Elméleti összefoglaló

Nyelvi modellezés a beszédfelismerésben című méréshez

Tarján Balázs
BME-TMIT
2016

Tartalom

1 Bevezetés

A korszerű beszéd-szöveg átalakító rendszerek egyik alapvető építőeleme a statisztikai nyelvi modell, melynek az a feladata, hogy a szavak kapcsolódási valószínűségét modellezze. Nyelvi modell alkalmazása nélkül, csupán az akusztikai információra támaszkodva nem működhetnének hatékonyan ezek a rendszerek. A mérés során a hallgatók betekintést nyerhetnek az ún. n-gram modellek tanításának, paraméterezésének és tesztelésének folyamatába. A mérés elsősorban a nyelvi modellek gépi beszédfelismerésben történő alkalmazását, optimalizálását mutatja be, de a megszerzett tudás más tématerületen is hasznosítható.

A statisztikai nyelvi modellek tanításakor az a célunk, hogy egy adott szósorozat ($W= w_1,..., w_K$) valószínűségét minél pontosabban meg tudjuk becsülni:

$$P(W) = P(w_1, w_2, w_3..., w_K)$$

A probléma felvetése elsőre öncélúnak tűnhet, de pár példán keresztül belátható, hogy a szóláncok valószínűségének ismerete sok előnnyel jár.

1.1 Gépi fordítás

Vegyünk először egy **gépi fordítási** problémát. Legyen a magyarra fordítandó szókapcsolat az, hogy „captain of the Red Army”. A szókapcsolatban szereplő „red” szót pirosra, de akár vörösre is lehet fordítani magyarul. Ebből adódik két lehetséges alternatíva is:

P(**vörös** hadsereg százados) > P(**piros** hadsereg százados)

Egy megfelelően tanított nyelvi modell egyértelműen a „vörös hadsereg” kifejezésre ad vissza nagyobb valószínűséget, mivel a „piros hadsereg” szókapcsolat nagyon ritka. Így a probléma szakértői segítség nélkül, gépi úton is kezelhető.

1.2 Helyesírás ellenőrzés

Hasonló problémakör a helyesírás-ellenőrzésé. Vannak szóalakok, melyek önmagukban helyesek ugyan, de a kontextus alapján sejthető, nem pont azt a szó szerettük volna leírni. Például:

P(**egyelőre** nem látom) > P(**egyenlőre** nem látom)

Mindössze egyetlen betű a különbség az egyelőre és az egyenlőre között, mégis teljesen más szavak környezetében fordulnak elő gyakran. Tehát például a hasonló alakú szavak egyértelműsítésében is nagy segítségünkre lehet a nyelvi modell.

1.3 Gépi beszédfelismerés

Mérésünk fókuszában a nyelvi modellek **gépi beszédfelismerés**ben történő felhasználása áll. A beszéd-szöveg átalakító rendszerekben az alsóbb felismerési rétegek kimenetének egyértelműsítésre használjuk a nyelvi modelleket. Egy nyelvi modellt nem alkalmazó beszédfelismerő rendszer képes lenne a szótárában fellelhető szavakat tetszőleges sorrendben visszaadni. Belátható azonban, hogy ha nem vesszük figyelembe a szavak kapcsolódási valószínűségét, akkor nagyon nehéz különbséget tenni a kiejtésben megegyező szósorozatok között:

P(mellény gomblyuk) > P(mell lény gombjuk)

A fenti két szósorozat kiejtése nagyjából megegyezik, így a beszédfelismerő rendszer számára akár azonos valószínűségűek is lehetnének, ha nem lenne nyelvi modell, mely egyértelműen az első változat javára dönt.

A beszéd-szöveg átalakító rendszerek nyelvi modelljét feladatspecifikus szövegek alapján szokás tanítani. Tanításhoz használt szövegek együttesét **szövegkorpusz**nak hívjuk. A következő fejezetben bemutatunk pár statisztikai módszert, mellyel jellemezhetjük a tanítókorpuszt.

2 Szövegkorpuszok statisztikai vizsgálata

A szövegkorpuszokhoz bármilyen úton is jutunk hozzá eredeti formájukban általában nem alkalmasak a nyelvi modellek tanításához. Ehhez először **normalizáláson** (adattisztítás) kell, hogy átessenek, melynek pontos lépései mindig a nyelvi modell felhasználási területétől függnek. Beszédfelismerési célokra tanított nyelvi modelleknél ez általában a kiejtéssel nem rendelkező karakterek (pl. írásjelek) eltávolítását és a szavak írásmódjának egységesítését jelenti. Például:

Eredeti alak:

```
To be, or not to be, that is the question:
Whether 'tis Nobler in the mind to suffer
```

Normalizált alak:

```
to be or not to be that is the question
whether 'tis nobler in the mind to suffer
```

2.1 Token, Type

Ezt a két fogalmat nagyon elterjedten használják a természetes nyelvfeldolgozásban. A **token** alapesetben bármilyen karakterláncot jelenthet függetlenül attól, hogy a karakterlánc éppen mit is jelöl. Természetes nyelveknél az esetek többségében megfeleltethető a szavaknak. Ha egy szövegkorpuszban a tokenek számáról beszélünk, akkor arra vagyunk kíváncsiak, hogy összesen hány space határolt karaktersor (szó, token) található benne.

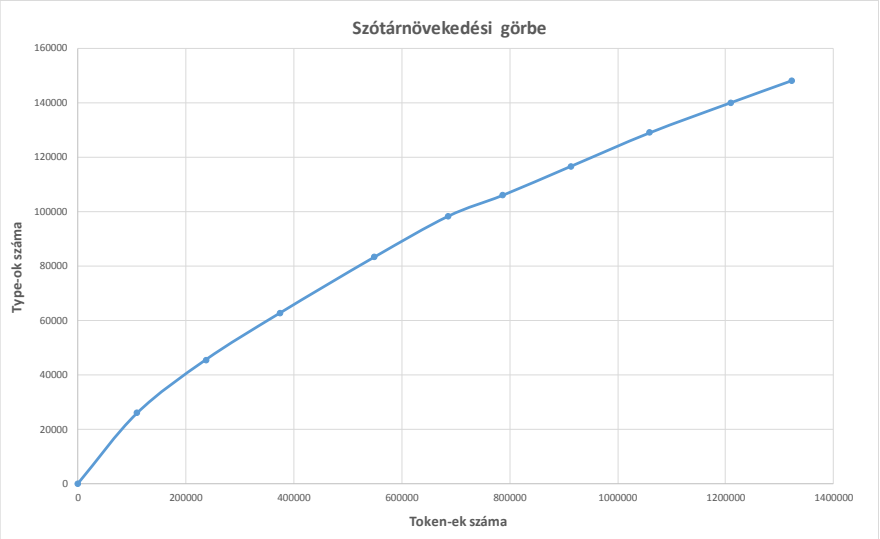
Nem csak azt hasznos tudni azonban, hogy hány token van egy korpuszban, hanem azt is, hogy összesen hány fajta ilyen token van. Vagy másképpen fogalmazva hány **type** van? A beszédfelismerésben a type-ok száma azért is fontos, mert meghatározza, hogy mely szavakat/szóalakokat képes felismerni a rendszer. Ebből következően a „szótárméret” (vocabulary size) kifejezés is használatos. Például:

to be or not to be that is the question
whether 'tis nobler in the mind to suffer

Token-ek száma: 18
Type-ok száma: 14

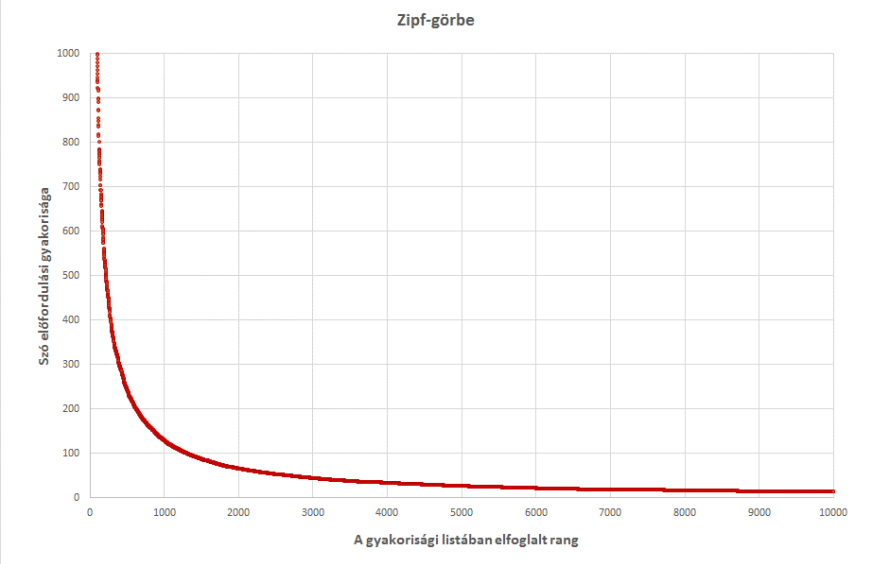
2.2 Szótárnövekedési görbe

Az előző pontban bemutatott két érték összefüggését hivatott bemutatni a **szótárnövekedési görbe**. Itt egyszerűen arról van szó, hogy a type-ok számának változását a token szám függvényében ábrázoljuk. Egy lehetséges megoldás a görbe felvételére, hogy a korpuszt kisebb darabokra bontjuk, melyeket folyamatosan egyesítjük, miközben minden lépésben megmérjük a token-en és type-ok számát. Így egy ehhez hasonló ábrát kapunk:



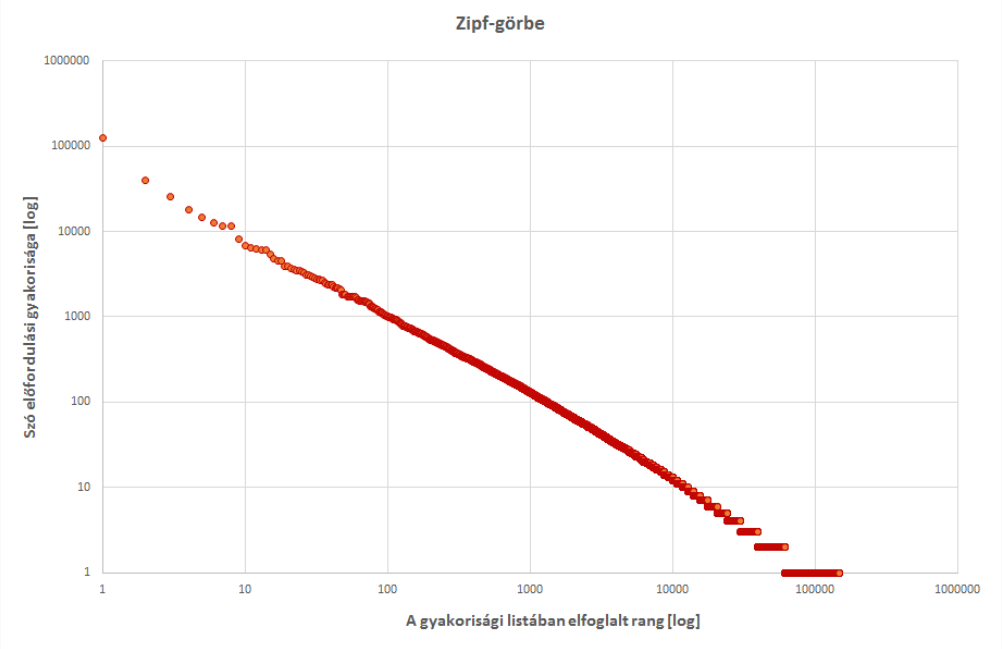
2.3 Zipf-görbe

A Zipf-görbe a token-ek elfordulási gyakoriságát ábrázolja a gyakorisági listában elfoglalt rangjuk függvényében. Ez gyakorlatban azt jelenti, hogy egy szövegben vesszük az összes type-ot és meghatározzuk, hogy melyik hányszor fordul elő a szövegben. Ezután csökkenő sorrendbe rendezzük a type-okat előfordulási gyakoriság szerint és így ábrázoljuk őket. Általában ehhez hasonlatos görbét kapunk természetes nyelveknél:



A Zipf-eloszlás jól közelíthető 1/x függvénnyel, ahol „x” a gyakorisági listában elfoglalt rang. Ez a gyakorlatban azt jelenti, hogy a második leggyakoribb type nagyjából fele olyan gyakori, mint a

leggyakoribb, a harmadik leggyakoribb kb. harmad olyan gyakori, és így tovább. Mint látható ez egy hosszú farokkal rendelkező eloszlás, tehát igaz rá, hogy a gyakorisági lista első felében található type-okkal lefedhető a korpusz nagyon nagy része, azonban a teljes korpusz lefedéséhez nagyon sok, nagyon ritka szót is modelleznünk kell. A Zipf-görbét gyakran ábrázolják log-log skálán is:



3 Nyelvi modellek tanítása

Ebben a fejezetben azt elméleti alapjait mutatjuk be annak a folyamatnak, mely segítségével egy szövegkorpuszokból nyelvi modelleket tanítunk.

3.1 Az n-gram közelítés

A nyelvi modell egy adott szósorozat ($W = w_1,..., w_K$) valószínűségét hivatott becsülni. Ezt a láncszabály alkalmazásával a következőképpen fejezhetjük ki:

$$P(W) = P(w_1,..., w_K) =$$
$$P(w_1) \cdot P(w_2 \mid w_1) \cdot P(w_3 \mid w_1, w_2) \cdot ... \cdot P(w_K \mid w_1,..., w_{K-1})$$

Ezzel a kifejezéssel az a probléma, hogy $P(W)$ pontos értékének meghatározásához ismernünk kellene minden szónak azt a valószínűségét abban a speciális esetben, amikor éppen az összes a mondatban előtte szereplő szó után következik. Ez nyilvánvalóan szinte lehetetlen feladat, mert nagyon sok adatot kellene nyilvántartanunk, illetve mert akárcsak egyetlen minta begyűjtéséhez is óriási tanítóadatbázisra lenne szükség.

Az elméleti érték becslésére az ún. n-gram modelleket szokás használni, melyek a szavak n-1 hosszúságú előzményét veszik csupán figyelembe.

$$P(W) = P(w_1,..., w_K) \approx$$
$$P(w_1) \cdot P(w_2 \mid w_1) \cdot P(w_3 \mid w_1, w_2) \cdot ... \cdot P(w_K \mid w_{K-(n-1)},..., w_{K-1})$$

A gyakorlati megvalósításokban „n" értéke tipikusan 3-5 szokott lenni, attól függően mennyi tanítóadat állt rendelkezésre. Az n-gram közelítés (vagy Markov közelítés), ha jól választjuk meg az „n" értékét, aránylag kis hibával jár, mert egy szó felbukkanásának valószínűsége döntően az azt közvetlenül megelőző szavaktól függ. A becslő tényezők **trigram** (n=3) alkalmazásával a következő alakra egyszerűsödnek:

$$P(W) \approx P(w_1) \cdot P(w_2 \mid w_1) \cdot P(w_3 \mid w_1, w_2) \cdot P(w_4 \mid w_2, w_3) \cdot ... \cdot P(w_K \mid w_{K-2}, w_{K-1})$$

Ugyanez a valószínűség **bigram** (n=2) modellel közelítve:

$$P(W) \approx P(w_1) \cdot P(w_2 \mid w_1) \cdot P(w_3 \mid w_2) \cdot P(w_4 \mid w_3) \cdot ... \cdot P(w_K \mid w_{K-1})$$

Végül pedig **unigram** (n=1) modellel közelítve:

$$P(W) \approx P(w_1) \cdot P(w_2) \cdot P(w_3) \cdot P(w_4) \cdot ... \cdot P(w_K)$$

Bár már az 1-gram (unigram) közelítés is elnagyoltnak tűnik, az ún. **0-gram**-okat is értelmezzük. Ennek az a lényege, hogy minden szót egyforma valószínűségűnek tekintünk, és ezt a valószínűséget a következő egyszerű képlettel számoljuk:

$$P(w) = \frac{1}{|V|}$$

Ahol „|V|" a type-ok számát (szótárméret) jelöli. Tehát egyszerű egyenletesen szétosztjuk a valószínűségi tömeget a szótári elemek között.

Megjegyzés: Bár az n-gram modellek a gyakorlatban nagyon hatékonyak, vannak esetek, amikor nem kielégítő a pontosságuk. Ilyen például, amikor két nyelvtani kapcsolatban álló szó nagyon távol kerül egymástól, így kívül kerülnek a modellezés hatókörén. Például:

„a **repülőgép** miután elérte a megfelelő **sebességet** felszállt a kifutópályáról"

3.2 Paraméterbecslés

A feladat az, hogy a tanítókorpusz minden *„k"* szavára és annak minden előtörténetére határozzuk meg a $P(w_k \mid w_{k-(n-1)},..., w_{k-1})$ valószínűséget, amely alapján az adatbázis szótárából előálló minden szósorozat valószínűsége becsülhetővé válik. A tanítószövegből az n-gram modell paramétereit általában Maximum Likelihood (ML) becsléssel számítjuk:

$$P_{ML}(w_k \mid w_{k-(n-1)},..., w_{k-1}) = \frac{c(w_{k-(n-1)},..., w_{k-1}, w_k)}{c(w_{k-(n-1)},..., w_{k-1})}$$

Ahol *c*(…) a zárójelben található szósorozat előfordulási számát jelöli. Ez a számítás bigram esetben így mutat:

$$P_{ML}(w_k \mid w_{k-1}) = \frac{c(w_{k-1}, w_k)}{c(w_{k-1})}$$

Bár ez a kifejezés egyszerűen számítható, a gyakorlatban problémát okoz az **elégtelen mennyiségű tanítóadat**. Például hiába próbálnánk így egy olyan n-gram valószínűségét becsülni, amely nem, vagy csak nagyon kevésszer fordul elő a tanító-adatbázisban. Az ilyen becslési hibák kiküszöbölésére alkalmazunk a nyelvi modellezéskor simítási eljárásokat [1].

3.3 A nyelvi modell simítása

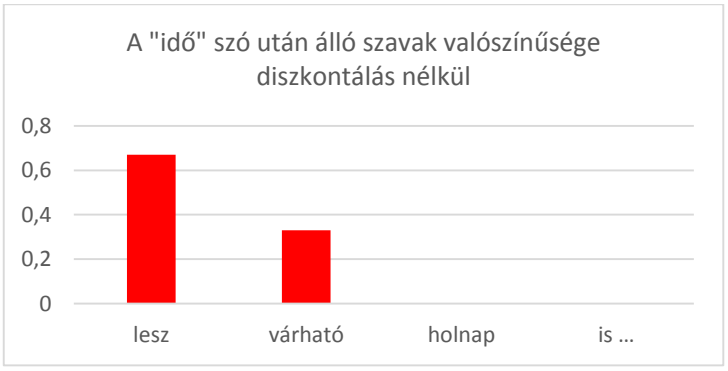
A nyelvi modellek simításának célja, hogy olyan n-gram-okra is képes legyen a modell 0-tól különböző valószínűséget visszaadni, melyek közvetlenül nem figyelhető meg a tanítószövegben. A simítási technikák alapvető eszköze a **diszkontálás** (discounting), melynek az a lényege, hogy a megfigyelhető n-gram minták valószínűségéből átcsoportosítunk egy részt a meg nem figyelhetők felé. Ezt egy példán keresztül lehet egyszerű bemutatni. Tegyük fel, hogy a következő szöveg alapján tanítunk nyelvi modellt:

holnap is erősen felhős idő várható
holnap is rossz idő lesz
felhős idő lesz holnap is

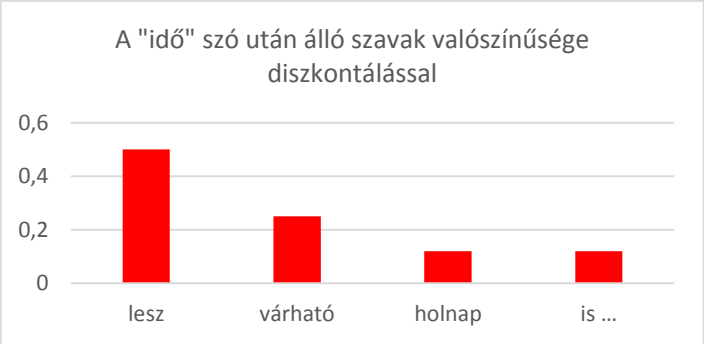
Vizsgáljuk meg, hogy az „idő” előtörténettel rendelkező bigramok milyen valószínűséget kaphatnak az ML becslés alapján:

P(lesz | idő) = c(idő lesz) / c(idő) = 2/3 = 0.67
P(várható | idő) = c(idő várható) / c(idő) = 1/3 = 0.33
P(holnap | idő) = c(idő holnap) / c(idő) = 0/3 = 0
…

Látható, hogy a sima ML becslés alapján csak a korpuszban előforduló n-gram-okra van 0-tól különböző valószínűség „idő” előtaggal:



A diszkontálás célja, hogy ezen a problémán segítsen és az „idő” szó után bármelyik szótári elemet berakva pozitív valószínűséget kapjuk:



A különböző nyelvi modellezési technikák főképp abban különböznek, hogy ezt a diszkontálást milyen algoritmussal végzik. A következőkben három módszert mutatunk be tömören, csak az alapötletre koncentrálva.

3.3.1 Add-1 (Laplace) simítás

Mind közül a legegyszerűbb megoldás, de mivel könnyű megérteni a működését, érdemes röviden kitérni rá. Az alapötlet annyiból áll, hogy minden n-gram elforduláshoz hozzáadunk 1-et, így az egyszer sem előforduló mintákhoz is kapunk papíron 1 mintát:

$$P_{Add-1}(w_k \mid w_{k-1}) = \frac{c(w_{k-1}, w_k)+1}{c(w_{k-1})+|V|}$$

A képletben V-vel a szótárat jelöljük, azaz a nevezőben a szótárérettel végzünk normalizálást. Ha ezt nem tennénk, akkor a w_{k-1} után álló szavak valószínűségének összege nagyobb lenne 1-nél.

Az Add-1 simításnak inkább elméleti jelentősége van, mert a gyakorlatban nem bizonyult túl hatékonynak. Ennek az az oka, hogy túlzottan nagy valószínűségi tömeget csoportosít át a megfigyelt n-gram-októl a meg nem figyeltekhez.

3.3.2 Good-Turing simítás

A Good-Turing [2] simítás már jóval bonyolultabb, mint az Add-1, így itt valóban csak az alapötletre koncentrálunk: úgy modellezünk a nem látott eseményeket, mint a ritkán látott eseményeket. Ez a gyakorlatban azt jelenti, hogy a Good-Turing simítás a nem látott n-gram-okhoz, az 1-szer előfordulók valószínűségét csoportosítja át, az 1-szer előfordulókhoz a 2-szer előfordulókat, stb. Ez már egy jól használható simítást eredményez, ezért például ez az alapértelmezett simítás a mérés során használt SRILM toolkit-ben [3].

3.3.3 Kneser-Ney simítás

A Kneser-Ney simítás az egyik legelterjedtebben alkalmazott módszer a beszédfelismeréshez készített nyelvi modellek területén. Alapvetően a korábbi simítások egy hibáját javítja ki, mely egy egyszerű példán keresztül érthető meg. Tegyük fel, hogy az „I love orange” mondat valószínűségét akarjuk kiszámítani bigram modell segítségével, azonban a „love orange” szókapcsolat nem fordult elő a tanítószövegben. Ez önmagában nem probléma, mert ilyenkor a modell az „orange” szó unigram valószínűsége alapján ad egy becslést arra, hogy P(orange | love). Történetesen azonban a tanítószöveg San Francisco-ról szól, ezért a „Francisco” szó sokkal gyakoribb benne, mint az „orange”. Ekkor a következő helyzet állhat elő:

P(I love Francisco) > P(I love orange)

Miközben nyilvánvaló, hogy hiába ritkább az „orange” szó az adott korpuszban a „Francisco”-nál, az „orange”sokkal többféle szó után állhat, mint a szinte csak a „San” után előforduló „Francisco”. A probléma feloldására a Kneser-Ney simításban bevezettek egy $P_{continuantion}$ nevű mértéket, mely ritka bigramoknál figyelembe veszi azt is, hogy hány bigram befejezése lehet az éppen vizsgált szó. Azaz mennyire valószínű, hogy egy korábban nem látott bigramot alkot.

4 Nyelvi modellek kiértékelése

A nyelvi modellek hatékonyságát egy tesztelési célokra elkülönített szövegkorpuszon szokás mérni. Két fontos mérőszámot ismertetünk ebben a fejezetben. Az egyik a **perplexitás**, a másik pedig a szótáron kívüli szavak (Out Of Vocabulary - **OOV**) aránya.

4.1.1 Perplexitás

A perplexitás egy mérőszám, melyet arra használunk, hogy egy nyelvi modell illeszkedését mérjük egy tesztszöveghez [1]. A nyelvi modellünk annál hatékonyabbnak tekinthető, minél nagyobb valószínűséggel jósolja meg a tesztszövegben előforduló szósorozatokat (mondatokat). A perplexitást ezért a tesztszöveg valószínűsége alapján számítjuk a következő módon:

$$PPL(W) = P(w_1, w_2, w_3..., w_K)^{-\frac{1}{K}} = \sqrt[K]{\frac{1}{P(w_1, w_2, w_3..., w_K)}}$$

Ahol „K” a token-ek száma a tesztszövegben.

Felmerül a kérdés, hogy miért nem használjuk közvetlenül P(W) értéket. A válasz az, hogy a P(W) egy rendkívül alacsony 0 és 1 közé eső szám lesz, ami az ember számára nem sok kézzelfogható jelentéssel bír. Ezzel szemben a fenti képlet nagyon is szemléletes mértéket eredményez. A következők miatt:

$$P(w_1, w_2, w_3..., w_K)^{\frac{1}{K}} = \sqrt[K]{P(w_1, w_2, w_3..., w_K)}$$

Negatív előjelű kitevő nélkül könnyen ráismerhetünk, hogy ez a kifejezés a nyelvi modell által egy szóra vonatkoztatott átlagos valószínűséget takarja. De miért szerepel a perplexitás képletében ennek a reciproka? Azért mert így egy kb. 10-1000 közé eső számot kapunk, amit sokkal könnyebb megjegyezni és összehasonlítani egymással. Ráadásul még egy szemléletes jelentéssel bír így a perplexitás: megmondja, hogy ha a vizsgált nyelvi modellben minden szó azonos valószínűségű lenne, akkor hány szó következhetne átlagosan egy másik szó után.

Például, ha az átlagos szóvalószínűség $P(W)^{1/K} = 0,01$, ez alapján a perplexitás $P(W)^{-1/K} = 100$, azaz 100 azonos valószínűségű szó jöhet minden szó után. Célunk tehát, hogy minél alacsonyabb perplexitással rendelkező nyelvi modellt tanítsunk, mely jól becsüli előre a tesztszöveget.

4.1.2 OOV arány

Emellett egy nyelvi modell jellemzésének fontos eszköze az **OOV** (Out of Vocabulary) arány, amely a tesztszövegben elforduló, de a nyelvi modell szótárában nem szereplő szavak aránya az összes tesztszövegben előforduló szóhoz képest. Az szótáron kívüli szavak arányának ismerete azért fontos,

mert ezeket a felismerő nem képes felismerni az adott nyelvi modellel, így egyfajta alsó becslését adja a szófelismerési hibának.

5 Gyakorlati technikák

Ebben a fejezetben két olyan további nyelvi modellezéssel kapcsolatos technikát ismertetünk, melynek ismeretére a mérés során építünk majd.

5.1 OOV modellezés

A szótáron kívüli szavak tehát azok a szavak, melyekkel a tesztelés során találkozik a modell, de a tanítószöveg nem tartalmazta őket. Ezekhez alapesetben nem képes 0-tól eltérő valószínűséget társítani egy nyelvi modell. Egy gyakorlatban használt technika, hogy a tanítószövegben bizonyos szavakat (például az 1-szer előfordulókat) egy speciális szimbólummal helyettesítjük (<unk>), majd az így módosított korpusz alapján tanítjuk a nyelvi modellt. A korábbi példaszöveg például így fest, ha az 1-szer előforduló szavakat lecseréljük <unk>-ra:

holnap is <unk> felhős idő <unk>
holnap is <unk> idő lesz
felhős idő lesz holnap is

Az ez alapján tanított nyelvi modell, már a szótáron kívüli szavakra is képes lesz 0-tól különböző értéket visszaadni. Mindezért cserébe viszont ki kell hagynunk ritka szavakat a modellünkből.

5.2 Szöveggenerálás

Érdekes játékként a mérés végén kipróbálhatjuk, hogy a nyelvi modell mennyire „beszéli jól” az adott nyelvet. A nyelvi modellek alapján ugyanis nem túl bonyolult szöveget generálni. Ha minden döntési helyzetben az adott kontextusban szóba jöhető szavak valószínűségét leképezzük egy véletlen szám értelmezési tartományára, akkor minden szó után egy véletlenszerű, de a nyelvi modell valószínűségi viszonyainak megfelelő szót kapunk. Ezzel a játékkal képet kaphatunk arról, hogy ezek az egyszerű statisztikai modellek mennyire képesek megragadni a nyelvek szabályszerűségeit.

6 Beugró kérdések

1. Mire használhatóak a statisztikai nyelvi modellek? Írjon példákat!
2. Mi jelent a token és a type fogalma?
3. Mi az a Zipf-görbe? Mit ábrázol, mik a tulajdonságai?
4. Mit jelent az n-gram közelítés?
5. Mi az n-gram modellek ML becslésének a lényege?
6. Miért van szükség simító eljárások használatára?
7. Hogyan definiáljuk egy nyelvi modell perplexitását? Mi ennek a szemléletes jelentése?
8. Mutasson be egy lehetséges módot az OOV szavak előfordulásának modellezésre!

7 Irodalom

[1] S. F. Chen and J. Goodman, “An empirical study of smoothing techniques for language modeling,” *Comput. Speech Lang.*, vol. 13, no. 4, pp. 359–393, Oct. 1999.

[2] I. J. Good, “The population frequencies of species and the estimation of population parameters,” *Biometrika*, vol. 40, no. 3–4, pp. 237–264, 1953.

[3] A. Stolcke, “SRILM – an extensible language modeling toolkit,” in *Proceedings International Conference on Spoken Language Processing*, 2002, pp. 901–904.