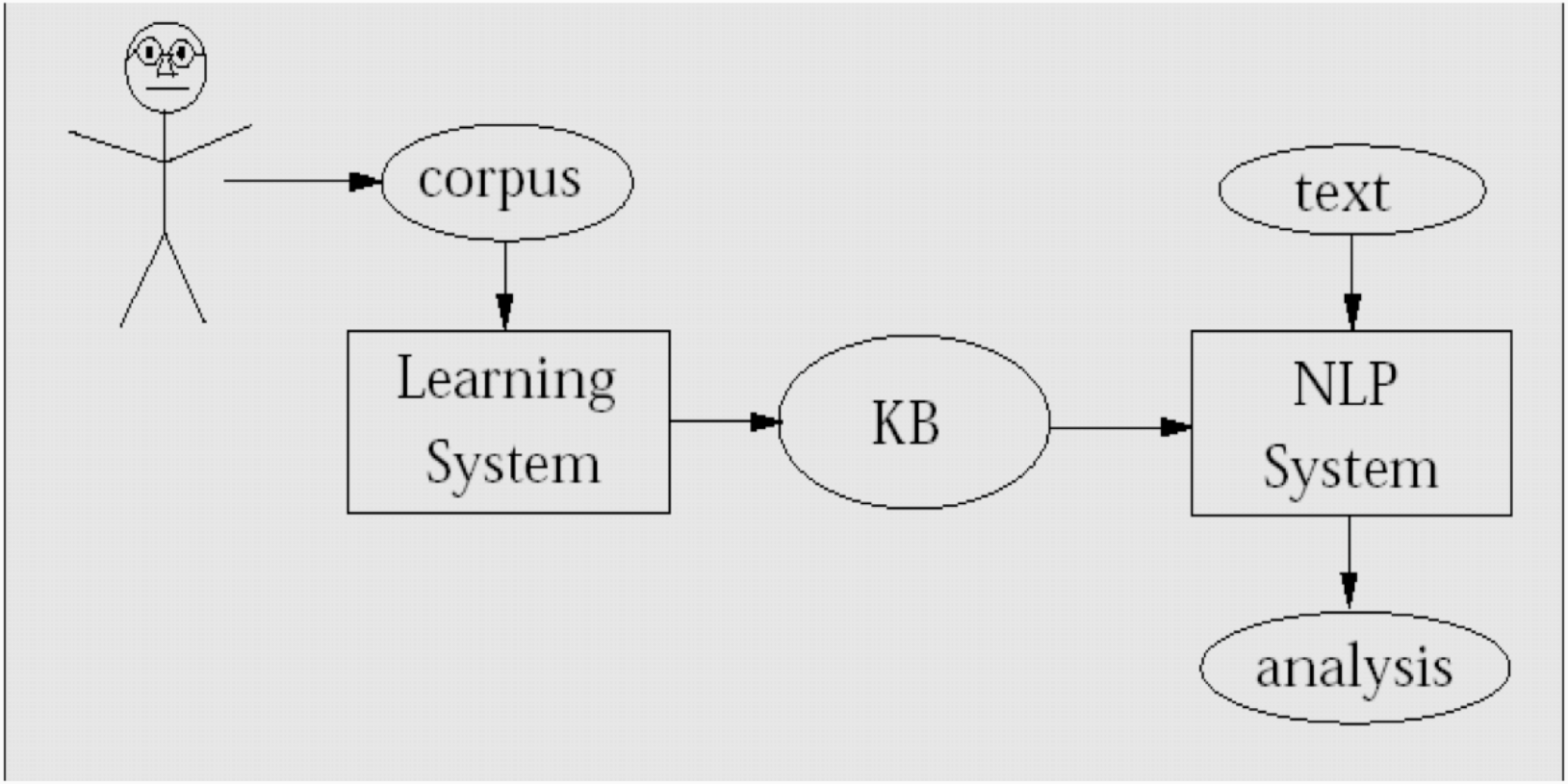
Szabályalapú metodológia

- racionalista filozófiai tradíció (Leibniz, Descartes)
- univerzális nyelvtan
- velünk született nyelvi képesség → introspekció
- grammatikalitási ítélet: 0 vagy 1
- a nyelvész írja a szabályokat → a nyelvi információt expliciten adja át a számítógépnek

Példák

e-mail cím: [a-z]+@[a-z]+\.[a-z]+
pl.: bubo@doktor.hu

Statisztikai metodológia



Statisztikai metodológia

- empirista filozófiai tradíció (Locke)
- az érzékszervi tapasztalat prioritása → tudásunk elsődleges forrása a tapasztalat
- gyakoriségi adatokból indul ki, adatorientált
- a számítógépes nyelvész az erőforrásokat adja oda a számítógépnek, ami azokat felhasználva magát tanítja → a szövegből gépi tanuló algoritmus tanulja ki a szabályszerűségeket
- a grammatikalitási ítélet nem kétértékű, hanem fokozatai vannak

A szabályalapú metodológia előnyei...

Előnyei:

- a fejlesztő nagyobb kontrollja a rendszer felett
- könnyebben értelmezhető visszacsatolás a rendszertől
- magas pontosság
- bizonyos kifejezéseknek (pl. dátumok) olyan transzparens belső szerkezetük van, hogy reguláris kifejezésekkel könnyen és hatékonyan felismerhetők

...ÉS HÁTRÁNYAI

Hátrányai:

- sok kézi munkát és nagy szakértelmet kíván
- nem hibatűrő
- bonyolult a fejlesztése, törékeny
- nehezen vihető át más nyelvre/doménre
- lehetetlen olyan szabályokat írni, amik mindent lefednek, amit kell, de semmit, amit nem kell
- alacsony fedés $\rightarrow$ a fedést a listák méretének növelésével és újabb szabályok felvételével lehet növelni

A statisztikai alapú metodológia 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
- flexibilisebb, robusztusabb

Hátrányai:

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

A statisztikai módszerek kategorizálása

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

Felügyelet nélküli tanulás

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

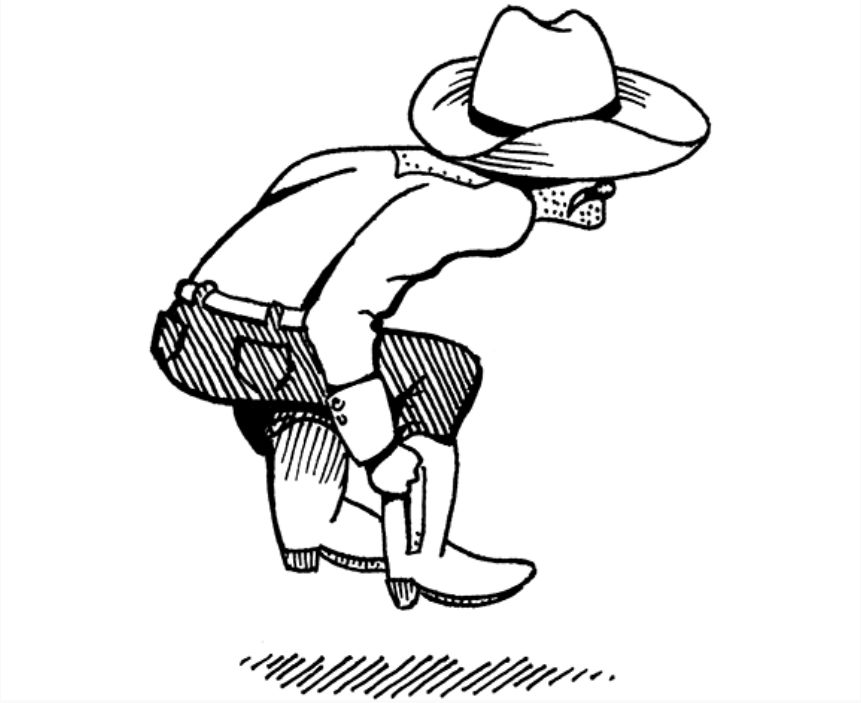
Klaszterezés:

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 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 → így tudunk következtetni a már látott nyelvi elemekből a még nem látottakra
- nyelvi annotációval ellátott korpusz → ebből tanulja ki az adott adatpontokra jellemző jegyeket, és ez szolgál majd a kiértékelés alapjául is

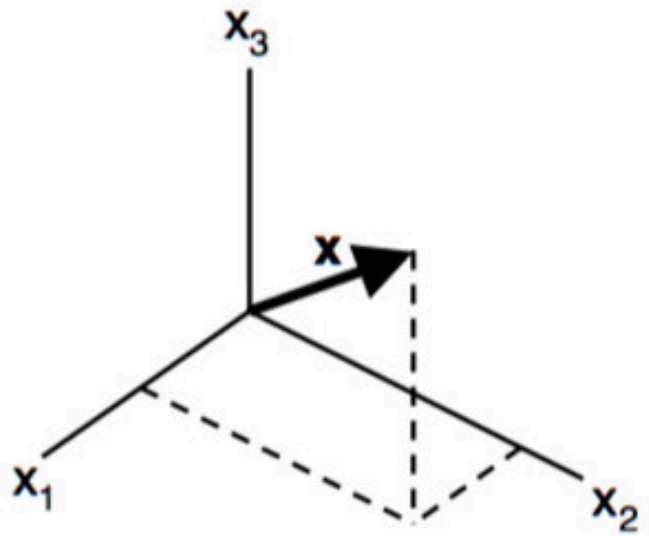
Általános séma

1. gold standard korpusz
2. tanító–fejlesztő–teszt halmaz
3. jegykinyerés
4. modellépítés
5. címkézés
6. kiértékelés

Jegykinyerés (feature extraction)

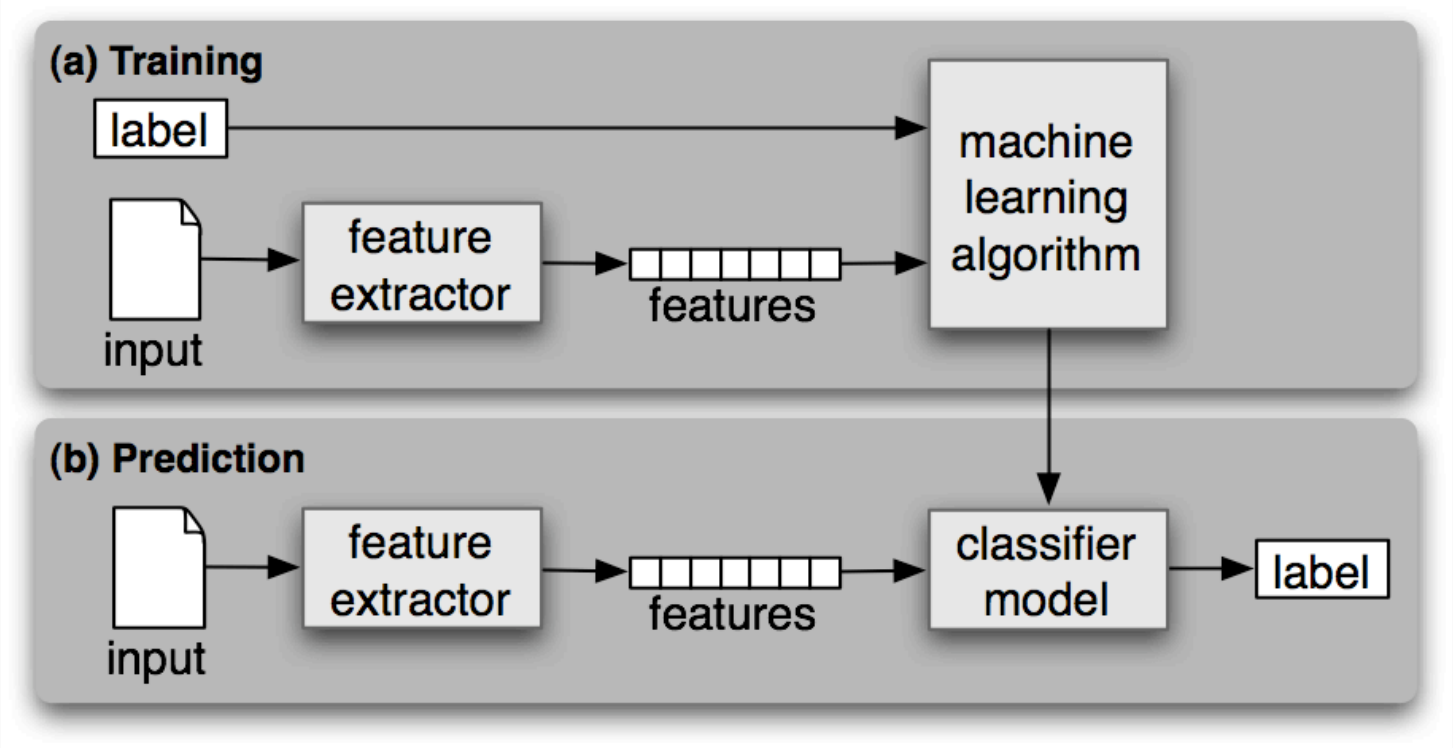
$$x = \begin{bmatrix} x_1 \\ x_2 \\ \\ x_d \end{bmatrix}$$

Feature vector



Feature space (3D)

Áttekintés


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

A neurális háló alapú metodológia

- self-supervised learning → automatikus reprezentációtanulás a kézi fícsörizálás helyett
- “deep learning”: azért mély, mert a neurális hálónak jellemzően több rétege van

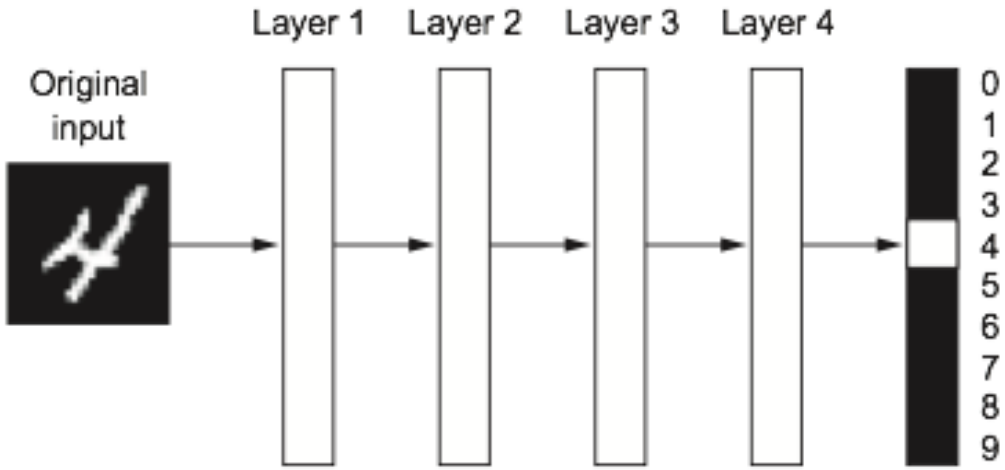


Figure 1.5 A deep neural network for digit classification

Visszacsatolás a tanulásban

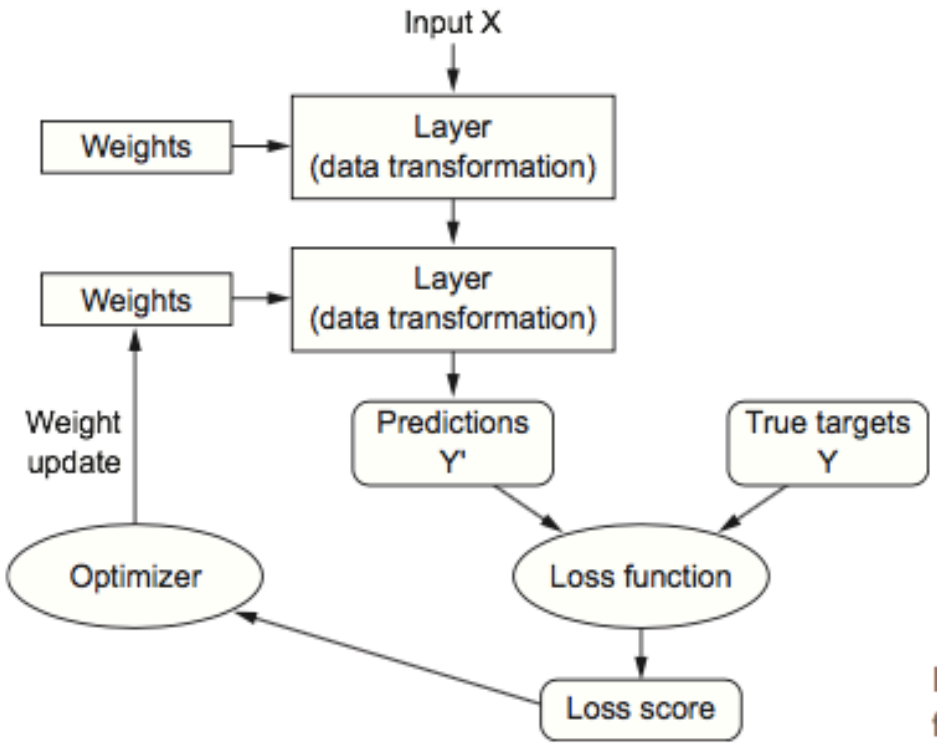


Figure 1.9 The loss score is used as a feedback signal to adjust the weights.

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.

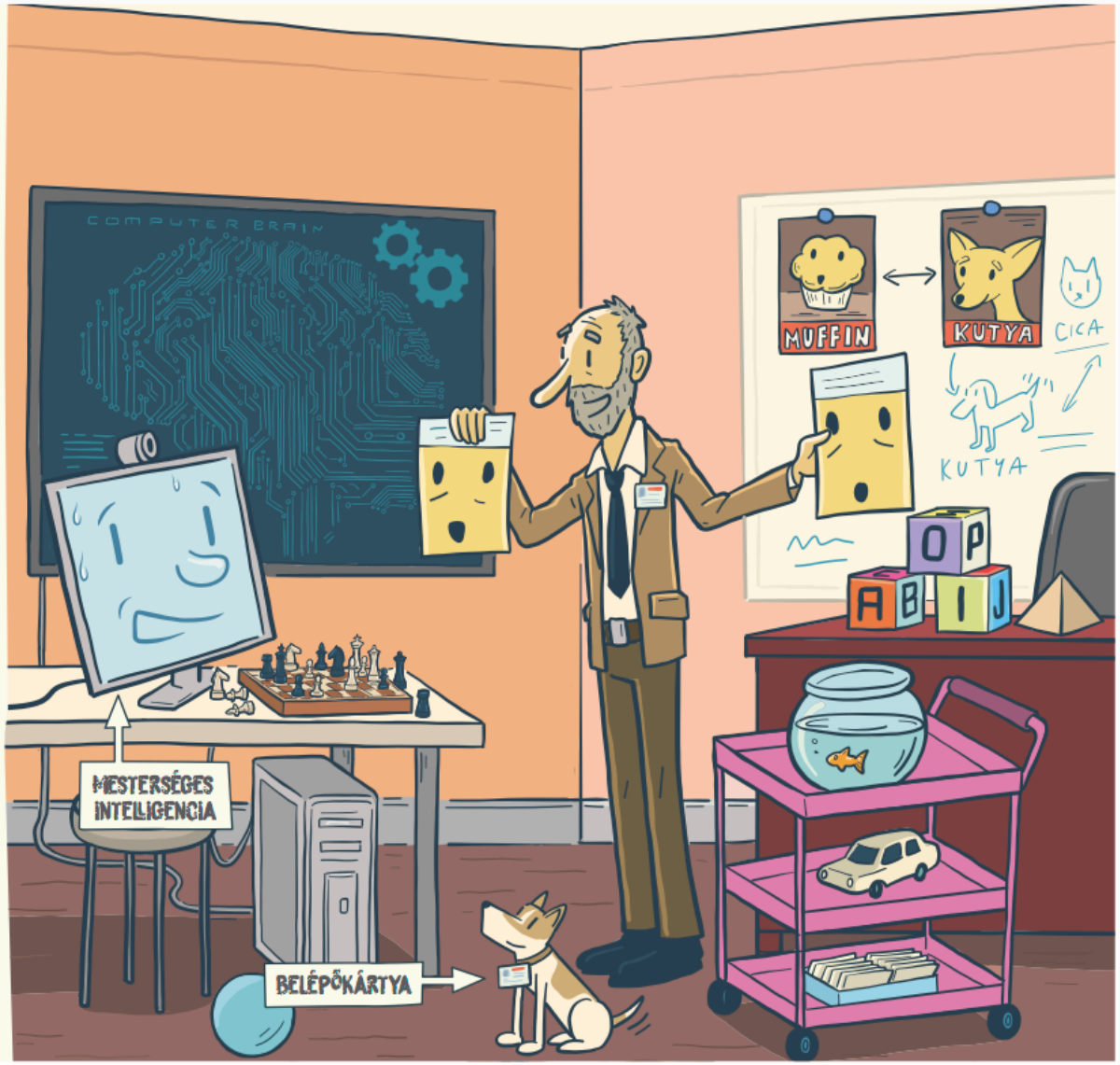
A férfiaktól a nőket
hogy különböztesse ő meg,
s mi egymástól nem áll távol:
a muffint a csivavától.

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



A korpuszok

Mi a korpusz? 1.

Kugler és Tolcsvai Nagy, 2000

“meghatározott szempontok alapján kiválasztott szövegmennyiség, amelyen a nyelvész vizsgálatát végzi”

A főbb kérdések

- mennyiség
- nyelvészeti vizsgálatokra alkalmas
- reprezentativitás, a kiválasztás szempontjai
- tárolás módja: elektronikus
- tartalom: szegmentálás, annotáció, metaadatok

Mi a korpusz? 2.

Sinclair, 2005

“a collection of pieces of language text in electronic form, selected according to external criteria to represent, as far as possible, a language or language variety as a source of data for linguistic research”

Gold standard korpusz

Követelmények:

- kihalt nyelvek esetében kimerítő, amúgy reprezentatív, de legalábbis kiegyensúlyozott
- nyelvi elemekre van bontva (token, mondat, bekezdés…)
- nyelvi annotáció van minden elemhez rendelve
- az annotáció vagy kézzel készül, vagy kézzel van ellenőrizve egy előre kidolgozott annotációs séma és útmutató alapján
- jellemzően előre meghatározott a méretük

Korpuszdizájn

Tisztázandó kérdések:

- kik és mire fogják használni a korpuszt
- a nyelvváltozat, amit le szeretnénk fedni
- a műfaj, amit reprezentálni szeretnénk
- a szükséges méret
- a korpusz jövőbeli elérhetősége, használhatósága → copyright kérdések és a szöveggyűjtés nehézségei

Az annotáció formátuma

inline (XML)

```
<s><w>Ez</w> <w>egy</w> <w>mondat</w> <c> .</c>
<s><w>Meg</w> <w>a</w> <w>második</w> <c> .</c>
```

standoff

Ez
egy
mondat
.

Meg
a
második
.

XML

EXtensible Markup Language

egyfajta jelölőnyelv (markup language) → vannak más hasonlók: YAML, JSON, MD

Előnyei:

- mind ember, mind gép számára olvasható formátum
- támogatja a Unicode-ot
- szabványos és platformfüggetlen
- képes a legtöbb általános számítástudományi adatstruktúra ábrázolására

Hátrányai:

- szintaxisa elég bőbeszédű és részben redundáns
- nagyobb tárolási költség
- nincs lehetőség a dokumentum egyes részeinek közvetlen elérésére
- átfedő adatstruktúrák modellezése nehéz/lehetetlen

Standoff annotáció

- az eredeti dokumentumok sima szöveg fájlok maradnak
- az annotációk nem szövegközi tagek, hanem egy külső fájlban jelöljük úgy, hogy megadjuk, hogy az eredeti szöveg melyik karaktertartományára vonatkozik a címkézés, és hogy milyen címkét kap a szövegrészlet
- az annotálást teljesen különválasztjuk a használt feldolgozó eszközöktől, és közben minden információt megtartunk
- az átfedő és beágyazott annotáció is könnyen kezelhető

Beágyazott és átfedő annotáció

Beágyazott annotáció

<LOC><PERSON>Kossuth Lajos</PERSON>utca</LOC>

Átfedő annotáció

a Kossuth Lajos és a Petőfi Sándor utca sarkán

BIE1

[...] közölte Wolf László, az OTP Bank vezérigazgató-helyettese az MTI érdeklődésére.

közölte	O
Wolf	B-PER
László	E-PER
,	O
az	O
OTP	B-ORG
Bank	E-ORG
vezérigazgató-helyettese	O
az	O
MTI	1-ORG
érdeklődésére	O
.	O

Annotációk összevetése

ugyanarra a szövegre vonatkozó két annotáció összevetése:

1. az egyik erősebb → egy automatikus eszköz kimenetének egy gold standard annotációhoz való hasonlítása
2. egyenrangúak → két vagy több annotátor által készített kézi annotáció összehasonlítása

cél: az akár kézzel, akár géppel készült korpuszannotáció minőségének mérése

Pozitívok és negatívok

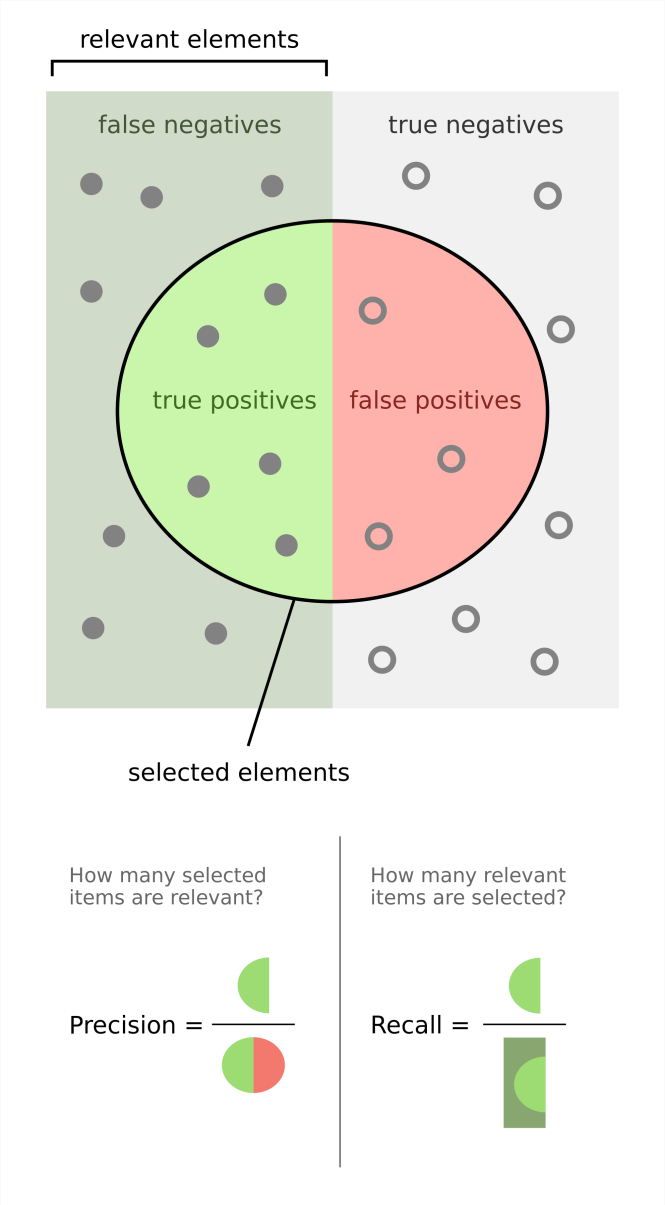
True Positive (TP): a rendszer helyesen felismerte a NE-t;

True Negative (TN): a rendszer helyesen bocsátott ki *O*-t, vagyis helyesen ismerte fel, hogy az adatpont nem NE;

False Positive (FP): a rendszer NE-nek jelölt egy adatpontot, ami nem az;

False Negative (FN): a rendszer nem ismert fel egy NE-t, pedig kellett volna.

Pontosság és fedés



Pontosság, fedés, F-mérték

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F_\beta = (1 + \beta^2) \times \frac{precision \times recall}{\beta^2 \times precision + recall}$$

F-mérték

- a *pontosság* maximalizálása: minél kevesebb tévedés $\rightarrow$ szigorítás
- a *fedés* maximalizálása: minél több találat $\rightarrow$ megengedőbb rendszer

$\beta = 1$

$$F = 2 \times \frac{precision \times recall}{precision + recall}$$

Accuracy

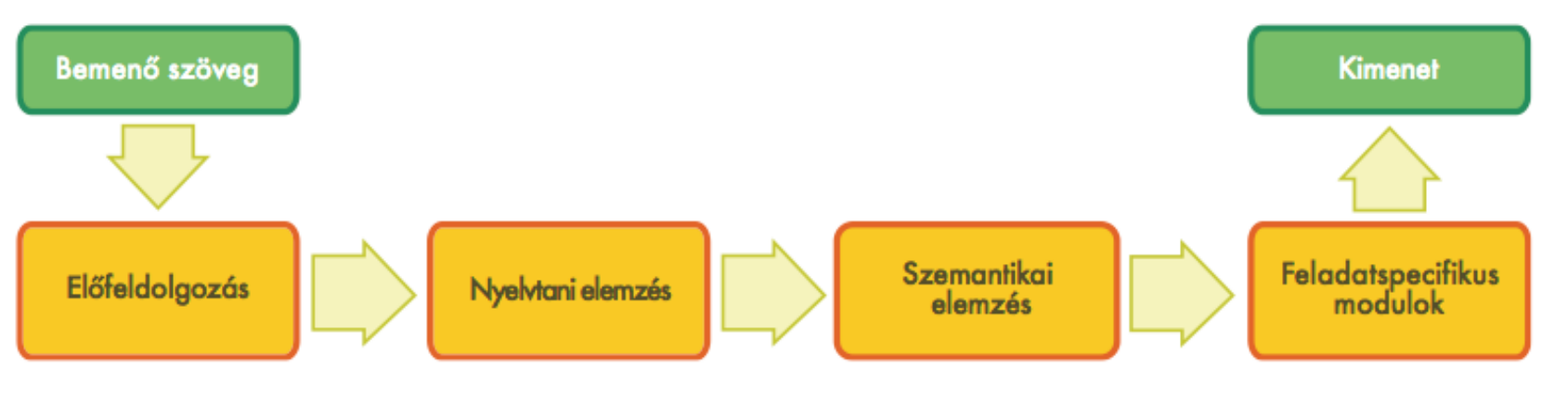
$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

A nyelvfeldolgozás lépései

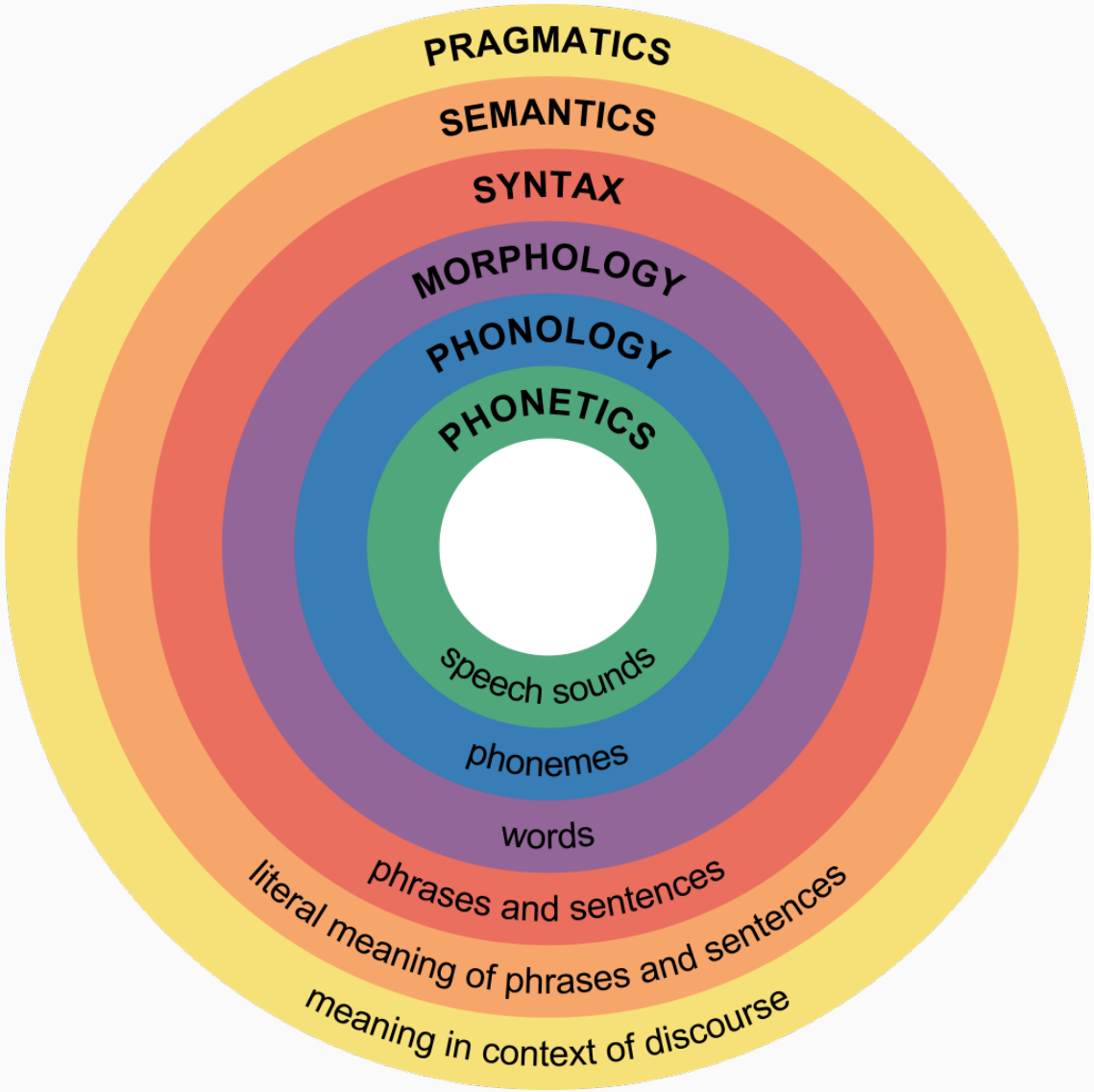
A nyelvfeldolgozás lépései

1. a digitális adatfolyam automatikus feldolgozása
2. az eredeti anyagban expliciten nem szereplő információ megtalálása
3. az adatok strukturált formába szervezése
4. az eredményeknek a felhasználó számára optimális prezentálása

Egy tipikus szövegfeldolgozó alkalmazás felépítése



A nyelvi elemzés szintjei



Részfeladatok

- mondatokra bontás
- szavakra bontás (tokenizálás)
- morfológiai elemzés
- morfológiai egyértelműsítés
- szintaktikai elemzés
- tulajdonnév-felismerés
- *koreferenciafeloldás*
- *mondatok közötti összefüggések felismerése*
- *szemantikai relációk detektálása*
- *érzelmek detekciója*

Mondatszegmentálás

- Minden mondat.
- Mondathatárok azonosítása.
- Pontos problémák:
  - Rövidítések (*du. 5-kor*).
  - Római számok (*V. László*).
  - Sorszámok (*10. éve, hogy …*).
- Egyéb nehézségek:
  - Idézeten belüli mondatok.
  - Zárójelen belüli mondatok.

Tokenizálás

- Detokenizálhatóság és elválasztás (és az *-e* partikula).
- Szóalkotó karakterek, szónemalkotó karakterek, és amik köztük vannak:
  - Zárójelek, idézőjelek, aposztrófok kezelése.
  - Rövidítések végén lévő pont vs. mondatvégi pont.
- Számok (space-szel tagolt számok, mértékegységek, képletek, dátumok).
- Informatikai kifejezések (URL, elérési út, emailcím).
- Smiley-k és emoji-k.

Morfológiai elemzés

tokenszintű elemzés → nem lát se előre, se hátra → no kontextus → többértelműség

kerekesszék

kerek/ADJ+esszé/NOUN<PLUR>
kerekes/ADJ+szék/NOUN
kerék/NOUN[ATTRIB]/ADJ+szék/NOUN
kerek/ADJ[ATTRIB]/ADJ+szék/NOUN

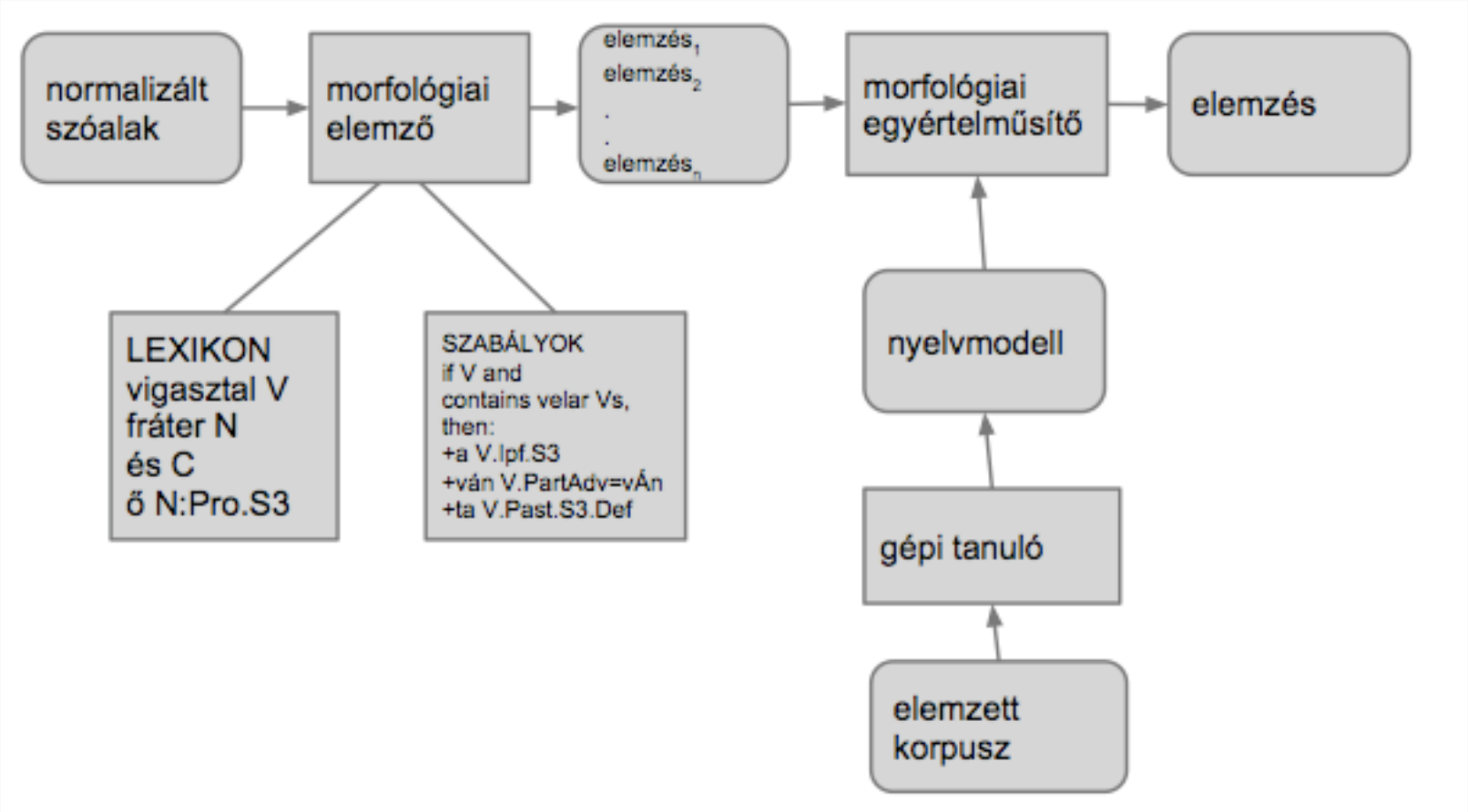
falucska

```
fa [/N] + luc[/N] + ska[/N] + [Nom]
fa[/N] + lucsok[/N]=lucsk + a[Poss.3Sg] + [Nom]
falu[/N] + cska[_Dim:cskA/N] + [Nom]
falucsok[/N]=falucsk + a[Poss.3Sg] + [Nom]
falucska[/N] + [Nom]
```

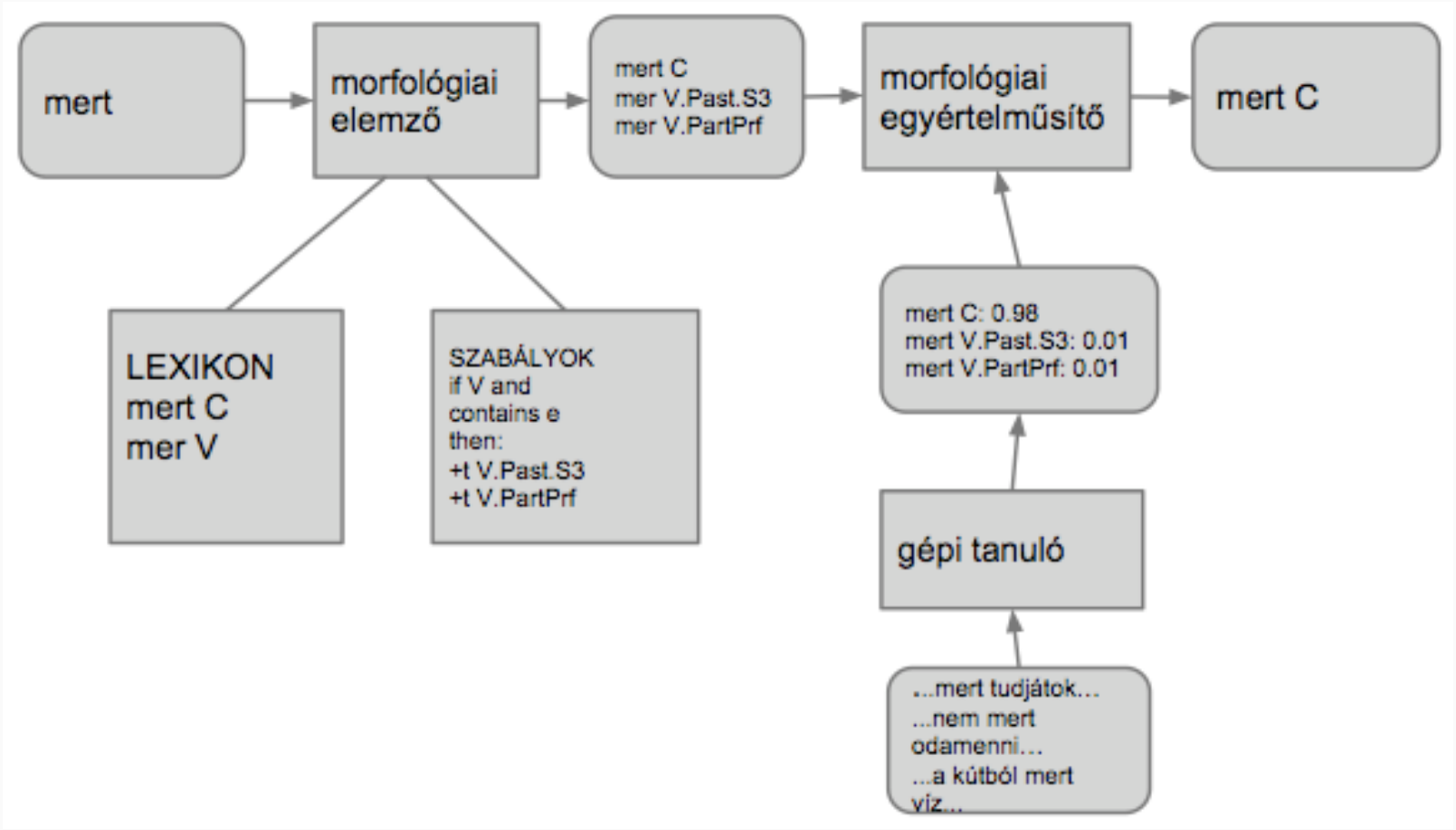
Mit tartalmazhat a kimenet?

- morfoszintaktikai információk
- jelentésre vonatkozó információk
- hangalakra vonatkozó információk (allomorfia)
- szófajkód
- lemma
- morfológiai szegmentumok

Morfológiai egyértelműsítés 1.



Morfológiai egyértelműsítés 2.



Named Entity Recognition (NER)

2 lépésből áll:

1. a nevek lokalizálása strukturálatlan szövegben
2. a megtalált elemek besorolása előre definiált névosztályokba
   - *Person*, *Location*, *Organization*, *Date*, *Time*, *Money*, *Percent*, *Measure* (MUC)
   - *Person*, *Location*, *Organization*, *Miscellaneous* (CoNLL)

Sekély szintaktikai elemzés

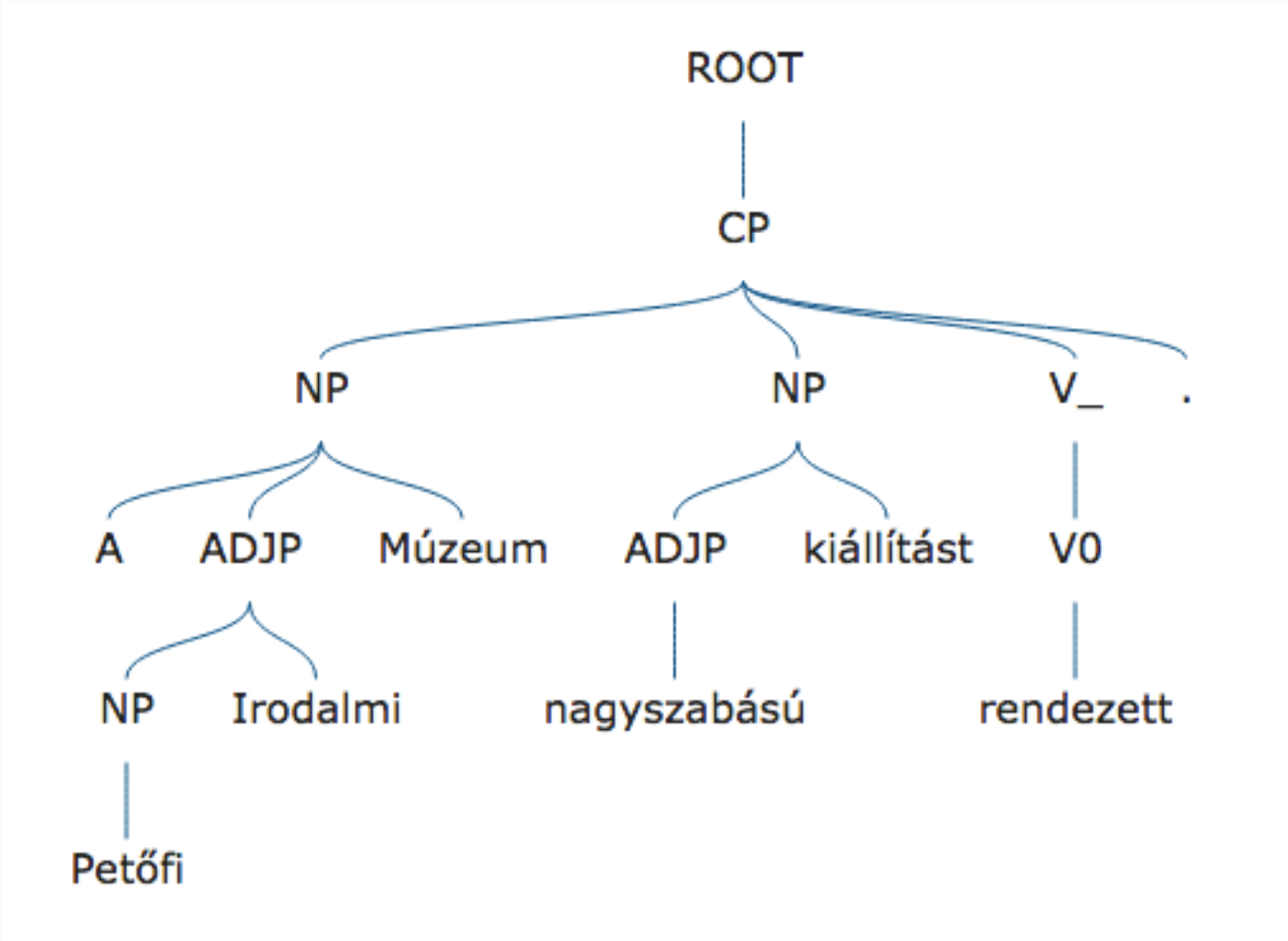
chunking

[Immár]$_{AdvP}$ [negyedik éve]$_{NP}$ [a Manchester United]$_{NP}$
[a világ leggazdagabb csapata]$_{NP}$ [bevétel szerint]$_{PP}$.

1. minden frázis megtalálása egy mondatban
2. maximális NP-k megtalálása
3. alap NP-k megtalálása

Összetevős elemzés

A mondatok összetevős szerkezeti elemzése azt tárja fel, hogy a szavak egymással kombinálódva milyen kifejezéseket alkotnak, illetve hogyan állnak össze egy mondattá.



Függőségi elemzés

A függőségi elemzés a mondatok szerkezeti egységei közötti függőségi viszonyokat (pl. alany, tárgy, jelző) tárja fel.

State of the art

mondatra bontás (ACC)	$\sim 97\%$
tokenizálás (WAC)	$\sim 99\%$
morfológiai egyértelműsítés (ACC)	$\sim 98\%$
tulajdonnév-felismerés (F1)	$\sim 97\%$
főnévi csoportok felismerése (F1)	$\sim 95\%$
függőségi elemzés (UAS)	$\sim 93\%$
függőségi elemzés (LAS)	$\sim 91\%$
vonzatkeretek kinyerése (F1)	$\sim 65\%$
metaforikus kifejezések detektálása (F1)	$\sim 43\%$

Szövegfeldolgozó eszközláncok magyarra

e-magyar

- fejlesztő: az MTA NYTI koordinálásával a magyar nyelvtechnológiai közösség
- az elérhető SOTA eszközöket integrálta egy egységes eszközláncba
- fontos célok: modularitás, nyílt forráskód, kutatás- és alkalmazásközpontú felhasználás
- egységes formátum egy egységes keretrendszerben: `xtsv`

https://github.com/dlt-rilmta/emtsv
http://e-magyar.hu/hu/

Az xtsv formátum

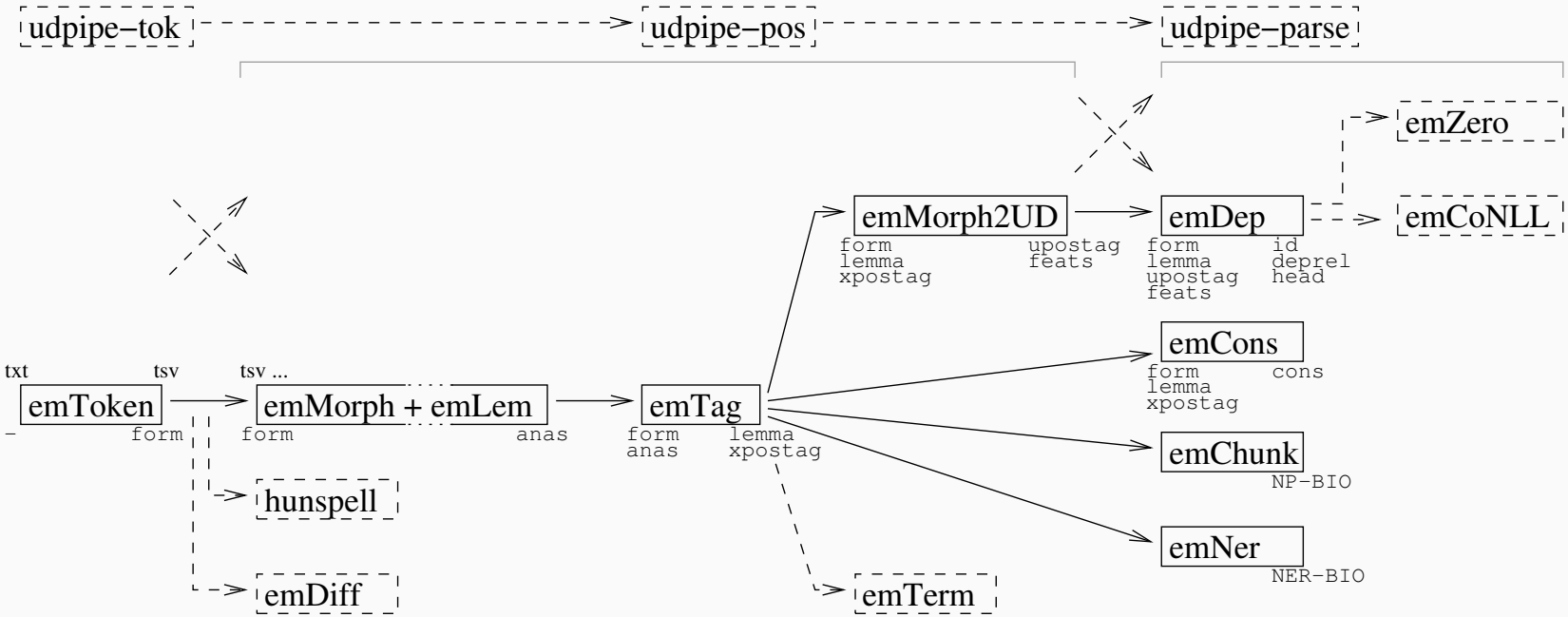
- fejléccel rendelkező *tsv* (*tab separated values*)
- egy sor egy token
- az oszlopokban az annotációk
- a mondathatárok üres sorok
- # karakter után megjegyzések

példa

```
form        lemma    xpostag
# Ez egy    mondat   eleji komment
A           a        [/Det|Art.Def]
kutyák      kutya    [/N][Pl][Nom]
ugatnak     ugat     [/V][Prs.NDef.3Pl]
.           .        [Punct]

A           a        [/Det|Art.Def]
...
```

Az e-magyar feldolgozó lánca



Magyarlánc 3.0

- fejlesztő: Szegedi Tudományegyetem
- elemzési szintek: tokenizálás, mondatrabontás, morfológiai elemzés és egyértelműsítés, függőségi elemzés, összetevős elemzés
- nem moduláris, csak egyben lehet futtatni az elejétől a végéig
- új modulokat nem lehet integrálni
- nyílt forráskódú, szabadon felhasználható

https://rgai.sed.hu/node/100

UDPipe

- fejlesztő: Institute of Formal and Applied Linguistics, Charles University
- neurális nyelvmodelleken alapul
- elemzési szintek: tokenizálás, mondatrabontás, morfológiai egyértelműsítés, függőségi elemzés
- szabadon felhasználható, nyílt forráskódú
- az egyes lépéseknél ki-be lehet szállni az egységes formátumnak köszönhetően, de új modulokat nem lehet integrálni

http://ufal.mff.cuni.cz/udpipe

huspaCy

- fejlesztő: spaCy és Orosz György
- elemzési szintek: tokenizálás, mondatrabontás, morfológiai egyértelműsítés, függőségi elemzés, tulajdonnév-felismerés, szóvektorok
- nyílt forráskódú, szabadon használható
- bármelyik lépésnél ki-be lehet szállni
- új modulokat könnyen lehet integrálni

https://github.com/oroszgy/spacy-hungarian-models/

STANZA

- fejlesztő: Stanford Egyetem
- elemzési szintek: tokenizálás, mondatrabontás, morfológiai elemzés és egyértelműsítés, függőségi elemzés
- nyílt forráskódú, szabadon használható
- kevésbé jó teljesítmény

https://stanfordnlp.github.io/stanza/

Az összehasonlítás

paraméterei:

- teljesítmény
- gyorsaság
- nyelvfüggetlenség
- ki- és beszállási lehetőség az egyes lépéseknél
- új modulok integrálhatósága
- könnyű használhatóság

résztvevői:

- *e-magyar*
- UDPipe
- huspaCy
- Magyarlánc 3.0

Teljesítmény

feladat	UDPipe	huspaCy	SOTA
morf. egyértelműsítés (ACC)	86,40	94,91	96,33
lemmatizálás (ACC)	88,50	95,49	96,33
dependenciaelemzés (UAS)	72,70	76,18	93,22
dependenciaelemzés (LAS)	67,10	66,58	91,42
NER (F1)	-	93,95	96,10

UDPipe: http://ufal.mff.cuni.cz/udpipe/models
huspaCy: https://tinyurl.com/y4ole3ul

Gyorsaság

elemző	POS	dependencia
e-magyar (CLI)	2.320	300
e-magyar (REST)	2.600	310
Magyarlánc	5.550	450
UDPipe	9.280	3.300
huspaCy	33.980	15.000

- a méréseket ugyanazon a 100.000 tokenes fájlon végeztük el, ötször → átlag
- RAM disk
- az inicializálási időt nem vettük figyelembe
- erősebb asztali gép

Összehasonlítás

	telj.	gyors.	függ.	ki-be	integr.	haszn.
emtsv	O	X	X	O	O	O
Magyarlánc	O	X	X	X	X	O
UDPipe	X	O	O	O	X	O
huspaCy	X	O	X	O	O	O

simon.eszterke@gmail.com