

Deep learning alapú szótagolás

Németh Gergely Dániel

Abstract

Kivonat

1. Bevezetés

2. Szótagoló programok

Az elterjedt szótagoló algoritmusok kétféle csoportba bonthatóak: szabály- vagy szó-táralapúak. A szabad szoftverek világában *de facto* a T_EXszótagolási algoritmusát használják.

A T_EX eredeti szótagoló algoritmusát Prof. Knuth tervezte 1977 nyarán[2]. Ez három fő szótagolási szabályt alkalmazott: 1) utótag leválasztás, 2) előtag leválasztás és 3) magánhangzó - mássalhangzó - mássalhangzó - magánhangzó (vccv) elválasztás, azaz ha ilyen betűnégyes található a szóban, legtöbb esetben a mássalhangzók mentén elválasztható. Ez a három szabály gyakran alkalmazható, azonban már az első algoritmusban kiegészült kisebb szabályokkal („break vowel-q” vagy „break after ck”) és kivételek listájával(300 szó).

A T_EX82 verzióhoz megjelent Liang szótagoló algoritmus[3], aminek legfontosabb újítása a minta(*patterns*) alapú szótagolás bevezetése volt. Ennek lényege, hogy a szótagolási szabályok mintákra definiálódtak és az algoritmus a szótagolás során ezeket a mintákat keresi a szóban.

A T_EXjelenlegi verziójában a Hunspell szótagoló algoritmust alkalmazzák[4]. Ez az eljárás Liang algoritmusán alapszik, amit nyelvfüggetlen speciális szótagolási kiterjesztésekkel (*non-standard hyphenation extension*) egészített ki.

2.1. Liang algoritmus

Liang szótagoló algoritmusának alapját a szótagolási minták alkotják[3]. Az algoritmust az angol *hyphenation* elválasztását az alábbi módon végzi:

Az algoritmus először megnézi, hogy a szó benne van-e a kivételek listájában (ez lényegében teljes szavakat tartalmaz mintaként). A *hyphenation* szó nincs a kivételek listájában.

Ezután a szó elejére és végére illeszt egy pont jelző karaktert. Ennek jelentősége azoknál a mintáknál lesz, amelyek akkor érvényesülnek, ha a szó elején, vagy a végén szerepelnek.

.hyphenation.

Ezt követően a minták között keres illeszkedést. A *hyphenation* szóra ezek az alábbiak: `hy3ph`, `he2n`, `hena4`, `hen5at`, `lna`, `n2at`, `ltio`, `2io` [3, 37. oldal] A megfelelő mintákat ráillesztve a szóra, a bennük szereplő számokat a szó karakterei közé szűrve az 1. ábrán szereplőket kapjuk.¹

```

. h y p h e n a t i o n .
  h y3p h
    h e2n
    h e n a4
    h e n5a t
      l n a
      n2a t
        l t i o
        2 i o
-----
.0h0y3p0h0e2n5a4t2i0o0n0.
  h y-p h e n-a t i o n

```

1. ábra. A *hyphenation* szótagolása

Az algoritmus következő lépésében minden két szomszédos karakter közé illesztünk egy számot. Alapértelmezetten 0-t és ha ennél nagyobb számot találtunk a minta-illesztésnél, a legnagyobb kapott értéket adjuk meg.

A szótagolás szabálya innen már csak egy lépés: a páratlan számok mentén elválasztunk, a párosoknál nem. Ezzel megkaptuk a `hy-phen-ation` elválasztást.

2.1.1. Minták választása

A fenti algoritmus hatékonyságát nyilvánvalóan a minták mennyisége és hatékonysága határozza meg. Csak azok a szavak választódnak el, amelyekre illeszkedik minta és csak azok lesznek jó elválasztások, melyekre jó minta illeszkedik.

2.2. Hunspell

2.2.1. Szótagolási hibák a magyar Hunspellben

Hunspell szótagolása:
au-tó-val

Helyesen:
a-u-tó-val

3. Deep learning alapú módszerek

3.1. Deep learning bevezetés

A gépi tanulás területén egyre nagyobb teret nyernek a deep learning alapú módszerek. A hagyományos osztályozási feladatokon túl, szekvenciális, ún. taggelési problémák megoldására is alkalmasak a neurális hálók, különös tekintettel a rekurrens neurális hálókra.

A mély tanulás bevetését a nyelvtechnológia területén az a tény élteti, hogy az elektronikus kommunikáció korában jelentős mennyiségű nyelvi adatbázisok állnak

¹Az ábrázolásmód Németh cikkéből származik[4].

rendelkezésünkre, így a deep learning legfontosabb eleme, a megfelelő méretű tanulóadat elérhető. Az ismertetett elválasztási algoritmusok kulcseleme, hogy megfelelő elválasztási mintákat ismerjünk fel az adathalmazon (*korpusz*). A képfelismerés terén sokkal komplexebb minták megtalálásában is jelentős eredményeket értek el a deep learning segítségével, így e módszer bevetése a szótagolás problémakörében megalapozott.

3.1.1. Keretrendszerek

Deep learning rendszerek tervezésénél az utóbbi időben ipari és kutatói területen egyaránt elterjedőben vannak a magas szintű keretrendszerek. Ezek legnagyobb előnye, hogy az aktív fejlesztői közösség a gyorsan fejlődő tudományág eredményeit alkalmazható technológiákként tárja a felhasználók elé. A jelentősebb keretrendszerek: TensorFlow², Torch7³, Keras⁴. Az itt ismertetett eljárások megvalósításához a Keras nyújtotta lehetőségekkel valósultak meg[1].

3.2. Előrecsatolt neurális hálók

3.2.1. Karakter címkézés

BMES és BM

3.2.2. Adatelőkészítés

One-hot, flatten, webcorpus

3.2.3. Előrecsatolt háló tanítása

3.3. Rekurrens neurális hálók

4. Eredmények

4.1. Előrecsatolt háló

| Tag | Length | Loss | Optim | Batch size | Error valid | Epochs |
|------|--------|-----------|-------|------------|-------------|--------|
| BMES | 3 | cat_cross | adam | 512 | 0.616 | 483 |

1. táblázat. Előrecsatolt háló tanítása különböző paraméterekkel

Hivatkozások

[1] François Chollet. Keras, 2015.

[2] Donald Ervin Knuth. *TEX and METAFONT: New directions in typesetting*. American Mathematical Society, 1979.

²TensorFlow: <https://www.tensorflow.org/>

³Torch7: <http://torch.ch/>

⁴Keras: <https://keras.io/>

- [3] Franklin Mark Liang. *Word hyphenation by computer*. Department of Computer Science, Stanford University, 1983.
- [4] László Németh. Automatic non-standard hyphenation in openoffice. org. *COMMUNICATIONS OF THE TEX USERS GROUP TUGBOAT EDITOR BARBARA BEETON PROCEEDINGS EDITOR KARL BERRY*, page 32, 2006.